Integrity 10.9 CLI command list

im deletelabel
im deleteproject
im deletequery
im deletereport
im deletestate
im deletetrigger
im deletetype
im deleteuser
im diffsegments
im disconnect
im dynamicgroups
im echo
im editchart
im editcolumnset
im editdashboard
im editdynamicgroup
im editfield
im editgroup
im editissue
im editproject
im editquery
im editreport
im editstate
im edittrigger
im edittype
im edituser
im exit
im exportissues
im exporttranslations
im extractattachments
im extractwordtemplates
im fields
im getdbfile
im groups
im gui
im importcontent
im importgroup
im importissue
im importsegment
im importtranslations
im importuser
im incrementrevision
im insertsegment
im installsolution
im issues
im loadrc
im lock
im logging
im movecontent
im obtainadminlock
im printissue
im projects
im propagatetraces
im purgeauditlog
im putdbfile
im queries
im refnode
im relationships
im releaseadminlock
im removebaseline
im removecontent
im reports
im runchart
im rundashboard
im runreport
im runtrigger
im serveralerts
im servers
im setnotification
im setprefs
im setproperty
im settimeentries
im states
im timeentries
im toggleinclude
im triggers
im types
im unlock
im updateclient
im users
im viewadminlock

im viewauditlog
im viewchart
im viewcolumnset
im viewcp
im viewdashboard
im viewduplicates
im viewdynamicgroup
im viewfield
im viewgroup
im viewissue
im viewpendingimports
im viewprefs
im viewproject
im viewquery
im viewreport
im viewsegment
im viewserveralert
im viewsourcetraces
im viewstate
im viewtrigger
im viewtype
im viewuser
integrity about
integrity acv
integrity admin
integrity admingui
integrity changemksdomainuserpassword
integrity_clients installing
integrity createmksdomaingroup
integrity createmksdomainuser
integrity deletemksdomaingroup
integrity deletemksdomainuser
integrity disconnect
integrity echo
integrity editmksdomaingroup
integrity editmksdomainuser
integrity exit
integrity fetchviewset
integrity getdbfile
integrity gui
integrity licenses
integrity loadrc
integrity logging
integrity mksdomaingroups
integrity mksdomainusers
integrity_objects_server performance
integrity publishviewset
integrity putdbfile
integrity serveralerts
integrity servers
integrity setprefs
integrity setproperty
integrity setserveralert
integrity stats
integrity updateclient
integrity viewmksdomaingroup
integrity viewmksdomainuser
integrity viewprefs
integrity viewserveralert
integrity viewsets
options
preferences
si
si about
si acceptcp
si activatedevpath
si acv
si add
si addlabel
si addmemberattr
si addmemberfromarchive
si addproject
si addprojectattr
si addprojectlabel
si addprojectmetric
si addsandboxattr
si addsubproject
si admingui
si annotate
si appendcheckpointdesc
si appendrevdesc

si applycp
si archiveinfo
si calculateprojectmetrics
si checkpoint
si ci
si closecp
si co
si configuresandbox
si configuresubproject
si connect
si copypolicysection
si cpissues
si createcp
si createdevpath
si createmetricinfo
si createproject
si createsandbox
si createsubproject
si deactivatedevpath
si deletearchive
si deletelabel
si deletepolicysection
si deleteproject
si deleteprojectlabel
si deleterevision
si demote
si demoteproject
si diff
si diffiles
si discardcp
si disconnect
si drop
si dropdevpath
si dropmemberattr
si dropproject
si dropprojectattr
si dropsandbox
si dropsandboxattr
si echo
si edit
si editcp
si exit
si extenddevpath
si freeze
si getdbfile
si gui
si importsandbox
si integrations
si loadrc
si locate
si lock
si locks
si logging
si makewritable
si memberinfo
si merge
si mergebranch
si mergechilddevpath
si migrate
si mods
si move
si movesandbox
si movesubproject
si opencp
si primeproject
si print
si projectadd
si projectci
si projectco
si projectcpdiff
si projectinfo
si projectlocks
si projects
si promote
si promoteproject
si purgeauditlog
si putdbfile
si rejectcp
si rename
si report
si restoreproject

si restrictproject
si resync
si resynccp
si retargetsandbox
si revert
si revertcp
si revisioninfo
si rlog
si sandboxes
si sandboxinfo
si serveralerts
si servers
si setmemberrule
si setpolicysection
si setprefs
si setprojectdescription
si setproperty
si sharesubproject
si snapshot
si submit
si submitcp
si thaw
si unlock
si unlockarchive
si unrestrictproject
si updatearchive
si updateclient
si updaterevision
si viewauditlog
si viewcp
si viewcpentries
si viewcps
si viewhistory
si viewlabels
si viewlocks
si viewmetricsinfo
si viewnonmembers
si viewpolicysection
si viewpolicysections
si viewprefs
si viewproject
si viewprojecthistory
si viewprojectmetrics
si viewrevision
si viewsandbox
si viewserveralert
tm createresult
tm createverdict
tm deleteresult
tm deleteverdict
tm editresult
tm editverdict
tm extractattachments
tm purgeresults
tm resulteditor
tm resultfields
tm results
tm setprefs
tm setresults
tm stepresults
tm testcases
tm verdicts
tm viewprefs
tm viewresult
tm viewuntested
tm viewverdict

aa about

displays product information

Synopsis

`aa about [--[no]batch] [--cwd=`*`directory`*`] [(-g|--gui)] [ --quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)]`

Description

`aa about` displays version information for the Integrity release, build, service pack (if installed), API, and HotFixes (if installed).

Options

`aa about` takes a subset of the universal options available to `aa` commands. For example,

`aa about --gui`

displays the product information in a GUI window. See the options reference page for descriptions.

See Also

- Miscellaneous: ACL, diagnostics, options

aa acls

displays ACLs on the Integrity Server

Synopsis

`aa acls [--height=value] [--width=value] [-x value] [-y value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] ACL name...`

Description

`aa acls` gives you a list of ACLs on a specified Integrity Server.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--height=value`

  used with the `-g` or `--gui` options, specifies the height of the GUI window, in pixels; *value* must be a whole number.
- `--width=value`

  used with the `-g` or `--gui` options, specifies the width of the GUI window, in pixels; *value* must be a whole number.
- `-x value`

  used with the `-g` or `--gui` options, specifies the x location in pixels of the window.
- `-y value`

  used with the `-g` or `--gui` options, specifies the y location in pixels of the window.
- *ACL name...*

  specifies the ACLs to view.

Note

The following wildcards may be used to specify ACL names: "*" for multiple characters, and "?" for a single character.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa availablepermissions, aa groups, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa acv

displays running commands

Synopsis

`aa acv [--[no]batch] [--cwd=`*`directory`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)] [--height=`*`value`*`] [--width=`*`value`*`] [-x=`*`value`*`] [-y=`*`value`*`] [--forceConfirm=`*`[yes|no]`*`] [(-Y|--yes)] [(-N|--no)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [--[no]persist]`

Description

`aa acv` displays the currently running commands.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]persist`

  controls the persistance of CLI views.

See Also

- Miscellaneous: diagnostics, options

aa addaclentry

creates ACL entries on the Integrity Server

Synopsis

`aa addaclentry [--acl=name] [--[no|confirm]allowNonexistentPrincipal] [--[no|confirm]createAcl] [--[no]inheritParentPermission] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] ACL entry...`

Description

`aa addaclentry` creates new ACL entries on a specified Integrity Server. Optionally, this allows creation of new ACLs. For example,

```
aa addaclentry --acl=mks:si:project:id:MKSProjects
g=developers:AddMember,!BreakLock,CheckIn,
CreateProject,CreateSubproject,!DropMember,
FetchRevision,Lock,ModifyMemberRev
```

creates ACL entries for the ACL named *mks:si:project:id:MKSProjects*, specifically assigning permissions to a principal group named *developers*, and results in:

```
Processing ACL entry...
developers: AddMember allowed
developers: BreakLock denied
developers: CheckIn allowed
developers: CreateProject allowed
developers: CreateSubproject allowed
developers: DropMember denied
developers: FetchRevision allowed
developers: Lock allowed
developers: ModifyMemberRev allowed
```

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--acl=name`

  describes the ACL name.

  The following server-level ACLs are shipped by default: *mks* default server level ACL providing root level administrative control over the Integrity Server; *mks:aa* controlling login access to the ACLs; *mks:aa:mks* allowing read and update access for managing the ACLs; *mks:patch* controlling access to the administration of service packs; *mks:im* controlling access to managing workflows and documents; and *mks:si* controlling access to configuration management. For configuration management, you work with project ACLs that control the permissions for a particular directory. Working with member ACLs is also possible, which control permissions for specific files -- this would be for those rare circumstances where security on specific, individual files must be heavily controlled and where the administrative costs are known and accepted.

  The ACL name itself follows a specific hierarchical format:

  - The default server-level ACL is named *mks:si*. All project and member ACLs will inherit permissions from this one.
  - Project-level ACL names include a specific prefix, taking the format *mks:si:project:id:<project directory>*. The project directory is relative to the root of the Integrity Server.
  - Subproject ACLs have the same format as projects, simply appending the subdirectories using colons (:) instead of slashes.
  - Variant project ACLs take the format *mks:si:project:devpath:<devpathname>:id:<project directory>*. The variant project directory is specified including colons.
  - Member ACLs simply specify the file name in the ACL name, such as *mks:si:project:id:<project directory>:<member file name>*.
  - Archive ACLs specify the archive name in the ACL name, such as *mks:si:archive:<archive path>*.
- `--[no|confirm]allowNonexistentPrincipal`

  controls whether to allow creation of an ACL entry for a nonexistent principal. When creating ACL entries, the Integrity Server will look at your network authentication system and determine whether the principal is valid. If you specify `--allowNonexistentPrincipal`, any principal will be allowed.
- `--[no|confirm]createAcl`

  controls whether to create a new ACL if it does not already exist.
- `--[no]inheritParentPermission`

  specifies if to inherit permission of the parent configuration management project. This option to inherit parent permissions is only applicable to ACLs for configuration management (source) projects and development paths. For more information on permission inheritance, see the *PTC Integrity Server Administration Guide*.
- *ACL entry...*

  identifies one or more ACL entries you want to add. Separate multiple entries with a space.

  The entry consists of two main elements: *u=* or *g=* for identifying the *principal* (user or group), then a colon (:) followed by the comma-separated list of *permissions* you want to allow or deny. Denied permissions are preceded by an exclamation (!) mark.

  - The principal is listed as *u=* or *g=*, representing user or group. For example, *u=mkern* or *g=developers*.
  - The comma-separated list of permissions are taken from those listed on the ACL reference page. Denied permissions are preceded by an exclamation (!) mark. Remember to precede the list of permissions with a colon (:) to identify it with the principal. For example, *g=developers:AddLabel,!BreakLock*.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa deleteaclentry, aa groups, aa users
- Miscellaneous: ACL, diagnostics, options

aa admingui

launches the Integrity Administration client

Synopsis

`aa admingui [--height=value] [-x value] [-y value] [--[no]restoreDesktop] [--width=value] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`aa admingui` launches the Integrity Administration Client GUI. The client interface provides a single, centralized access point for the most common administration tasks. More specifically, you can manage Access Control Lists (ACLs); manage and distribute ViewSets; set up workflows and documents; configure configuration management policies; and send alert messages.

Note

The Integrity Administration Client GUI interface essentially acts as a container for several administration applications (workflow and document management, configuration management). From the client interface, you can specify one or more servers to administer, allowing you to have multiple Administration windows open. In addition, you can specify application preferences. Similar CLI commands that offer full administrative functionality include: `im admingui`, `si admingui`, and `integrity admingui`. These commands and `aa admingui` should not be confused with the `integrity admin` command, which launches an Administration window instance for a specific server. While you can administer all of the administrative applications for that server, you cannot connect to other servers or specify application preferences.

For information on using the Integrity Administration Client, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]restoreDesktop`

  restores the desktop from a previous session.
- `--height=value`

  specifies the height in pixels of the window.
- `--width=value`

  specifies the width in pixels of the window.

See Also

- Commands: aa about, aa disconnect, aa exit, aa loadrc, aa servers, aa updateclient
- Miscellaneous: options

aa availablepermissions

displays available permissions for ACLs on the Integrity Server

Synopsis

`aa availablepermissions [--height=value] [--width=value] [-x value] [-y value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] ACL name...`

Description

`aa availablepermissions` lists permissions available for specified ACLs on a specified Integrity Server. For example,

```
aa availablepermissions mks:si
```

results in:

```
mks:si
 AddMember
 AddProject
 AdminProxy
 AdminServer
 ApplyLabel
 ApplyProjectLabel
 BreakLock
 ChangePackageAdmin
 CheckIn
 Checkpoint
 ConfigureSubproject
 CreateDevpath
 CreateProject
 CreateSubproject
 DebugProxy
 DebugServer
 DeleteLabel
 DeleteProjectLabel
 DeleteRevision
 Demote
 DemoteProject
 DropDevpath
 DropMember
 DropProject
 DropSubproject
 EditPolicy
 FetchRevision
 Freeze
 Lock
 Login
 ModifyAuthor
 ModifyMemberAttribute
 ModifyMemberRev
 ModifyProjectAttribute
 MoveLabel
 MoveProjectLabel
 OpenProject
 Promote
 PromoteProject
 RestoreProject
 SelfReview
 ShareArchive
 SnapshotSandbox
 Thaw
 ViewPolicy
```

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- *`ACL name...`*

  identifies one or more ACLs you want permissions listed for. Separate multiple ACL names with spaces. For details on ACL names, see the ACL reference page.

Note

The following wildcards may be used to specify ACL names: "*" for multiple characters, and "?" for a single character.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa groups, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa connect

establishes a connection to an Integrity Server

Synopsis

`aa connect [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa connect` establishes a connection to an Integrity Server host. Most commands implicitly connect to the host, this does so explicitly. In fact, all the other `aa` commands call `aa connect` to establish the connection. The client supports multiple connections to multiple servers. You can use aa disconnect to disconnect from an Integrity Server host. For example:

```
aa connect --hostname=abcFinancial --port=7001 --user=jriley --password=jriley
```

connects to the *`abcFinancial`* server, logged in as *`jriley`*.

Options

This command takes the universal options available to all `aa` commands. See the options reference page for descriptions.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa disconnect, aa exit, aa servers
- Miscellaneous: ACL, diagnostics, options

aa copyacl

copies an ACL entry

Synopsis

`aa copyacl [--destAcl=`*`value`*`] [--sourceAcl=`*`value`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa copyacl` copies all ACL entries from the source ACL to the destination ACL. For example,

```
aa copyacl --sourceACL=mks:si:project:id:Projects:DemoApplication --destACL=mks:si:project:id:Projects:Application
```

copies the ACL for the *DemoApplication* configuration management project to the *Application* configuration management project.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--destAcl=`*`value`*

  specifies the destination ACL name to create. This option is mandatory.
- `--sourceAcl=`*`value`*

  specifies the source ACL name. This option is mandatory.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa groups, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa deleteacl

deletes ACLs on the Integrity Server

Synopsis

`aa deleteacl [--[no]confirm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`ACL name...`

Description

`aa deleteacl` deletes ACLs on a specified Integrity Server. For example,

```
aa deleteacl mks:si:project:id:Projects:DemoApplication
```

deletes the ACL for the *DemoApplication* configuration management project.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- *`ACL name...`*

  identifies one or more existing ACLs you want to delete. Separate ACL names with spaces. For details on ACL names, see the ACL reference page.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa deleteaclentry, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa deleteaclentry

deletes ACL entries on the Integrity Server

Synopsis

`aa deleteaclentry [--acl=name] [--[no]confirm] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] ACL entry...`

Description

`aa deleteaclentry` deletes ACL entries on a specified Integrity Server. For example,

```
aa deleteaclentry --acl=mks:si:project:id:MKSProjects:DemoApplication g=developers:AddMember,CheckIn
```

deletes the *AddMember* and *CheckIn* permissions for a group named *developers*, within the ACL for the *DemoApplication* configuration management project.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--acl=`*name*

  the ACL name.

  The following server-level ACLs are shipped by default: *mks* default server level ACL providing root level administrative control over the Integrity Server; *mks:aa* controlling login access to the ACLs; *mks:aa:mks* allowing read and update access for managing the ACLs; *mks:patch* controlling access to the administration of service packs; *mks:im* controlling access to workflow and document management; and *mks:si* controlling access to configuration management. For configuration management, you work with project ACLs that control the permissions for a particular directory. Working with member ACLs is also possible, which control permissions for specific files -- this would be for those rare circumstances where security on specific, individual files must be heavily controlled and where the administrative costs are known and accepted.

  Note

  The *mks* ACL cannot be deleted and the ACL entries cannot be removed.

  The ACL name itself follows a specific hierarchical format:

  The default server-level ACL is named *mks:si*. All project and member ACLs will inherit permissions from this one.

  Project-level ACL names include a specific prefix, taking the format *mks:si:project:id:<project directory>*. The project directory is relative to the root of the Integrity Server.

  Subproject ACLs have the same format as projects, simply appending the subdirectories using colons (:) instead of slashes.

  Variant project ACLs have a slightly different prefix, taking the format *mks:si:project:devpath:<devpathname>:id*.

  Member ACLs simply specify the file name in the ACL name, such as *mks:si:project:id:<project directory>:<member file name>*.

- *ACL entry...*

  identifies one or more ACL entries you want to delete. Separate multiple entries with a space.

  The entry consists of two main elements: *u=* or *g=* for identifying the *principal* (user or group), then a colon (:) followed by the comma-separated list of *permissions* you want to delete.

  The principal is listed as *u=* or *g=*, representing user or group. For example, *u=mkern* or *g=developers*.

  The comma-separated list of permissions are taken from those listed on the ACL reference page. Remember to precede the list of permissions with a colon (:) to identify it with the principal. For example, specifying *g=developers:AddLabel,BreakLock u=mkern:OpenProject* would delete *AddLabel* and *BreakLock* ACL permissions for a group named *developers*, and the *OpenProject* ACL permission for a user named *mkern*.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa addaclentry, aa deleteacl, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa deleteaclsubtree

deletes an ACL subtree

Synopsis

`aa deleteaclsubtree [--[no]confirm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`ACL name...`*

Description

`aa deleteaclsubtree` deletes all ACLs under the specified ACL subtree.

Related project ACLs may be structured in a subtree, for example:

```
mks:si:project:id:Aurora_Program
mks:si:project:id:Aurora_Program:DemoApp
mks:si:project:id:Aurora_Program:DemoDatabase
```

This nesting structure allows you to delete a subtree, which is essentially performing a batch deletion on a group of related ACLs.

Note

Caution: When deleting ACLs, ACL entries, or ACL subtrees, be sure your selection is correct. Deletion is permanent, so there is no way to undo the command, and you will have to recreate ACLs if you accidentally delete the wrong ACLs.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  confirms the delete confirmation policy.
- *`ACL name...`*

  ACL subtree to delete.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa groups, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa disconnect

disconnects from the Integrity Server

Synopsis

`aa disconnect [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--[no]confirm] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa disconnect` disconnects the client connection to the host Integrity Server. For example:

```
aa disconnect --hostname=abcFinancial --port=7001 --user=jriley
```

logs out *`jriley`* and disconnects from the *`abcFinancial`* server.

Note

When disconnecting a connection that is the current connection, all open client interfaces will use the first initialized connection as the new current connection.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  controls whether to implement the Integrity Server disconnection confirmation policy.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa connect, aa servers
- Miscellaneous: ACL, diagnostics, options

aa echo

echoes a string to user interface

Synopsis

`aa echo [(-?|--usage)] [(-F` *`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`*`string...`*

Description

`aa echo` echoes a string to the appropriate user interface when used in an event trigger.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- *`string...`*

  specifies a string to display in the appropriate user interface.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers
- Miscellaneous: ACL, diagnostics, options

aa editacl

edits an ACL on the Integrity Server

Synopsis

`aa editacl [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--acl=value] [--[no]inheritParentPermission] [selectionFile=value]`

Description

`aa editacl` edits an ACL on a specified Integrity Server. For example,

`aa editacl mks:si:project:id:Projects:DemoApplication`

edits the ACL for the *DemoApplication* configuration management project.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--acl=value`

  identifies the ACL you want to edit. For details on ACL names, see the ACL reference page.
- `--[no]inheritParentPermission`

  specifies if to inherit permission of the parent configuration management project. This option to inherit parent permissions is only applicable to ACLs for configuration management (source) projects and development paths. For more information on permission inheritance, see the *PTC Integrity Server Administration Guide*.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa editacl, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa exit

exits the current client session

Synopsis

`aa exit [--[no]abort] [--[no|confirm]shutdown] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa exit` exits the current client session. When you run any `aa` command from the CLI, or when you open the Authorization Administration GUI, you start a client session. Only one client session is running at a time, regardless of how many GUI windows you have open or how many CLIs you are using. To close the GUI you use the appropriate menu commands, and to close the CLI you can use the `aa exit` command. In both cases you may have the further option of shutting down the Authorization Administration client altogether, based on your preferences (see `aa setprefs`); if you do not, the client is still running and available for additional interaction. If you do shut down the client completely, then running any Authorization Administration command from the CLI or opening the Authorization Administration GUI starts a new client session.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]abort`

  controls whether to shut down any other Authorization Administration commands that may be running. Some commands allow you to specify a `--persist` option which keeps those commands active during a client session. Using `--abort` with `aa exit` is recommended for stopping all persistent views that have been specified with another command's `--persist` option.

- `--[no|confirm]shutdown`

  controls the shutting down of the Authorization Administration client without getting a prompt.

Note

Specifying `--noshutdown` with `aa exit` is essentially a non-operation: it does nothing.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa about, aa setprefs, aa viewprefs
- Miscellaneous: ACL, diagnostics, options

aa getdbfile

retrieves a configuration file from the database

Synopsis

`aa getdbfile [--encoding=value] [--output=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...`

Description

`aa getdbfile` retrieves a configuration file from the database, such as a ViewSet or item presentation template (IPT). Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing. Once you are finished editing these files, you can store them in the database using the `aa putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=value`

  specifies the code set to save the file in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--output=value`

  specifies the name of the file to store the output to on the local file system.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag-listdbfiles` or `im diag --diag-listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\`*`templatename`*`.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

aa groups

lists groups on the Integrity Server

Synopsis

`aa groups [--height=value] [--width=value] [-x value] [-y value] [--members] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]][principal name...]`

Description

`aa groups` lists groups on a specified Integrity Server. For example,

```
aa groups --members developers
```

list the members for the *developers* group, resulting in something similar to:

```
developers
 user administrator
 user mkern
```

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--members`

  displays members of the specified group(s).
- *principal name...*

  identifies one or more principals you want to list groups for. Separate principal names with spaces.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa availablepermissions, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa loadrc

loads the Authorization Administration client preferences file

Synopsis

`aa loadrc [--[no]merge] [--rc=`*`value`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa loadrc` loads the user's `IntegrityClient.rc` file, which contains your personal preferences for configuring Authorization Administration. If for some reason your personal `IntegrityClient.rc` file has changed, this command will reload your preferences. Your preferences file shouldn't change, unless you happen to copy someone else's file or if you happen to restore a backup that you made.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]merge`

  controls whether settings from the loaded file should be merged into existing preferences.
- `--rc=`*`value`*

  identifies the file containing settings for running Authorization Administration. The default is the `IntegrityClient.rc` file in your home directory.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa setprefs, aa viewprefs
- Miscellaneous: ACL, diagnostics, options

aa putdbfile

puts a configuration file into the database

Synopsis

`aa putdbfile [--encoding=value] [--input=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...`

Description

Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing using the `aa getdbfile` command. Once you are finished editing these files, you can store them in the database again using the `aa putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=value`

  specifies the code set the file is saved in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--input=value`

  specifies the name of the file on the local file system containing the input.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag=listdbfiles` or `im diag --diag=listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\templatename.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

aa serveralerts

displays Integrity Server alert messages for all currently connected servers

Synopsis

```
aa serveralerts [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]
```

Description

`aa serveralerts` displays Integrity Server alert messages for all servers that you are currently connected to. The alert message displays who sent the message, the server it came from, when it was sent, and the message. If you are not connected to any servers, a message informs you that there are no alert messages. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

Note

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

See Also

- Commands: aa viewserveralert, aa admingui, aa servers
- Miscellaneous: options

aa servers

displays the current connection to an Integrity Server

Synopsis

`aa servers [--[no]showVersion] [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [(-F file|--selectionFile=file)] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`aa servers` displays active server connections in the format *user@host_name:port*. In the current release of Authorization Administration, multiple connections to multiple Integrity Servers may be established at the same time.

The default server connection is indicated by *user@host_name:port (default)*.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showVersion`

  controls whether to show build version information for the connected server. The presentation of this information is in the format `[Build: 4.5.0.4652.7.1]`.

- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa connect, aa disconnect
- Miscellaneous: ACL, diagnostics, options

aa setprefs

sets Authorization Administration preferences

Synopsis

`aa setprefs [--command=value] [--[no]resetToDefault] [--[no]save] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] pref[=value]...`

Description

`aa setprefs` sets Authorization Administration preferences. These settings are used to determine default behaviors for other commands - each option on each command has a preference key associated with it. The aa viewprefs command lists the commands and preference keys. Changes to your preferences are either for the current client session (until aa exit is used) or may be permanently saved in your system's `home` directory, into a file named `IntegrityClient.rc`, using the `--save` option.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--command=value`

  identifies the command to be set. For an easy way to see a list of commands and values that may be set, simply type the aa viewprefs command, either piped through `|more` or redirected to a file, for example:

  ```
  aa viewprefs --global --showValidValues prefs.txt
  ```

  The commands and preference keys are also listed on the preferences reference page.

- `--[no]resetToDefault`

  controls whether to revert specified settings to the default values as shipped with Integrity Client. If specifying `--resetToDefault`, you must not specify *=value* for each preference.

- `--[no]save`

  controls whether changes should be permanently saved.

- `--[no]ask`

  controls prompts to the user for specific preferences. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the *pref=value* must supply one of the following four valid `ask` values:

  - *once*

    asks the user the first time only, and then uses the provided value every time after.

  - *never*

    never asks the user for a response, but uses the current setting (which may be specified by a preference).

  - *element-last*

    asks the user for each element of the selection, providing the most recently used value as the default.

  - *element-pref*

    asks the user for each element of the selection, resetting the default to the value specified by the preference.

  For example, to set the server host for aa connect to a specific host name, you specify something like:

  ```
  aa setprefs --command=connect
  server.hostname=specific.hostname.com
  ```

  but to set the preference to ask for a host name when using aa connect, you specify something like:

  ```
  aa setprefs --command=connect --ask
  server.hostname=element-last
  ```

- *--ui=[unspecified|gui|cli|api]*

  controls whether to apply the preference to the graphical user interface, the command line interface, the application programming interface, or when the interface is unspecified. By default, `--ui=`*unspecified* is used and applies any changes you make to all interfaces unless there is already an interface specific preference configured.

- These correlate to settings in the `IntegrityClient.rc` file, which can be seen as having the *gui.aa.*, *gui.api.*, or *cli.aa.* prefix, or simply the *aa.* prefix when it is unspecified.

- *pref[=value]...*

  identifies the preference string. If you specified the `--resetToDefault` option, then you only need to specify the preference name; otherwise specify a value for the preference. Use spaces to specify multiple preferences.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa loadrc, aa viewprefs
- Miscellaneous: diagnostics, options, preferences

aa setproperty

configures Integrity properties stored in the database

Synopsis

`aa setproperty [--comment=value] [--restoreDefault] [--value=value] [-?|--usage] [-N|--no] [-Y|--yes] [--hostname=server] [--port=number] [--user=name] [--password=password] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [-Ffile|--selectionFile=file] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...`

Description

`aa setproperty` configures Integrity properties stored in the database. Properties specify information that affects the operation of the Integrity Server, workflows and documents, and configuration management. Other properties are stored and configured in properties files on the server's file system. For a complete list of configurable properties and possible values, see the *PTC Integrity Server Administration Guide*.

Note

- Some properties require the server to be restarted for the changes to take effect.
- Access to configuring properties is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document properties, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management properties, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all properties.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--comment=value`

  specifies a comment associated with the change made to the property. While PTC recommends specifying a comment for troubleshooting purposes, a comment is optional.
- `--restoreDefault`

  restores the property to the default value.
- `--value=value`

  specifies the new value of the property.
- *`string...`*

  specifies the name of the property you want to configure.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa admingui, integrity gui
- Miscellaneous: diagnostics, options, preferences

aa updateclient

updates the Integrity Client with a service pack

Synopsis

`aa updateclient [--[no|confirm]download] [--[no|confirm]shutdown] [--[no|confirm]rollback] [--[no|confirm]rollbackshutdown] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`aa updateclient` updates the Integrity Client with a service pack from the server if one is available. A service pack may be designated as required to address a known issue, or may provide enhancements. Client side service pack numbers are designated with a ID, for example, `C04030003`.

Options

This command takes some of the universal options available to `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]download`

  automatically downloads a service pack if one is available.
- `--[no|confirm]shutdown`

  automatically shutdowns the client if a service pack is downloaded.
- `--[no|confirm]rollback`

  automatically initiates a service pack rollback, if required to connect to the Integrity Server.
- `--[no|confirm]rollbackshutdown`

  automatically shutdowns the client if a service pack rollback is initiated.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa groups, aa users, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa users

lists users on the Integrity Server

Synopsis

`aa users [--groups] [--height=value] [--width=value] [-xvalue] [-yvalue] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [principal name...]`

Description

`aa users` lists users on a specified Integrity Server. For example,

```
aa users --groups administrator
```

list the groups that the *administrator* user belongs to, resulting in something such as:

```
administrator
 Administrators
 Developers
 Managers
 SysAdmins
```

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--groups`

  displays group(s) that the specified member belongs to.
- *principal name...*

  identifies one or more principals you want to list users for. Separate principal names with spaces.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa availablepermissions, aa groups, aa viewacl
- Miscellaneous: ACL, diagnostics, options

aa viewacl

displays permissions and principals for a specified ACL on the Integrity Server

Synopsis

`aa viewacl [--acl=name] [--height=value] [--width=value] [-x value] [-y value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]showHeader]`

Description

`aa viewacl` displays a list of principals and permissions for a specified ACL on a specified Integrity Server. For example,

```
aa viewacl --acl=mks:si
```

results in:

```
 AddMember allowed group everyone
 AddProject allowed group everyone
 ApplyLabel allowed group everyone
 ApplyProjectLabel allowed group everyone
 BreakLock allowed group everyone
 ChangePackageAdmin allowed group everyone
 CheckIn allowed group everyone
 Checkpoint allowed group everyone
 ConfigureSubproject allowed group everyone
 CreateDevpath allowed group everyone
 CreateProject allowed group everyone
 CreateSubproject allowed group everyone
 DeleteLabel allowed group everyone
 DeleteProjectLabel allowed group everyone
 DeleteRevision allowed group everyone
 Demote allowed group everyone
 DemoteProject allowed group everyone
 DropDevpath allowed group everyone
 DropMember allowed group everyone
 DropProject allowed group everyone
 DropSubproject allowed group everyone
 FetchRevision allowed group everyone
 Freeze allowed group everyone

 Lock allowed group everyone
 Login allowed group everyone
 ModifyAuthor allowed group everyone
 ModifyMemberAttribute allowed group everyone
 ModifyMemberRev allowed group everyone
 ModifyProjectAttribute allowed group everyone
 OpenProject allowed group everyone
 Promote allowed group everyone
 PromoteProject allowed group everyone
 RestoreProject allowed group everyone
 SelfReview allowed group everyone
 ShareArchive allowed group everyone
 SnapshotSandbox allowed group everyone
 Thaw allowed group everyone
```

and

```
aa viewacl --acl=mks:aa:mks
```

results in:

```
Read allowed group everyone
Update allowed group everyone
```

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--acl=name`

  identifies the ACL name you want to view. To get a list of ACL names on the Integrity Server, use the aa acls command. For details on ACL names, see the ACL reference page.
- `--[no]showHeader`

  specifies if to display the header part of the view.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa acls, aa availablepermissions, aa groups, aa users
- Miscellaneous: ACL, diagnostics, options

aa viewprefs

displays preferences for Authorization Administration commands

Synopsis

`aa viewprefs [--[no]global] [--command=value] [--[no]showValidValues] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`aa viewprefs` displays preferences for Authorization Administration commands.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions. For an easy way to see a list of commands and values that may be set, simply type the `aa viewprefs` command, either piped through `|more` or redirected to a file, for example:

```
aa viewprefs --global --showValidValues >prefs.txt
```

Alternatively, the `--gui` option presents a common preferences dialog box that lets you view and configure the preferences for Authorization Administration; workflows and documents; configuration management; Integrity Administration Client; and the Integrity Client.

- `--[no]global`

  controls whether to view all preferences.
- `--command=value`

  identifies the command preference to be viewed.
- `--[no]showValidValues`

  controls whether to list valid values for the preferences.
- `--[no]ask`

  controls whether to view the ask control over the preference options. See the description for `--[no]ask` on the aa setprefs command.
- `--ui=[unspecified|gui|cli|api]`

  controls whether to apply the preference to the graphical user interface, the command line interface, the application programming interface, or when the interface is unspecified. By default, `--ui=unspecified` is used and applies any changes you make to all interfaces unless there is already an interface specific preference configured.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: aa loadrc, aa setprefs
- Miscellaneous: ACL, diagnostics, options, preferences

aa viewserveralert

displays Integrity Server alert messages for a target server and all related servers

Synopsis

`aa viewserveralert [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`aa viewserveralert` displays Integrity Server server alert messages for a target sever and all related servers, for example, a configuration management server and a proxy server. The alert message displays who sent the message, the server it came from, when it was sent, and the message. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.
- A connection to the target server is required for the `aa viewserveralert` command to display alert messages.

Options

This command takes the universal options available to all `aa` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=server`

  specifies the host name of the target Integrity Server to retrieve alert messages from.
- `--port=number`

  specifies the port of the target Integrity Server to retrieve alert messages from.

See Also

- Commands: aa serveralerts, aa admingui, aa servers
- Miscellaneous: options

diagnostics

applicable to all CLI commands

Description

The exit status values for CLI commands (`si`, `im`, `tm`, `aa`, `integrity`) can be used by event triggers for automating processes with Integrity. This reference page is provided as a guide to the exit status values you may see.

Diagnostics

Possible exit status values for CLI commands are:

- 0

  Successful completion.

  or

  No differences between the files being compared (using `si diff`).
- 1

  Command usage error.
- 2

  Command was canceled by user. This does not include cancellations using CTRL-C, which overrides this exit status value. In those cases, the return code will be `130`.
- 3

  Invalid element in the selection for the command.
- 4

  Sandbox specified was ambiguous (using an `si` command). Command not executed.
- 5

  Unable to create or utilize the selection for the command.
- 6

  Unable to continue with the selection for the command because the program cannot find the next element.
- 10

  Connection failed: a network error caused the command to terminate.
- 16

  diff compared the files and found them to be different (using `si diff`).
-
- 17

  Failure due to any of the following (using `si diff`):

  - invalid command line argument
  - cannot open one of the input files
  - out of memory
  - read error on one of the input files
  - more than LINE_MAX characters between newlines
- 19

  At least one of the files is a binary file containing embedded NUL (\0) bytes (using `si diff`).
- 23

  Command succeeded, but with warnings.
- 128

  General command failure.
- 130

  Command was canceled by the user using CTRL-c.
- 255

  Unknown exception or error code.

See Also

- Miscellaneous: ACL, options, preferences

im intro

introduction to reference pages

Description

A description of an individual topic (for example, a command) is called the `reference page` for that topic, even if it is actually several pages long.

There are three alternatives for accessing the reference pages to each command through the CLI man command.

First, you can type the `im` prefix and the command together as one word. Second, you can type the `im` prefix and the command with an underscore between them. Third, you may quote the `im` prefix and the command, with a space in the middle. For example:

```
man imabout
```

```
man im_about
```

```
man "im about" (Windows client only)
```

See the reference page for the man command itself, by typing `man man`, to find out more details. You can also use the `-h` option which allows you to view the reference pages in HTML Help format (Windows client only).

Note

To view Integrity online reference pages (CLI commands) in HTML Help, you must have Microsoft Internet Explorer 8.0 or higher installed. PTC recommends Microsoft Internet Explorer 8.0 or higher to avoid any problems with HHCTRL.OCX.

This reference page describes the parts of a reference page with examples taken from real Integrity reference pages.

The following sections discuss the various elements of a reference page.

Name

The `NAME` section provides the name of the command and a brief functional description.

Synopsis

In the reference page for a command, the `SYNOPSIS` section provides a quick summary of the command's `format`. For example, here is the synopsis of the im createissue command.

```
im createissue [--[no]showWorkflow] [--addAttachment=value] [--addRelationships=value] [--type=type] [--field=value] [--
richContentField=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--
selectionFile=value)][--quiet] [--settingsUI=[gui|default ]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [--forceConfirm=[yes|no]]
```

The synopsis takes the form of a command line as you might type it into the system; it shows what you can type in and the order you should do it in. The parts that are enclosed in square brackets are *optional*; you may omit them if you choose. Parts that are not enclosed in square brackets must be present for the command to be correct.

The synopsis begins with the name of the command itself. The workflow and document commands all include the *im* prefix. In Integrity documentation, command names are always written in `bold Courier` font.

After the command name comes a list of options. A typical Integrity command option consists of either a single dash (-) followed by a single character, usually an uppercase or lowercase letter; or it may consist of a double dash (--) followed by a multi-character option name. Often there are single-character and multi-character options that do the same thing. The multi-character strings are not case sensitive, but are shown in mixed case to facilitate readability. For example, you might have `-?` or `--usage`.

Note

If you do not specify any options when you type an *im* command, Integrity prompts you to fill out the values for the mandatory options.

To view a list of all available im CLI commands, enter `im`.

The synopsis line shows options in `bold Courier` font.

In some cases, *value* provides extra information for using an option. For example, the im editissue command allows you to edit one or more Integrity issues; here is the command's synopsis:

```
im editissue [--[no]showWorkflow] [--[no]batchEdit] [--query=[user:]query] [--removeAttachment=value] [--removeRelationships=value]
[--addAttachment=value] [--addRelationships=value] [--field=value] [--richContentField=value] [--hostname=value] [--port=value] [--
password=value] [--user=value] [(-?|--usage)][(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]]
[--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] issue id...
```

In this example, note the option

```
--query=query
```

This option tells the im editissue command to select the issues returned in the specified query for editing. In a command synopsis, anything appearing in *italics* is a *placeholder* for information that you are expected to supply. Sometime after the synopsis, the reference page explains what kind of information is expected in place of the placeholder.

When you specify a value for an option, such as `--query=query`, values that contain spaces must be enclosed in double quotes, for example, `--query="Cosmos Defects"`. However, since the query `Defects` does not contain spaces, `--query=Defects` is acceptable. Values that contain special characters must be enclosed in double quotes.

The end of the im editissue synopsis is

```
issue...
```

Since there are no square brackets around the list, in this example, it is mandatory.

The ellipsis means one or more issue IDs. The ellipsis (`...`) stands for repetitions of whatever immediately precedes it. Most Integrity commands allow you to specify lists of multiple issues using spaces between them.

See the options reference page for more details on general and universal options that apply to CLI commands.

The order of issues on the command line is important. When you type in a command line, you should specify the parts of the command line in the order they appear in the command synopsis. The exceptions to this are options marked with a - or a --; they do not have to be given in the exact order shown in the synopsis. However, all the - or -- options must appear in the correct area of the command line. For example, you can specify

```
im editissue --field=State=Verified --addRelationships=42,45 32
im editissue --addRelationships=42,45 --field=State=Verified 32
```

but you will not get correct results if you specify

```
im editissue 32 --field=State=Verified --addRelationships=42,45
im editissue --field=State=Verified 32 --addRelationships=42,45
```

and so on.

Description

The DESCRIPTION section outlines what the command does and how each option works.

Inside the DESCRIPTION section, the names of files and directories are written in `normal Courier` font. The names of environment variables are written in *`italic Courier`* font.

See Also

The SEE ALSO section refers to other reference pages that may contain information relevant to the reference page you have just read.

im about

displays product information

Synopsis

`im about [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]batch] [--cwd=`*value*`] [( -?|--usage)]`

Description

`im about` displays version information for the Integrity release, build, service pack (if installed), API, and HotFixes (if installed).

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

See Also

- Miscellaneous: options

im acv

displays running commands

Synopsis

`im acv [--[no]batch] [--cwd=`*directory*`] [(-g|--gui)] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`] [(-?|--usage)] [--height=`*value*`] [--width=`*value*`] [-x=`*value*`] [-y=`*value*`] [--forceConfirm=`*[yes|no]*`] [(-Y|--yes)] [(-N|--no)] [(-F`*file*`|--selectionFile=`*file*`)] [--[no]persist]`

Description

`im acv` displays the currently running commands.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]persist` controls the persistance of CLI views.

See Also

- Miscellaneous: diagnostics, options

im addlabel

adds a label to an issue based on a current or historical date

Synopsis

```
im addlabel [--[no]movelabel] [--asOf=[<date>|label:<label>] [--comment=value] [(L=label|--label=label)][--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] issue id...
```

Description

`im addlabel` assigns a label to an Integrity issue.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]movelabel`

  determines that if the label exists, it can be moved using `--asOf`. To move the label "Release ABC" to an issue based on a specific time you would enter

  ```
  im addlabel --movelabel -L "Release ABC" --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

- `--asOf=[<date>|label:<label>]`

  allows you to add or move a label as of a specific date or label. For example, to add the label with a specific date, type

  ```
  im addlabel -L "Release ABC" --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If a value is not provided the label is added as of the server's current time. This field is optional.

- `--comment=value`

  specifies a comment to associate with the label.

- `-L label`
- `--label=label`

  identifies a label name for an issue. To add the label "Release ABC " to an issue you would enter

  ```
  im addlabel -L "Release ABC"
  ```

- *issue id...*

  identifies the ID of the issue(s) associated with the label you want to add or move.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID*`-`*major*`.`*minor*, for example, `184-1.2`.

See Also

- Commands: im deletelabel
- Miscellaneous: ACL, diagnostics, options, preferences

im admingui

launches the Integrity Administration Client

Synopsis

```
im admingui [--height=value] [--[no]restoreDesktop] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [-xvalue] [-yvalue] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]
```

DESCRIPTION

`im admingui` launches the Integrity Administration Client GUI. The client interface provides a single, centralized access point for the most common administration tasks. More specifically, you can manage Access Control Lists (ACLs), manage and distribute ViewSets, set up workflows and documents, configure configuration management policies, and send alert messages.

Note

The Integrity Administration Client GUI interface essentially acts as a container for several administration applications (workflow and document management, configuration management). From the client interface, you can specify one or more servers to administer, allowing you to have multiple Administration windows open. In addition, you can specify application preferences. Similar CLI commands that offer full administrative functionality include: `aa admingui`, `si admingui`, and `integrity admingui`. These commands and `im admingui` should not be confused with the `integrity admin` command, which launches an Administration window instance for a specific server. While you can administer all of the administrative applications for that server, you cannot connect to other servers or specify application preferences.

For information on using the Integrity Administration Client, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]restoreDesktop`

  restores the desktop from a previous session.
- `--height=value`

  specifies the height in pixels of the window.
- `--width=value`

  specifies the width in pixels of the window.

See Also

- Commands: integrity about, integrity disconnect, integrity exit, integrity gui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: options

im analytics

calculates a computed field within a specified timeframe, storing the value in the issue history

Synopsis

`im analytics [--recomputeHistory] [--endTime=value] [--field=field] [--increment=value] [--incrementType=value] [--startTime=value] [--[no]clearHistory] [--[no]confirm] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [-g|--gui] [(-N|--no)] [--quiet] [(-Y|--yes)] [--[no]batch] [--settingsUI=[gui|default]] [--cwd=directory] [--status=[none|gui|default]] [--forceConfirm=[yes|no]]`

Description

When you create a computed field, you typically configure it to calculate at a specific time in the future, storing the value in the history of each issue containing the computed field. `im analytics` calculates what the value of the computed field would have been at specific times and stores the value in the history of each issue containing the computed field. This command is useful for discovering historical trends, for example, you could determine how long a Defect issue typically remained in the In Development state in a current project compared to that of a previous project. For more information on computed fields, see the im createfield command.

Key Considerations:

- This command is required only if a new calculation if required over existing historical data.
- To use the `im analytics --recomputeHistory -g` command, you must either be an administrator or have type administrator permissions for all types that the specified field displays in.
- Depending on the options you specify, this command may delete or modify issue history.
- Depending on the following factors, this command may take a long time to complete:
  - the complexity of the underlying computed expression
  - the number of issues containing the specified computed field
- The `im analytics --recomputeHistory -g` command renders issue data as it existed at the specified time. For non-historied data source (attachments, change packages containing entries that have moved, permissions, and administrative objects), the `im analytics --recomputeHistory -g` command approximates the historical data. To render non-historied data sources as they were known at a specific time, PTC recommends defining computed fields that use the `--staticComputation` option, which captures historical values.
- Computed fields are calculated in the order that they appear in the issue; however, if a computed field depends on the value of another computed field, it is calculated after the computed field containing the dependant value. Depending on the number of issues in your database and the number of issues containing computed fields that contain dependencies, it may take a long time to calculate the value of the computed fields.
- This command may be cancelled at any time; however, it will not take effect until the specified interval calculation has completed. If you cancel this command, previous field values remain stored in the database. If the clear history option is selected and you cancel this command, issue history is still cleared.

Example

This example illustrates how to determine the historical trend for defects in projects.

1. Create a new type to represent projects, for example, Project.
2. Create some Project issues to represent projects.
3. From the command line, create a computed field, for example, Defect Count, that calculates the number of defects related to each project:

   ```
   im createfield --type integer --computation='Query(\u201cFinancial Toolkit Defects: In Dev\u201d, Project, count()) --name 'Defect Count'
   ```
4. Make the `Defect Count` field visible in the Project type only.
5. From the command line, project the `Defect Count` field back in time:

   ```
   im analytics --recomputeHistory --starttime=06/01/2002 --endtime=08/24/2007 --increment=1 --incrementtype=week --field='Defect Count'
   ```
6. Create a query that lists all Project issues, for example, `All Projects`.
7. From the command line, create a chart:

   ```
   im createchart --trendstep=week --charttitle='Defect Count'--startdate=06/01/2002 --enddate=02/01/2007 --computations='\u201dDefect Count\u201d'--query 'All Projects'--charttype='Issue Fields Trend'--issueidentifier {Summary} -name xx
   ```

Note

To use the `im analytics` command, you must either be an administrator or have type administrator permissions for all types that the specified field displays in.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--recomputeHistory`

  performs the recompute history operation, calculating what the value of the specified computed field would have been at specific times and storing the value in the history of each item containing the computed field.
- `--endTime=value`

  specifies the ending calculation date of the computed field using the mm/dd/yyyy hh/mm/ss format, where mm/dd/yyyy hh/mm/ss is the month, day, year, hour, minutes, and seconds. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM.
- `--field=field`

  specifies the computed field to calculate.
- `--increment=value`

  specifies an integer that represents an increment unit. For example, if you specify an increment unit of 1 and an increment type of day, the computed field calculates once a day.
- `--incrementType=value`

  specifies the size of the increment unit. Acceptable sizes are: day, week, month, or year.
- `--startTime=value`

  specifies the starting calculation date of the computed field using the mm/dd/yyyy hh/mm/ss format, where mm/dd/yyyy hh/mm/ss is the month, day, year, hour, minutes, and seconds. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM.
- `--[no]clearHistory`

  clears previous history entries for the selected field between the specified times.
- `--[no]confirm`

  confirms if to calculate the computed field.

See Also

- Commands: im createfield, im editfield
- Miscellaneous: options

im baseline

create or move an existing baseline

Synopsis

```
im baseline[--[no]move] [--asOf=[<date>|label:<label>] [--[no]subSegment] [(L=label|--label=label)] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] segment id...
```

Description

`im baseline` creates a baseline or moves an existing baseline. A baseline is a meaningful date and time for a document, marked with a label. For example:

```
im baseline -L "Release ABC" --asOf="January 8, 2007 10:00:00 AM EST" 123
```

adds the baseline label "Release ABC" to document ID 123 as of January 8, 2007 10:00:00 AM EST.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]move`

  if the baseline exists, determines whether it will be moved to the new baseline specified using `--asOf`.

- `--[no]subSegment`

  specifies to baseline the referenced subsegment instead of the owning segment. To learn about referenced and included subsegments, see your *PTC Integrity User Guide*.

- `--asOf=<date>|label:<label>`

  allows you to create a baseline or move a baseline as of a specific date or label. This field is optional. If a value is not provided the label is added as of the current time.

- `-L label`
- `--label=label`

  specifies a label for the baseline. Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.

  Note

  Labels that include spaces must be enclosed by quotes.

- `segment id...`

  the ID of the segment and/or node you want to baseline. To baseline a subsegment you must specify `--subsegment`.

See Also

- Commands: im removebaseline
- Miscellaneous: diagnostics, options, preferences

im branchsegment

branches a segment

Synopsis

`im branchsegment [--asOf=<date>|label:<label>] [--project=value] [--hostname=server] [--port=number] [--parentID=project] [--insertLocation[=number|first|last|before:name|after:name] [--[no]subSegment [--[no]recurseInclude] [--[no]recurseReference] [--[no]swap] [--[no]includeContent] [--[no|confirm]large] [--[no]branch] [--threshold=value] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] segment id...`

Description

`im branchsegment` branches a segment. A segment can be a document root or a subsegment. Branching creates a copy of the segment and adds it to the project history. You can branch a segment as of a specific date or label. For example:

```
im branchsegment --asOf="January 8, 2012 10:00:00 AM EST" 123
```

branches segment 123 as of January 8, 2012.

For more information on branching, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--asOf=<date>:label:<label>`

  allows you to branch the segment as of a specific date or label.

  If a value is not provided, the segment is branched as of the server's current time. This field is optional.
- `--project=project`

  the project you want to assign the branched segment to.

Note

To restrict access to specific information in documents, all documents are subject to visibility rules set by your administrator. You must have visibility to the project you want to assign the branched segment to. For more information, contact your administrator.

- `--[no]subSegment`

  branches the subsegment if the selected node references a subsegment.
- `--[no]recurseInclude`

  specifies to branch all included subsegments.
- `--[no]recurseReference`

  specifies to branch all inserted subsegments.
- `--[no]swap`

  specifies to branch the referenced subsegment instead of the owning segment and swaps in the new branch. Specified with `--subSegment`. The default value is `--noswap`.
- `--[no]includeContent`

  includes all content from the original segment.

  When you specify *--noincludeContent*, no content is copied from the original segment. Only the segment's structure is copied.

  You cannot use the *--noincludeContent* option with the *--[no]recurseInclude* or *--[no]recruseReference* options.
- `--[no]branch`

  indicates that the copy operation you are performing will be a branch. Branching creates a copy of the segment and adds it to the project history. You can branch a segment as of a specific date or label.
- `--[no|confirm]large`

  species to display a message to users confirming that they are copying a large number of content issues. The default is 50 and can be modified using the `--threshold` option.

Note

A confirmer message displays by default when you run this command from the API. You must specify `--noconfirmlarge`if you want the command to complete.

- `--threshold=value`

  specifies the number of content issues to allow users to copy. The default threshold is 50. Users will be prompted to confirm a copy operation of more than 50 issues. If you increase the default threshold, you can specify `--[no]confirmLarge` option to either notify users or bypass notification.
- `--parentID=value`

  the ID of the parent segment that the segment will be added to. A parent ID is optional.
- `--insertLocation[=number|first|last|before:name|after:name]`

  determines where the segment should go in the parent segment's structural relationship list. You must specify a `--parentID`. The options are as follows:

  `first` inserts the segment at the beginning of the list.

  `last` inserts the segment at the end of the list.

  `before:name` inserts the segment before the specified ID.

  `after:name` inserts the segment after the specified ID.

  [0,...] inserts the segment at the specified location. If a negative is specified, the segment is inserted at the beginning of the list. If the number specified is too large, the segment is inserted at the end of the list. The default insert location is last in the structural relationship list.
- *segment id...*

  the ID of the segment and/or node you want to branch. To branch a subsegment you must specify `--subsegment`.

Note

To restrict access to specific information in documents, all documents are subject to visibility rules set by your administrator. You must have visibility to the segment and/or node you want to branch. For more information, contact your administrator.

See Also

- Commands: im createsegment, im viewsegment, im insertsegment
- Miscellaneous: diagnostics, options, preferences

im changesegmentproject

changes the project defined for an entire segment, including all referenced and/or included subsegments

Synopsis

```
im changesegmentproject [--project=project][--[no]recurseInclude][--[no]recurseReference][--[no]subSegment][--user=name][--hostname=server][--password=password][--port=number][--quiet][(-?|--usage)][(-F file|--selectionFile=file)][(-N|--no)][(-Y|--yes)][--[no]batch][--cwd=directory][--forceConfirm=[yes|no]][-g|--gui][--settingsUI=[gui|default]][--status=[none|gui|default]]segment id...
```

Description

`im changesegmentproject` changes the project for an entire segment, including all referenced and/or included subsegments. The project on every node in a segment must be the same as the project on the segment root.

For example:

```
im changesegmentproject --project=/Test2 4
```

Note

You cannot modify projects for shared items in Integrity. Running this command on a shared item has no effect.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--project=`*`project`*

  specifies the project you want to change to for the segment and/or subsegment.

  Note

  To restrict access to specific information in documents, all documents are subject to visibility rules set by your administrator. You must have visibility to the project you want to change to for the segment and/or subsegment. For more information, contact your administrator.

- `--[no]recurseInclude`

  specifies that the project will be changed on all nodes included or referenced by the Segment ID. If this is not specified, only the segment root project changes.

- `--[no]recurseReference`

  specifies to change the projects associated with all referenced included subsegments parented by the specified segment ID. To learn more about included subdocuments, see the *PTC Integrity User Guide*.

- `--[no]subSegment`

  changes the subsegment's project if the node specified references a subsegment. `--subSegment` has no effect if none of the selections are a reference to a segment root. If the node references a shared issue or the option is clear, the segment root is changed.

- *`segment id...`*

  the ID of the segment and/or nodes containing the project you want to change.

  Note

  To restrict access to specific information in documents, all documents are subject to visibility rules set by your administrator. You must have visibility to the segment and/or node containing the project you want to change. For more information, contact your administrator.

See Also

- Commands: im viewsegment, im branchsegment, im insertsegment, im importsegment
- Miscellaneous: diagnostics, options, preferences

im charts

displays a list of charts

Synopsis

```
im charts[--fields=field1[:width1],field2[:width2]...][--fieldsDelim=value][--[no]showHistory][--[no]showReferences][--height=value]
[--width=value][-x value][-yvalue ][--user=value] [-- hostname=value ] [--password=value][--port=value][(-?|--usage)] [(-g|--gui)]
[(-F value |--selectionFile= value )] [--quiet] [--settingsUI=[ gui |default ]] [--status=[ none | gui | default ]] [(-N|--no)][(-Y|-
-yes)] [--[no]batch] [--cwd= value ] [--forceConfirm=[yes | no]] chart...
```

Description

`im charts` displays a list of Integrity charts. By default, the command displays all charts that are currently shared with you.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fields=` *`field1[:width1],field2[:width2]...`*

  specifies the chart fields to display and the width of each field in characters. If the output is directed to the GUI, the width is specified in pixels.

  For example,

  ```
  im charts --fields=name,chartType,createdBy --fieldsDelim=, "Defect Analysis By Project"
  ```

  displays the name, chart type and creator for the Defect Analysis By Project chart, with each field separated by a comma.

  The chart fields you can specify are:

  `chartType`

  displays the type of chart.

  `createdBy` displays the name of the user who created the chart.

  `description`

  displays a description of the chart.

  `graphStyle`

  displays the style of graph used for the chart.

  `lastModified`

  displays the date the chart was last modified.

  `name`

  displays the name of the chart.

  `query`

  displays the name of the query the chart is based on.

  `shareWith`

  displays the users and groups that the chart is shared with.

  `sharedGroups`

  displays the groups that the chart is shared with.

  `id`

  displays the database ID of the chart. This is for PTC - Integrity Support only.

  `references`

  displays all system provided and user objects that reference the chart.

  `isAdmin`

  displays whether the chart is a system provided object.
- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between fields.
- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the chart.
- `--[no]showReferences`

  specifies whether to display all system provided and user objects that reference the chart.
- *`chart`*

  identifies the names of the charts to view.

See Also

- Commands: im copychart, im createchart, im deletechart, im editchart, im viewchart, im runchart
- Miscellaneous:options

im checksourcetraces

displays invalid source traces for issues

Synopsis

`im checksourcetraces [--fields=field,field,...] [--fieldsDelim=value] [--query=[user:]query] [--queryDefinition=query] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--hostname=value] [--port=value] [--password=value] [--user=value] [--settingsUI=[gui|default]] [--status=[none|gui|default]] item id...`

Description

`im checksourcetraces` lists issues with traces to source files that have an invalid project configuration path or that do not trace to the current member revision.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field,field...`

  specifies the issue field(s) to check for source traces. Only source trace field types can be specified.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between the fields in the display.
- `--query=[user:]query`

  specifies the query used to populate the issue selection. Includes the name of the user who the query belongs to and the query name, for example, `mchang:ActiveDefects`. You do not have to specify the user name, but you must specify the query name. This option is useful when multiple users have the same name for a query.

Note

Integrity initially assumes that text before the first colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `mchang:FunctionalSafetyRequirements`, Integrity searches for the `Functional Safety Requirements` query created by `mchang`. If Integrity cannot find the query and/or user, it searches for the `mchang:FunctionalSafetyRequirements` query created by any user.

- `--queryDefinition=query`

  specifies a string to define the query constraints for the query used to populate the issue selection. The `<querydefinition>` must be of the same format as the query definition for a query. For details of the query definition format, see the reference page for the `--im createquery` command.
- `issue id`

  specifies the issue identification number of the issue you want to check source traces for. Overrides the `--query` option.

See Also

- Miscellaneous: options

im ci

checks in a new version for documents and content items

Synopsis

```
im ci [--[no|confirm]continueOnSingleValuedRelationship] [--[no]recurse]] [--major] [--minor] [--hostname= server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file |--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue id...
```

Description

`im ci` checks in a new version for documents and content items.

To track the evolution of your documents through the project lifecycle, you create unique document versions at various stages of the project. You create a new version by checking in your document or content item. Creating a document or content version provides you with an exact record of the document as it was at the time you performed the check-in operation.

Requirements are dynamic in that they can change over their lifetime. When communicating with internal or external stakeholders about a specific requirement, it is vital that discussions are based on the same requirement definition. Yet it is often difficult to identify the exact definition of the requirement being referenced. While the live document ID provides a unique identifier for the requirement, it applies to any point in the requirement's history. Consequently, there is a need for a simple way to refer to an exact point in the history of a requirement and its definition.

Document versions solve this problem by providing a clear reference to a specific point in the history of a requirement document. When viewing a versioned document or its contents, or when looking at trace relationships for decomposition or verification, you have clear visibility into the information as it was at the time of the check-in operation.

In this way, a document or content version represents a specific and meaningful point in the history of the artifact. For example, with the Requirement type, these are points that are meaningful to those working with the output of the requirements analyst. There is no benefit to versioning after each individual edit by an analyst, since all edits (including edits between versions) are always available in the history.

However, there is a need to identify key points, especially those when the analyst is ready to have their requirements reviewed or used by others. For example, during a review, when working to verify or validate the requirements, or when teams begin working to fulfill the requirements. At these key points, it is crucial that everyone references the exact same definition in a specified version.

For documents and content, the version identifier is numbered in the format *`Live Item ID-major.minor`*, where the *Live Item ID* is the number of the originating live document. The number to the left of the decimal point is the *major version*, and the number to the right of the decimal point is the *minor version*. Your Integrity administrator defines the pattern for version numbers by specifying values in workflow and document properties. For example, the default version number sequence is 0.1, 0.2, 0.3, ... 1.0, 1.1, 1.2. For information on the defined pattern for version numbers, contact your administrator.

You create a new version by checking in the document or content item. While the live document can continue to evolve through its lifecycle, a document version cannot be edited once you create it.

You cannot set an arbitrary version number. When an item is versioned, the system sets the next available version number according to the pattern defined by your administrator. When copying a versioned document, the version number on the existing document is not copied to the new document.

Documents can be versioned regardless of significant changes in the document item itself or in any of the containing items; however, if a content item is checked in and there are no significant changes, Integrity does not create a new version of the content.

For example, consider the following document versions:

- Document 100 is at version number 1.5 (document ID is 100-1.5). There have been significant changes to the document since the last version.
- Content item 200 is at version number 2.8 (content ID is 200-2.8). There have been no significant changes to the content item since the latest version.
- Document 300 is at version number 2.5 (document ID is 300-2.5). There have been no significant changes to the document since the last version.

Specifying:

```
im ci --minor 100
```

checks in a new version of Document 100 and updates the version number to 1.6 (the document ID of the new version is 100-1.6).

Specifying:

```
im ci --major 100
```

checks in a new version of Document 100 and updates the version number to 2.0 (the document ID of the new version is 100-2.0).

Specifying:

```
im ci --major 200
```

does not check in a new version of Content 200. Since there were no significant changes in the content item, Integrity does not create a new version.

Specifying:

```
im ci --major 300
```

checks in a new version of Document 300 and updates the version number to 3.0 (the document ID of the new version is 300-3.0).

Note the following:

- You must specify the type of check in you want to perform (either `--major` or `--minor`). If you do not specify either the`--major` or `--minor` option, an error occurs.
- If a content item is checked in and there are no significant changes, Integrity does not create a new version of the content.
- If a command does not support the use of a version ID, Integrity displays an error message when you enter the ID of a versioned document or content item. An error also occurs if you attempt to increment the revision of a versioned document or content item.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--major`

  increments the major version number. For example, if the current version is 100-1.5, incrementing the major version updates the version to 100-2.0.
- `--minor`

  increments the minor version number. For example, if the current version is 100-1.5, incrementing the minor version updates the version to 1.6.
- `--[no|confirm]continueOnSingleValuedRelationship`

  specifies whether confirmation is required to continue with a check-in operation when a single-valued relationship is found on the type. To be copied to the document version, the backward relationship field must allow multiple values.
- `--[no]recurse`

  indicates if child items of the selection should also be checked in. This option applies only to content items that do not reference documents. This option is ignored for documents and content items that reference documents.
- *`segment id...`*

  the ID of the document or content item you are creating the version for. To specify more than one document or content item, use a space separated list. If you specify a list that mixes documents and content items, the check-in operation is not performed. In addition, the check in operation is not performed if you specify a list that mixes two different types of content (content items that reference documents and content items that do not reference documents).

See Also

- Commands: im incrementrevision
- Miscellaneous: ACL, diagnostics, options, preferences

im columnsets

provides a tabular list view of column sets based on the selection

Synopsis

```
im columnsets[--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value ] [--hostname=value ] [--port=value ] [--password=value ] [--user=value ] [(-?|--usage)] [(-g|--gui)][--height=value ][--width=value ] [-x value ] [-y value ] [(-F value |--selectionFile= value )] [--quiet] [--settingsUI=[gui|default]] [--status=[ none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value ] [--forceConfirm=[yes|no]] columnset
```

Description

Note

Column sets are no longer supported for the Integrity Client. This command can only be used to view column sets for the MKS Worktray used in integrations. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

Column sets are a grouping of Issue fields into columns for viewing. The issue field names are used as the column headings and are referred to as "column types." The order of the column headings and the rows may be sorted by field name. Column sets are individually saved for each user. A user may not view, modify, or delete another user's column sets.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=` *`field1[:width1],field2[:width2]...`*

  specifies the columnset fields, and their respective widths, to be included in the tabular list view. Field*n* can be `fields`, `name`, `sortDirection`, or `sortField`. Use commas to specify more than one field. For output directed to the CLI, the field width is specified in characters. For output to the GUI, the field width is specified in pixels.

- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between the fields in the tabular display.

- *`columnset`*

  specifies the column set you want to display. If you do not specify a column set, all column sets display.

See Also

- Commands: im copycolumnset, im createcolumnset, im deletecolumnset, im editcolumnset, im viewcolumnset
- Miscellaneous: options

im connect

establishes a connection to an Integrity Server

Synopsis

```
im connect [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im connect` establishes a connection to an Integrity Server host. Most commands implicitly connect to the host; this does so explicitly. In fact, all the other commands call `im connect` to establish the connection. You can use im disconnect to disconnect from an Integrity Server host.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=value`

  identifies the name of the host server where the Integrity Server is located.
- `--port=value`

  identifies the port on the host server where the Integrity Server is located.
- `--password=value`

  identifies the password to use for connecting to the Integrity Server.
- `--user=value`

  identifies the user to use for connecting to the Integrity Server. This typically defaults to the name you have used to log into your client machine.

See Also

- Commands: im disconnect, im exit, im servers
- Miscellaneous: options

im copychart

copies the common fields of an existing Integrity chart to a new chart

Synopsis

```
im copychart [--bgColor=value] [--chartFootnote=value] [--chartTitle=value] [--dataColors=value] [--descriptionFont=value] [--
[no]displayDescription] [--[no]displayLegend] [--[no]displayLabels] [--endDate=value] [--fieldFilter=field=[value,value,...] [--
fieldValues=value] [--footnoteFont=value] [--graphStyle=
[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]] [--groupingValues=value] [--[no]is3D]
[--[no]isAutoColors] [--[no]isShowZeroFieldCount] [--[no]isShowZeroGroupingCount] [--legendBgColor=value] [--legendPosition=
[Right|Bottom|Left|Top]] [--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]] [--legendTitle=value] [--
outlineColor=value] [--projectedTrendExpressions=value] [--query=[user:]query] [--startDate=value] [--numberOfSteps=value] [--
titleFont=value] [--trendStep=[Hour|Day|Week|Month|Quarter|Year]] [--[no]useIssueDefinedOrigin] [--[no]xReverse] [--[no]xShowGrid] [-
-[no]xShowTitle] [--yLabelRotation=[Horizontal|VerticalUp]] [--[no]yReverse] [--[no]yShowGrid] [--[no]yShowTitle] [--
description=value] [--name=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--
sharedAdmin] [--computations=value] [--startDateField=field] [--runDateIsEndDate] [--[no]deltasOnly] [--issueIdentifier=value] [--
[no]displayShapesForLineGraphs] [--[no]swapRowsAndColumns] [--[no]displayRowTotals] [--[no]displayColumnTotals] [--
rangeDefinitions=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--
selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [--forceConfirm=[yes|no]] [user:]chart
```

Description

`im copychart` copies the common fields of an existing Integrity chart to a new chart. Integrity displays a chart selection dialog box when you use the `-g` or `--gui` option.

For example,

```
im copychart --name="Release 3.0 Docs Issues By User" --query="Release 3.0 Docs Issues" "Release 2.0 Docs Issues"
```

copies the existing Release 2.0 Docs Issues by User chart to create a new chart that shows the same type of information for Release 3.0.

For more information on charts, refer to the *PTC Integrity User Guide*.

Note the following:

- A chart can be edited by the user who created it. Principals (users and groups) that a chart is shared to can edit it if they have edit permissions assigned to them by the chart creator. A chart can only be deleted by the user who created it or by an administrator.
- The minimum information required to create a distribution chart is a chart name, a field, and a query. The minimum information required to create a trend chart is a chart name, step type, start and end date, and a field. The minimum information required to create an issue fields chart is a chart name, query, and aggregate expression. The minimum information required to create an issue fields trend chart is a chart name, query, step type, start and end date, and numeric field. All other modifications and additional information are optional.
- Charts can do more than just display field information in a graphical format. You can also perform arithmetic calculations between numeric fields, displaying the values in the chart. For example, you can calculate the average for a group of field values or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- All charts are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or see your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--chartFootnote=value`

  specifies the footnote text of the chart.
- `--chartTitle=value`

  specifies the title of the chart.
- `--titleFont=value`

  specifies the font to be used for the chart title. Use the following format: `name,style,size`; where `style` is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic. For example, helvetica,1,10. When the chart is run, if the specified font cannot be found, Integrity uses a substitute font.
- `--descriptionFont=value`

  specifies the font to be used for the description. Use the following format: *name,style,size* format; where *style* is 0 for plain, 1 for bold and 2 for italic. For example, helvetica,1,10.
- `--[no]displayDescription`

  specifies whether to display the chart description.
- `--trendStep=[Hour|Day|Week|Month|Quarter|Year]`

  specifies the interval for each point on a trend or issue fields trend chart graph.
- `--[no]useIssueDefinedOrigin`

  specifies whether to use the start date defined in an issue field for an issue fields trend chart.
- `--startDate=value`

  specifies the start date for trend or issue fields trend charts. To specify a date and time, type *MM/dd/yyyy h:mm:ss [AM|PM]*.

  Other acceptable date formats include:
  MM/dd/yyyy h:mm:ss a zMM/dd/yyyy h:mm:ss.SSS a zMM/dd/yyyy h:mm:ss aMM/dd/yyyy h:mm:ss.SSS aMM/dd/yyyy
- `--endDate=value`

  specifies the end date for trend or issue fields trend charts. To specify a date and time, type *MM/dd/yyyy h:mm:ss [AM|PM]*. See the `--startDate=value` option for additional date and time formats.
- `--description=value`

  specifies a short description for the chart.
- `--name=value`

  specifies the new name of the chart. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`

  specifies the users and groups that can use and modify the chart. Your administrator defines users and groups.
- `--fieldFilter=field = [value,value,...]]`

  specifies how field filters can be applied to the chart when it is run. The first component of the value is the field name. Currently, only project field filters are supported. The second component specifies the project(s) that you want to filter the chart data by when it is run. For example, `--fieldFilter="Project=/Project1"` filters for issues that have a value of `Project1` in the `Project` field. If you do not specify a value, Integrity filters for issues with a value of `Unspecified` in the `Project` field.

  Note

  You can also define project filters for dashboards. Depending on how you design your dashboard, when a chart is run through a dashboard, the dashboard's project filter can override the chart's project filter.

- `--fieldValues=value`

specifies the field, field values and aliases used by the chart. For example: `--fieldValues=Type=Documentation, Development[Feature Request, Bug]` would include issues that have a `Type` field with a value of `Documentation`, `Feature` or `Bug`, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

Use * to include all field values, and + to automatically include all future field values. For example: --fieldValues=Type=*, +, Development[Feature Request, Bug] would include all current values and any future values for the `Type` field, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

For more information on specifying chart values, see the *PTC Integrity User Guide*.

- `--footnoteFont=`*`value`*

specifies the font to use for the footnote. Use the following format: *`name,style,size`* format, where *`style`* is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic, for example, helvetica,1,10.

- `--groupingValues=`*`value`*

specifies the field, field values, and aliases to use to group the data in the chart. For example:

`--groupingValues=State=Submit, In Work[In Progress, In Development]` would group chart data into separate components for `Submit` and `In Work`, with `In Work` being a combination of the `In Progress` and `In Development` states.

Use * to include all field values, and + to automatically include all future field values. For example: `--groupingValues=State=*, +, In Work[In Progress, In Development]` would group chart data into separate components for all current values and any future values for the State field, with In Work being a combination of the In Progress and In Development states.

For more information on specifying chart values, see the *PTC Integrity User Guide*.

- `--projectedTrendExpressions=`*`value`*

When creating or editing an item fields trend chart, you can specify whether to display a projected trend for one of the displayed item field series, thus supporting the use of *`burn-down charts`* for planning work remaining within a fixed duration project. The projected trend displays a line between a starting value corresponding to the starting date and the end value corresponding to the end date. In addition, if you choose to display the projected trend, you can also display the actual trend as a line between the last actual series value to the end value corresponding to the end date.

This option specifies attributes for a projected trend graph in the item fields trend chart, where *value* consists of the following attributes separated by a colon:

`chartedExpression` is the expression (numeric field) to chart. This value is mandatory.

`startValueExpression` is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the historic item field value. This value is mandatory.

`endValueExpression` is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the "now" item field value. This value is mandatory.

`projectedTrendLabel` is a text string used for the chart legend and tooltips.

`showUpdatedTrend` displays the projected trend in the chart. Specify `true` or `false` (default).

`updatedTrendLabel` is a text string used for the chart legend and tooltips.

Tip: To clear the value for `--projectedTrendExpressions=` *`value`*, specify an empty string.

For example:

`--projectedTrendExpressions=Effort:0:"Effort":"Planned Effort":true:"Unplanned Effort"`

- `--query=`*`[user:]query`*

specifies the name of the query that the chart is based on.

Note

If the chart is a shared admin object, an admin query is required.

- `--graphStyle=[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]`

specifies the graph style used of the chart.

- `--dataColors=`*`value`*

specifies the custom data colors to be used using the RGB color model. For example:

'R,G,B;R,G;R,G,B'

where R,G and B are within the range 0-255.

If the chart has more data points than the data colors you specify, the colors are repeated. If the

`--[no]isAutoColors`

option is true, the colors specified here are ignored.

Note

This option is invalid for table style graphs.

- `--bgColor=`*`value`*

specifies the background color of the chart using the RGB color model. For example:

'R,G,B'

where R,G and B are within the range 0-255.

Note

This option is invalid for table style graphs.

- `--[no]displayLegend`

specifies whether to display the chart legend.

Note

This option is invalid for table style graphs.

- `--[no]displayLabels`

specifies whether to display labels for values in the chart. If you select a pie graph style, this option is automatically selected. `--nodisplayLabels` is the default option.

Note

This option is invalid for table graphs.

- `--[no]is3D`
- specifies whether to display bar and pie graphs in 3D.

Note
This option is invalid for table style graphs.

- `--[no]isAutoColors`
specifies whether to use the default chart colors. If false, you must provide colors through the data colors option.

Note
This option is invalid for table style graphs.

- `--[no]isShowZeroFieldCount`
specifies whether to include empty field values in the chart.
- `--[no]isShowZeroGroupingCount`
specifies whether to include empty grouping values in the chart.
- `--legendBgColor=`*`value`*
specifies the background color for the chart legend using the RGB color model. For example:
'R,G,B'
where R,G and B are within the range 0-255

Note
This option is invalid for table style graphs.

- `--legendPosition=[Right|Bottom|Left|Top]`
specifies the legend position in relation to the graph.

Note
This option is invalid for table style graphs.

- `--legendTitle=`*`value`*
specifies the title for the chart legend.

Note
This option is invalid for table style graphs.

- `--outlineColor=`*`value`*
specifies the outline color of the graph using the RGB color model. For example:
'R,G,B'

Note
This option is invalid for table style graphs.

- `--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]`
specifies the rotation of the horizontal axis labels for the chart.

Note
This option is invalid for table style graphs.

- `--[no]xReverse`
specifies whether the chart uses a horizontal axis with a reverse orientation (left).

Note
This option is invalid for table style graphs.

- `--[no]xShowGrid`
specifies whether to display horizontal grid lines.

Note
This option is invalid for table style graphs.

- `--[no]xShowTitle`
specifies whether to display the title for the horizontal axis.

Note
This option is invalid for table style graphs.

- `--yLabelRotation=[Horizontal|VerticalUp]`
specifies the rotation of the vertical axis labels for the chart.

Note
This option is invalid for table style graphs.

- `--[no]yReverse`
specifies whether the chart uses a vertical axis with a reverse orientation (down).

Note

This option is invalid for table style graphs.

- `--[no]yShowGrid`

  specifies whether to display vertical grid lines.

Note

This option is invalid for table style graphs.

- `--[no]yShowTitle`

  specifies whether to display the title for the vertical axis.

Note

This option is invalid for table style graphs.

- `--sharedAdmin`

  specifies the chart as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.

- `--computations=`*`expression:name:pattern:axis name:minRangeValue:maxRangeValue:tickUnitValue`*

  specifies an expression and numeric axes attributes.

- Note the following about specifying numeric axes attributes:
  - If you specify one set of numeric axes attributes (minimum range, maximum range, and tick unit), these attributes are specified for the X and Y axes. For XY (scatter) charts, PTC recommends against setting individual numeric axes attributes for the X and Y axes.
  - For bubble charts, PTC recommends against specifying numeric axes attributes because they override the calculated values provided by the underlying expression and users will have to zoom in/out to properly view chart values.

  expression specifies an aggregate expression for a distribution chart, a computed expression for an issue fields chart, or a numeric field for an issue fields trend chart. For information on creating expressions, see the *PTC Integrity Server Administration Guide*.

  name specifies the label name for the aggregate expression, computed expression, or numeric field as you want it to appear in the chart. If you do not define a label, the aggregate expression, computed expression, or numeric field name displays.

  pattern specifies the display pattern for the value of the aggregate expression, computed expression, or numeric field value.

  axis name specifies a name for the numeric axis as you want it to appear in the chart.

  minRangeValue specifies the minimum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

  maxRangeValue specifies specifies the maximum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

  tickUnitValue specifies the units that display on the numeric axis. For example, if you specify a minimum range of 0, a maximum range of 100, and a tick unit of 10, the numeric axis displays 0, 10, 20, 30, 40, and so on up to 100.

Note

Field names in expressions and expression labels with colons must be enclosed by escaped double quotes, and the whole computation must be enclosed in double quotes, for example,

```
--computations="\"Actual Dev Time\":\"Hours: Development\":pattern:..."
```

- `--startDateField=`*`field`*

  specifies the date field containing the date you want to use as the start date for each issue in an issue fields trend chart.

- `--numberOfSteps=`*`value`*

  specifies the trend chart's time span. If this option is specified, the chart's end date is determined by the specified step type multiplied by the specified number of steps.

Note

You cannot have more than 500 steps in a trend chart.

- `--runDateIsEndDate`

  specifies that the chart's run date is the end date. This option replaces the `--endDate`=value option.

- `--[no]deltasOnly`

  specifies whether to display only the differences between the current and previous values of the reported numeric fields in an issue fields trend chart.

- `--issueIdentifier=`*`value`* specifies the field that you want to identify issues by in an issue field or issue fields trend chart. For example, if you specify `--issueIdentifier={Project}`, each issue in the chart is identified by the value of the `Project` field.

  If you want to add text that precedes the specified field, type it before the field, for example, `--issueIdentifier=Project:{Summary}`. The chart then identifies each issue by displaying

  Project: *Summary field value*

- `--[no]displayShapesForLineGraphs`

  specifies whether to display shapes in a line graph chart. The shapes in the chart represent data, allowing you to more easily differentiate the data in the chart.

- `--[no]swapRowsAndColumns`

  specifies whether to invert the appearance of columns and rows in a table chart.

- `--[no]displayRowTotals`

  specifies whether to display row totals in a table chart.

- `--[no]displayColumnTotals`

  specifies whether to display column totals in a table chart.

- `--rangeDefinitions=`*`value`*

  specifies range definitions for computed expressions included in a table chart, where *value* consists of the following attributes: *expression name;range field name;range label:lower limit:upper limit:icon:background color:text color:text style:display format; lower limit:upper limit:.....;extend to axis* .

  `expression name` specifies the name of the computed expression that the range definition applies to. An expression name is mandatory and must be a valid expression in the chart. For column or row totals, valid expression names are `-Column Totals-` and `-Row Totals-`. For distribution charts containing multiple computed expressions, row or column totals must be followed by the expression name.

  `range field name` specifies a valid field name if you want to relate the range definitions to an existing range field. For one chart range, specify an empty string as the range field name. If a valid range field name is defined for each range, define a range label, background color, text color text style, and display format. For individual range definitions, define a range label, lower limit, upper limit, icon, background color, text color text style and display format for each range.

  `range label` specifies a label for the range.

lower limit specifies the lower limit of the range. If a lower limit is not specified, `-Infinity` is automatically specified.

upper limit specifies the upper limit of the range. If an upper limit is not specified, `Infinity` is automatically specified.

Note

A numeric value must be specified for a defined range; range intersections are invalid. For example, the following ranges are invalid: 0 - 5 and 4 - 8, or 0 - 5 and 5 - 10 . For an integer field, an acceptable range would be 0 - 5 and 6 - 10. For a floating point field, an acceptable range would be 0 - 5 and 5.01 - 10.

---

icon specifies an image file representing the range category. This is optional.

background color specifies the background color of the range using the RGB color model, for example, 'R,G,B', where R, G, and B are within the range 0-255.

text color specifies the text color of the range using the RGB color model, for example, 'R,G,B', where R, G, and B are within the range 0-255.

text style specifies the text style. Available text styles are plain, bold, italic, bolditalic, or defaultplain.

display format specifies how to display the range in the table chart. Available options are value, iconvalue, icon, label, iconlabel, or blank.

extendToAxis specifies whether to apply the range definition associated with a computed expression to all computed expressions in the chart. This option can be false or true. By default, false is specified.

Note the following:

- You cannot specify a range for the Count expression.
- You can specify a range for each computed expression; however, only one computed expression can specify the *extendToAxis* option.
- If a table cell contains a display definition that conflicts with the *extendToAxis* option of another table cell, both table cells display the *background color* option of the table cell with the enabled *extendToAxis* option.

- *[user:]chart*specifies the name of the chart to copy, and the user who created that chart. This is useful when multiple users have the same name for a chart.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a chart name. If Integrity fails to find a matching user name and chart name, it searches for a chart name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` chart created by `jhoyt`. If Integrity cannot find the chart and/or user, it searches for the `jhoyt:CosmosDefects` chart created by any user.

See Also

- Commands: im editchart, im deletechart, im createchart, im viewchart, im runchart
- Miscellaneous: options

im copycolumnset

copies the properties of an existing columnset and allows you to rename it as your own

Synopsis

```
im copycolumnset [--fields=field,field,...] [--name=value] [--[no]sortAscending] [--sortField=field] [--hostname=value] [--
port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=
[gui|default]] [--status=[none|gui|default]][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] columnset
```

Description

Note: Column sets are no longer supported for the Integrity Client. This command can only be used to copy column sets for the MKS Worktray used in integrations. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

`im copycolumnset` copies the properties of an existing column set and allows you to rename it as your own.

Important:

You cannot use the name of an existing column set. If you do not specify a name for the new column set, `Copy of` is prefixed to the original column set name, for example, `Copy of Cosmos Defects View`.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fields=`*`field,field,...`*

  specifies the issue fields to be included in the column set, for example, `ID`, `Type`, `Summary`, `State`, `Project`. Your administrator defines the fields in an issue type. Use commas to specify more than one field.
- `--name=`*`value`*

  specifies the name of the column set to create.
- `--[no]sortAscending`

  specifies whether to sort the specified field in ascending or descending order.
- `--sortField=`*`field`*

  specifies the field to sort issues by.
- *`columnset`*

  specifies the column set you want to copy.

See Also

- Commands: im createcolumnset, im editcolumnset, im viewcolumnset, im deletecolumnset, im columnsets
- Miscellaneous: options

im copycontent

copies nodes(s) to a new segment

Synopsis

```
im copycontent [--asOf=<date>:label:<label>] [--parentID=value] [--insertLocation[=number|first|last|before:name|after:name] [--
[no]recurse] [--[no]confirmLarge] --threshold=value] [--[no]traces] [--refmode=[reuse|share] [--[no]branch] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] node id...
```

Description

`im copycontent` copies nodes to a new segment. You can insert the copied content at a specified location in the segment structure. For example:

```
im copycontent --parentID=23 --insertLocation=3 15 242 590
```

copies issues 15, 242 and 590 and inserts them at locations 3, 4 and 5 in the structural relationship list for segment 23.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--asOf=<date>:label:<label>`

  allows you to copy content as of a specific date or label. For example, to copy content based on a specific date, type:

  ```
  im copycontent --asOf="January 8, 2012 10:00:00 AM EST" 123
  ```

- `--[no]recurse`

  indicates whether to copy all nodes and/or segments beneath the specify.

- `--[no]confirmlarge`

  species to display a message to users confirming that they are copying a large number of content issues. The default is 50 and can be modified using the `--threshold` option.

- `--[no]traces`

  indicates whether to copy all existing trace relationships to the new (copied) nodes. The default is `--notraces`.Use im propegatetraces to copy traces to the branched document.

- `--[no]branch`

  indicates that the copy operation you are performing will be a branch. Branching creates a copy of the segment and adds it to the project history. You can branch a segment as of a specific date or label.

- `--threshold=value`

  specifies the number of content issues to allow users to copy. The default threshold is 50. Users will be prompted to confirm a copy operation of more than 50 issues. If you increase the default threshold, you can specify `--[no]confirmlarge` option to either notify users or bypass notification.

- `--parentID=value`

  the ID of the parent segment or node that will contain the reference to the node being copied. This option is required.

- `--refmode=[reuse|share]`

  specifies the reference mode you want to apply to the copied content. See the *PTC Integrity User Guide* for details about reference modes and standard copy and paste operations and/or copy and paste for reuse.

- `--insertLocation[=number|first|last|before:name|after:name]`

  determines where the copied content should go in the parent segment's structural relationship list. The options are as follows:

  `first` inserts the content at the beginning of the list.

  `last` inserts the content at the end of the list.

  `before`:*name* inserts the content before the specified ID.

  `after`:*name* inserts the content after the specified ID.

  [0,...] inserts the content at the specified location. If a negative is specified, the content is inserted at the beginning of the list. If the number specified is too large, the content is inserted at the end of the list.

- *node id...*

  the ID of the node you want to copy. Use a space separated list to specify more than one node ID, for example, `240 241 242`.

Note

To restrict access to specific information in documents, all documents are subject to visibility rules set by your administrator. You must have visibility to the nodes you want to copy. For more information, contact your administrator.

See Also

- Commands: im createcontent, im createsegment im removecontent, im movecontent, im importcontent
- Miscellaneous: diagnostics, options, preferences

im copydashboard

copies an Integrity dashboard

Synopsis

```
im copydashboard [--shareWith=u=user1[:modify],user2[:modify],...; g=group1[:modify],group2[:modify],...] [--description=value] [--
name=value ] [--fieldFilterConstraint=field:[Open[:value,value,...] | [Fixed[:value,value,...] | [Restricted[:value,value,...]
[:value,value,...]] [--layout=value] [--layoutFile=file] [--sharedAdmin] [--hostname=value] [--port=value] [--password=value] [--
user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=
[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [user:]dashboard
```

Description

`im copydashboard` copies the properties of an Integrity dashboard to a new dashboard. For more information on dashboards, refer to the *PTC Integrity User Guide*.

For example,

```
im copydashboard --name="Release 2.0 Defect Trends" --fieldFilterConstraint=project:Fixed:/"Release 2.0" jriley:"Release 1.0 Defect Trends"
```

copies the Release 1.0 Defect Trends dashboard created by `jriley` to a new dashboard called `Release 2.0 Defect Trends`, with a fixed filter for the release 2.0 project.

Options

This command takes the universal options available to all`im` commands, as well as some general options. See the options reference page for descriptions.

- `--shareWith=u=`*`user1[:modify],user2[:modify],...; g=group1[:modify],group2[:modify],...`*

  specifies the users and groups that can use and modify the dashboard. Your administrator defines users and groups.
- `--name=`*`value`*

  specifies the name of the dashboard. Names may be a maximum of 100 characters and cannot contain square brackets.

Note

If you do not specify a different name for the new dashboard, Integrity adds `Copy of` as a prefix to the query name, for example, `Copy of Project Overview`.

- `--description=`*`value`*

  specifies a short description for the dashboard.
- `--fieldFilterConstraint=`*`field`*`:[Open[:`*`value,value,...`*`]|[Fixed[:`*`value,value,...`*`]|[Restricted[:`*`value,value,...`*`][:`*`value,value,...`*

  specifies how field filters can be applied to the dashboard at runtime. The first component of the value is the field name. Currently, only project field filters are supported. The second component is the filter type.

  `Open` specifies that all projects can be selected as filter values when the dashboard is run. You can also specify default filter values to apply.

  `Fixed` specifies that when the dashboard is run it will be filtered by the specified values. You cannot change this filter at runtime.

  `Restricted` specifies that when the dashboard is run you can select any of the specified filter values. You can also specify default filter values to apply.

Note

Depending on how you design your dashboard layout, the dashboard filter may not be applied to chart, report, report link or query link dashboard components. If this option is not specified, the `Open` filter is used.

- `--layoutFile=`*`value`*

  specifies the file that contains the complete definition of the dashboard layout.
- `--layout=`*`value`*

  the XML representation of the dashboard layout. The layout must conform to a specified format. For more information, see the *PTC Integrity User Guide*. This setting is optional.
- `--sharedAdmin`

  specifies the dashboard as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:
  Once a user object is converted to a system provided object, you cannot revert it to a user object again.

Note

If the dashboard you are copying is a system provided dashboard, the `--sharedAdmin` option is not set in the copy.

- *`[user:]dashboard`*

  specifies the name of the dashboard to copy, and the user who created that dashboard. This is useful when multiple users have the same name for a dashboard.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the `ProjectOverview` dashboard created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the `jhoyt:ProjectOverview` dashboard created by any user.

See Also

- Commands: im editdashboard, im deletedashboard, im createdashboard, im viewdashboard, im dashboards, im rundashboard
- Miscellaneous: options

im copyissue

copies the common fields of an existing Integrity issue to a new issue

Synopsis

```
im copyissue [--asOf=[date|label:label] [--[no]showWorkflow] [--[no]copyFields] [--copyFieldsList=field],field]] [--[no]link] [--linkToField=field]] [--addAttachment=value] [--addFieldValues=value] [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--type=type] [--customFieldDefinition=value] [--customFieldValue=value] [--field=value] [--richContentField=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] issue
```

Description

`im copyissue` copies the common fields of an existing Integrity issue to a new issue. This includes any attachments referenced in rich text fields. You can complete fields in the new issue, and add relationships and attachments. You can also use the `im copyissue --link` or the `im copyissue --linkToField` command to create a relationship between the source issue and the copied issue. To select an issue, use the `-g` or `--gui` option and Integrity displays an issue selection dialog box.

For example,

```
im copyissue --link --type=Docs --asOf="January 8, 2007 10:00:00 AM EST" 123
```

creates a new Docs issue from issue 123, copying the content of the common fields as of January 8, 2008, and linking the new issue to the original issue through the Forward Relationship field.

Note the following:

- Your administrator can configure long text fields to support rich content. *`Rich content`* enhances the display of text in long text fields by adding formatted text, tables, background colors, images, and hyperlinks. From the CLI, rich content is applied using a limited set of HTML elements and attributes. For a complete list of supported elements and attributes, see the *PTC Integrity User Guide*.
- Your administrator defines which issue types and custom fields you are allowed to edit. If your administrator defines a field as a logging text field, you may only enter new text and not edit existing text.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields vary based on the time zone that a user is operating in.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- The list of users in the Assigned User field is limited to those with permissions to the issue's project. The same applies to the Assigned Group field.
- Depending on your workflow, you may not be able to edit an issue that is in an end state.
- Your administrator may include the time in date fields. You can specify the time when you select a date from the calendar. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM. If you do not specify a time, the current time displays in the date field.
- To retrieve metrics from a configuration management project related to the issue you are copying, your administrator may define a field that accepts a configuration management project as a value. Optionally, you can specify a checkpoint revision or development path. If you specify a configuration management project and a checkpoint, then save the issue, one or both of the following may occur when you view the issue in the GUI or Web interface:
  - One or more computed expressions in the issue calculate specific metrics about the project, displaying the results as a read-only value in a computed field (the visibility of the computed field depends on the field's relevance rules). For example, once you specify a project for the `Source Code` field, a `Lines of Code` field could calculate and display the number of lines of code in that project. As lines of code are added or removed from the project, the `Lines of Code`field updates to display the new value.
  - A metrics hyperlink displays in the configuration management project field. Clicking the hyperlink displays various configuration management metrics about the project.

  In addition, the server and project information display in the configuration management project field as a hyperlink. Clicking on the hyperlink displays the project in a Project view.

  To select a configuration management project, you require the `OpenProject` permission for the specified project. Once a configuration management project has been specified, metrics can be obtained by any user with permissions to view the configuration management project field. For more information on selecting configuration management projects and viewing configuration management metrics, refer to the *PTC Integrity User Guide*. For more information on creating configuration management metrics, refer to the *PTC Integrity Server Administration Guide*.

  Important:

  Metrics are only maintained against project checkpoints; therefore, to generate metrics, you must specify a checkpoint when you specify the configuration management project.
- You cannot set a date field to null if the date has been previously set.
- Your administrator may include the time in date fields. You can specify the time when you select a date from the calendar. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. If you do not specify a time, the current time displays in the date field.
- You can modify a value from a mandatory field and save it.
- If your administrator has set up electronic signatures, you may need to provide your user name and password when making specific edits to an issue. For example, you could be required to provide an electronic signature when you change an issue's state to `Completed`.
- By default, Integrity allows file attachments to a maximum size of 4 MB and a maximum of 255 characters for file names. Your administrator may define a higher or lower limit depending on the requirements of your system. You can also attach more than one file to a single issue.
- Integer fields allow a maximum of nine digits and floating point fields allow a maximum of 15 digits. Your administrator can define default, minimum, and maximum values.
- If you attempt to save changes to an issue after another user saves changes to the same issue, the following error message may appear:

  `Could not save modified issue`:The issue was changed by another user after you began your edit.

  Typing `Cancel` discards your changes. Typing `OK` displays your unsaved changes to the issue. PTC recommends copying your changes, canceling the issue, then re-editing the issue and adding your changes.
- Copying an issue containing user, group or project fields with inactive values results in an error message.
- Inactive pick list values cannot be specified for pick list fields; however, pick list fields retain inactive pick list values. If you edit a multi-valued pick list field, inactive pick list values can no longer be specified, even if only one of the values was previously inactive.
- If an issue contains multi-valued user or group fields with inactive values, editing the inactive values and saving the issue prompts you to leave the fields unchanged or clear the inactive values.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

-
- `--asOf=[`*`date`*`|label:`*`label`*`]`

  allows you to copy an issue as of an historical date or label. This option is used with the `--issues` option. If a value is not provided the issue is copied as of the server's current time. This field is optional.
- `--[no]showWorkflow`

  specifies whether to display the workflow for the issue type, if your administrator has enabled it. This option can only be specified with `-g` or `--gui`. Viewing the Workflow panel is useful for determining where you can progress in the workflow. The Workflow panel displays the complete workflow for the issue type, unvisited states, visited states, the current state, other state transitions, and phases, as indicated by the Legend.
- `--addSourceTrace=`*`value`*

  adds a trace to a source file, where `value` is of the form "*`field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName]`*

[,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber] ".

Note

- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addSourceLink=`*value*

  adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]* ".

Note

- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--[no]copyFields`

  specifies whether to copy the common fields of the specified issue to the new issue. Common fields refer to fields that are visible for both the original and new issue types and editable for the new issue type. The default is to copy common fields.
- `--copyFieldsList`

  specifies which common fields to copy to the new issue. Only fields that are visible for both the original and new issue types and editable for the new issue type can be copied. The default is to copy the common fields that were specified by your administrator.
- `--[no]link`

  specifies whether to link the new issue to the Forward Relationships field on the source issue. The ID of the source issue appears in the Backward Relationships field on the new issue.

Note

Your administrator determines which issue types can be linked to other issues.

- `--linkToField=`*field*

  specifies the relationship field on the source issue to use to link to the newly created issue. The ID of the new issue appears in the specified field.

Note

Your administrator defines relationship fields and determines which issue types can be linked to other issues.

- `--addAttachment=`*value*

  adds attachments, where *value* is of the form "*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]* " .

Note

The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

For example,

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
```

adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. An issue cannot have more than one attachment with the same name.

Note

Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

- `--addFieldValues=`*value*

  specifies new field values to add, where *value* is of the form "*fieldName*=*fieldValue*."

Note

This option is only valid for the following multi-valued field types: pick, user, group, item backed picklist (IBPL), and relationship.

- `--addRelationships=`*value*

  specifies a relationship field, the ID of the related issue, and any relationship flag for the related issue, where *value* is of the form [*fieldName*:]*id*[*relationshipFlags*][,...]. Use commas to specify more than one issue ID, for example, *23*, *242* If no *fieldName* is specified, the *Forward* `Relationships` field is used.

Note

This option is deprecated. Instead, use the `--addFieldValues=`*value* option.

- `--type=`*type*

  specifies the issue type to create. Your administrator defines issue types. This option is mandatory.
- `--field=`*value*

  specifies a field and its value for the new issue, where *value* is of the form "*fieldName*=*fieldValue*", for example, `--field=Severity=Critical`. If the field is multi-valued, *value* is of the form "*fieldName*=*fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add to the issue.
- `--richContentField=`*value*

  specifies a rich content field and its value for the new issue, where *value* is of the form "*richcontentfieldname*=*fieldValue*".

  To specify a link to an item by ID: `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

For example:

```
<a href="mks:///item?itemid=4547">Part Requirement</a>
```

To specify a link to an item by ID and label:

```
<a href="mks:///item?itemid=ID&label=LABEL">DisplayText</a>
```

For example:

```
<a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>
```

To specify a link to an item by ID and revision:

```
<a href="mks:///item?itemid=ID&revision=REVISION"> DisplayText</a>
```

For example:

```
<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>
```

To specify a link to an item by ID and date:

```
<a href="mks:///item?itemid=ID&asof=DATETIME">DisplayText</a>
```

where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

For example:

```
<a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>
```

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "`<b>Feature Overview</b>`". In the Windows command line interface (cmd.exe), the ^escape character must precede the <and > characters in HTML tags, for example, `^<b^>Feature Overview^/b^>` . You may need to escape nested " characters, for example, `--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"`.

- *issue*

  specifies the ID of the issue you want copy.

See Also

- Commands :im createissue, im editissue, im extractattachments, im viewissue
- Miscellaneous: options

im copyquery

copies the properties and constraints of a query and allows you to rename it as your own

Synopsis

```
im copyquery[--image=[none|default|<path>] [--copyColumnsFromQuery[=[user:]query] [--fields=field,field,...] [--[no]sortAscending][--
sortField= field] [--image=[none|default|path][--description=value][--shareWith=u=user1[:modify],user2[:modify],..
;g=group1[:modify],group2[:modify],..] [--name=value] [--hostname=value] [--queryDefinitionFile=value][--queryDefinition=query][--
[no]referenceOriginalQuery][--sharedAdmin] [--port=value] [--password=value] [--user=value][(-?|--usage)][(-g|--gui)][(-F value|--
selectionFile=value)][--quiet][--settingsUI=[gui|default]][--status=[none|gui|default]] [(-N|--no)][(-Y|--yes)][--[no]batch][--cwd=value
][--forceConfirm=[yes|no]][user:]query
```

Description

`im copyquery` copies the properties and constraints of a query and allows you to rename it as your own.

For example,

```
im copyquery --name="Release 2.0 Docs Issues Hours" --fields="State","Estimated Hours","Actual Hours","Assigned User" "Release 2.0 Docs Issues"
```

copies the existing Release 2.0 Docs Issues query to create a new query that shows the estimated and actual hours for the same issues.

Note the following about using the `im copyquery` command:

- Copying a Quick Query creates a named query that can be shared with others.
- If the query you are copying is a system provided object, the `--sharedAdmin` option is not set in the copy.
- Unique names must be used for all queries created by you.
- If you do not specify a different name for the new query, Integrity adds `Copy of` as a prefix to the query name, for example, `Copy of Cosmos Critical Defects`.
- You cannot query on configuration management project fields.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.
- When specifying a date range in a query, all dates are treated as timestamps, converted to the time zone on the Integrity Server, and then truncated to a date-only format. The resulting date-only format is then converted to a Structured Query Language (SQL) statement format on the Integrity Server, and the query is run based on the time zone of the server. If the user defining the date range is not in the same time zone as the Integrity Server, a day can be lost or gained at either end of the defined range. The rollback of dates can cause the query results to vary.

  Important:

  The date range conversion does not cause any problems when the user is in the same time zone as the Integrity Server.

  For example, if the Integrity Server is in the America/New_York time zone and the following query date range is defined by a user in Germany: Jan 1, 2006 to Jan 31, 2006.

  The final conversion of the date range is: Dec 31, 2005 06:00:00 AM GMT+01:00 to Jan 31, 2006 06:00:00 AM GMT+01:00.

  To avoid any date rollbacks when working with query date ranges in Integrity, PTC recommends that users specify the time zone that is used by the Integrity Server they are connecting to.
- Symbolic dates are evaluated on the Integrity Client's time zone.
- Because dynamically computed fields are not stored in the database, dynamically computed short text fields cannot be located with an all text field search in the Integrity Client. To search for dynamically computed short text fields, create a query that includes a specific \u201cfield contains\u201d comparison. For more information about computed fields, see your administrator.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--copyColumnsFromQuery[=[user:]query]` specifies a different query to copy the column configuration from to use as the default for the new query. If you do not specify a query to copy from, the new query uses the column configuration of the original query.

Note

`--fields=field, field,...`, `--[no]sortAscending`, and `--sortFieldfield=` are not mandatory, but if specified, they overwrite the current default column configuration.

- `--fields=field,field,...`

  specifies the fields to use as the default columns for your query. This overrides the current default columns.
- `--[no]sortAscending`

  specifies the sort direction that issues are displayed in. This overrides the current default sort direction.
- `--sortField=field`

  specifies the field that issues are sorted by. This overrides the current default sort field.
- `--shareWith =u=user1[:modify],user2[:modify],.. ;g=group1[:modify],group2[:modify],..`

  specifies the users and groups that can use and modify the query. Your administrator defines users and groups.
- `--image=[none|default|path ]`

  specifies whether an image appears for the new query.

  `--image=none` does not specify an image for the query.

  `--image=default` specifies the default funnel image for the query.

  `--image=path` specifies the path and name of a custom image for the query, for example, .

  `c:\images\defect_icon.gif`

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--shareWith==user1[:modify],user2[:modify],..;g=group1[:modify],group2[:modify],..`

  specifies the users and groups that can use and modify the query. Your administrator defines users and groups.
- `--description=value`

  specifies a short description for the new query.
- `--name=value`

  specifies the name of the new query.
- `--[no]referenceOriginalQuery`

  specifies whether the original query definition is referenced in the new query as a subquery. If the original query definition is referenced, any changes to the original query definition are reflected in the new query. The default is to copy the original query, not reference it.
- `--queryDefinition=query`
- specifies a string to define the query constraints. For details of the query format, see the `im createquery` reference page.

Note

To reference the original query definition in the new query so that any changes to the original query definition are reflected in the new query, define <*subquery*> as *subquery[OriginalQuery]*.

- `--queryDefinitionFile=`*`value`*

  specifies a file that contains the complete definition of the query. See `--queryDefinition` for the file format.
- `--sharedAdmin`

  specifies the query as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

Caution

Once a user object is converted to a system provided object, you cannot revert it to a user object again.

- *`[user:]query`*

  specifies the name of the user who the copied query belongs to and the query name, for example, `jhoyt:Cosmos Critical Defects`. You do not have to specify the user name, but you must specify the query name. This option is useful when multiple users have the same name for a query.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` query created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the `jhoyt:CosmosDefects` query created by any user.

See Also

- Commands: im createquery, im deletequery, im editquery, im viewquery, im queries
- Miscellaneous: options

im copyreport

copies the properties of an existing report to create a new report

Synopsis

```
im copyreport[--query=[user:]query] [--reportTemplate=value] [--reportTemplateFile=value] [--
shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--name=value] [--description=value] [--
sharedAdmin][--hostname= value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-
N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]][user:]report
```

Description

`im copyreport` copies the properties of an Integrity report to a new Integrity report. For more information on reports, refer to the *PTC Integrity User Guide*.

For example,

```
im copyreport --query=mchang:"Docs Items Worked On This Week" --name="Docs Weekly Status" jriley:"Docs Daily Status"
```

copies the `Docs Daily Status` report created by `jriley` to a new report called `Docs Weekly Status`, based on a query created by `mchang`.

Note the following:

- Reports can do more than just display field information. You can also perform arithmetic calculations between numeric fields, displaying the values in the report. For example, you can add up column totals or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- A report can be edited by the user who created it. Principals (users and groups) that a report is shared to can edit it if they have edit permissions assigned to them by the report creator. A report can only be deleted by the user who created it or by the administrator.
- Because reports are based on queries, reports are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or contact your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Creating deeply nested reports with a large number of inter-related issues can create extremely large reports and/or cause the Integrity Server to stop responding. When creating a report, take into consideration that the average number of links per issue and the number of levels in the report multiply the size of the report.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- Although the electronic signature fields Signed By and Signature Comment are only visible in an issue's history (if enabled by your administrator), you can report on the historical values by specifying the fields in the report.

Options

This command takes the universal options available to all`im` commands, as well as some general options. See the options reference page for descriptions.

- `--query=`*`[user:]query`*

  specifies the name of the query that defines the selection criteria for the report, and the user who created the query.
- `--reportTemplate-`*`value`*specifies the report template on which the report is based.

  For information on the report template format, see the *PTC Integrity Server Administration Guide*.
- `--reportTemplateFile=`*`value`*

  specifies the file name that contains a report template. For information on the report template file format, see the *PTC Integrity Server Administration Guide*.
- `--shareWith=u=`*`user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`*specifies the users and groups that can use and modify the report. Your administrator defines users and groups.
- `--name=`*`value`*

  specifies the name of the report.

Note

Note: If you do not specify a different name for the new report, Integrity adds `Copy of` as a prefix to the query name, for example, `Copy of Cosmos Critical Defects`.

- `--description=`*`value`*

  specifies a description for the report.
- `--sharedAdmin`

  specifies the report as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.

Note

If the report you are copying is an admin report, the `--sharedAdmin` option is not set in the copy.

- *`[username:]report`*

  specifies the name of the report to copy, and the user who created that report. This is useful when multiple users have the same name for a report.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a report name. If Integrity fails to find a matching user name and report name, it searches for a report name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` report created by `jhoyt`. If Integrity cannot find the report and/or user, it searches for the `jhoyt:CosmosDefects` report created by any user.

See Also

- Commands: im createreport, im editreport, im viewreport, im runreport, im deletereport, im reports
- Miscellaneous: options

im copytrigger

copies the details of an existing event trigger to create a new trigger

Synopsis

```
im copytrigger [--name=name] [--type=[scheduled|rule|timeentry|copytree|branch|label|testresult|lock|unlock]] [--runAs=user] [--
query= [user:]query] [--position=<number>|first|last|before:<name>|after:<name>] [--description=value] [--frequency=
[manual|hourly|daily|monthly]] [--script=filename] [--scriptParams= [arg=value[;arg2=value2...]]] [--
scriptTiming=pre|post|pre,post|none] [--assign= [field=value[;field2=value2...]]] [--rule=value] [--copyRule=trigger] [--
ruleFile=filename] [--hostname=server] [--user=value] [--password=password] [--port=number] [(-?|--usage)] [(-Ffile|--
selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--
settingsUI=[gui|default]] [--status=[none|gui|default] trigger
```

Description

`im copytrigger` copies the details of an existing event trigger for workflows and documents and creates a new trigger. Copying a trigger is useful for quickly creating a new trigger that shares common details with an existing trigger. Any options that are used override the settings in the original trigger.

When naming a copied trigger, you cannot use a name that already exists.

An event trigger for workflows and documents contains a list of script files that Integrity runs based on either a defined rule, or a scheduled query. Event triggers must reside on the server machine, where the Integrity Server executes them. Integrity does not support client-side event triggers for workflows and documents. As a result, all references must be relative to the server. For more information on event triggers, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=`*name*

  specifies the name of the trigger. May be a maximum of 100 characters and cannot contain square brackets. If not specified, the name is recorded as `Copy of` original trigger name.
- `--type=`*[scheduled|rule|timeentry|copytree|branch|label|testresult|lock|unlock]*

  specifies the type of trigger. If you do not specify a type, `rule` is specified by default. If you specify `timeentry`, the values for `--rule`, `ruleFile`, `--runAs`, `--frequency`, `--assign`, and `--query` are ignored.

Note

Scheduled event triggers are evaluated on the Integrity Server's time zone.

- `--runAs=`*user*

  specifies the user for a scheduled trigger. All modifications appear as though they were made by the specified user.
- `--query=`*[user:]query*

  specifies the name of the query and to use for a scheduled trigger, and the username of the user who created the query.
- `--position=`*<number>|first|last|before:<name>|after:<name>*

  specifies the position of trigger in the run order.
- `--description=`*value*

  specifies description of what the trigger does.
- `--frequency=`*[manual|hourly|daily|monthly]*

  specifies the frequency of a scheduled trigger. May be the following:

  ```
  [manual]
  [hourly [start=[00:]mm]              [hours=00,01,...,23]]
  [daily [start=hh:mm]                 [days=mon,tue,...]]
  [monthly [start=hh:mm]               [day=1-31] [months=jan,feb,...]]
  ```

- `--script=`*filename*

  specifies the filename of the script, for example, `scriptfile.js`. The script must be located under the following directory: `<installdir>/data/triggers/scripts`.
- `--scriptParams=`*[arg=value[;arg2=value2...]]*

  specifies the list of script arguments.
- `--scriptTiming=`*[pre|post|pre,post|none]*

  specifies if the script should be run pre, post, both pre and post on issue commit to the database, or no timing is associated with the script. If you do not specify an option, `none` is specified by default.
- `--assign=`*[field='Value1';'Field2'='Value2'...]]*

  specifies the list of field assignments.
- `--rule=`*<rule>*

  specifies the rule to associate with the trigger. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *today*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can choose yourself by specifying "`me`". "`me`" is a symbolic user which refers to the currently logged in user. For example, you could create an event trigger that specifies Project issues can only be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
  - Configuration management project fields are invalid in event trigger rules.

- `--copyRule=`*trigger*

  copies the given trigger's rule as it exists at the time of the copy. This is not a pointer to another rule.
- `--ruleFile=`*filename*

  `--rule` for notes on the command use and where to obtain file format information.
- *trigger*

  specifies the name of the trigger you want to copy.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers, im echo
- Miscellaneous: options

im copytype

copies the details of an existing issue type and allows you to rename it as a new issue type

Synopsis

```
im copytype [--addWordTemplate=displayName,templatePath=path[,description=description][,defaultEdit]] [--removeWordTemplate=value] [--editPresentation=value [--printPresentation=value [--printReport=report [--viewPresentation=value [--name=value] [--image=[none|<path>] [--description=value] [--[no]enableRevision] [--majorRevisionRule=rulename] [--minorRevisionRule=rulename] [--copyMajorRevisionRule=type] [--copyMinorRevisionRule=type] [--majorRevisionRuleFile=filename] [--minorRevisionRuleFile=filename] [--position=number] [--[no]allowChangePackages] [--createCPPolicy=value] [--visibleFields=field:group,group...[;...]] [--notificationFields=field,field,...] [--stateTransitions=state:state:group|dynamic group,group|dynamic group,...[;...]] [--[no]duplicateDetectionMandatory] [--duplicateDetectionSearchField=field] [--documentClass=[none|segment|node|shareditem]] [--associatedType=type] [--significantFields=field,field...] [--defaultReferenceMode=[Reuse|Share]] [--addLabelRule=value] [--moveLabelRule=rule] [--copyMoveLabelRule=type] [--moveLabelRuleFile=filename] [--properties=name:value:description[;...]] [--addLabelRuleFile=value] [--branchRule=rule] [--branchRuleFile=value] [--copyAddLabelRule=type] [--copyBranchRule=type] [--copyCopyTreeRule=type] [--copyDeleteLabelRule=type] [--copyTreeRule=type] [--copyTreeRuleFile=value] [--deleteLabelRule=rule] [--deleteLabelRuleFile=value] [--[no]enableDeleteItem] [--deleteItemRule=value] [--deleteItemRuleFile=value] [--copyDeleteItemRule=type] [--[no]enableBranch] [--[no]groupDocument] [--[no]enableCopyTree] [--[no]enableLabel] [--testRole=[none|testSession|testCase|testStep|testSuite]] [--[no]enableTestSteps] [--[no]enableTestResultRelationship] [--testCaseResultFields=field,field,...] [--testSessionDateField=value] [--modifyTestResultPolicy=value] [--editabilityRule=rule] [--editabilityRuleFile=filename] [--copyEditabilityRule=type] [--copyFields=field,field,...] [--mandatoryFields=state:field,field,...[;...]] [--addFieldRelationship=constraint] [--fieldRelationships=constraint[;constraint]] [--removefieldRelationship=constraint] [--fieldRelationshipsFile=value] [--[no]showHistory] [--[no]showWorkflow] [--phaseField=field] [--[no]enableProjectBacking] [--[no]enableTimeTracking] [--permittedAdministrators=u=user1,user2,...;g=group1,group2] [--permittedGroups=group,group,...] [--[no]allowDocumentLocks] [--documentLockPolicy=value] [--documentUnlockGroup=group] [--[no]allowAdditionalLockFields] [--additionalLockFieldsRule=rule] [--additionalLockFieldsRuleFile=filename] [--copyAdditionalLockFieldsRule=type] [--additionalSegmentLockFields=field,field,...] [--additionalContentLockFields=field,field,...] [--[no]lockingRequired] [--lockingRequiredRule=rule] [--lockingRequiredRuleFile=filename] [--copyLockingRequiredRule=type] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]] [--status=[none|gui|default]] type
```

Description

`im copytype` copies the details of an existing issue type and allows you to rename it as a new issue type. Copying a type is useful for quickly creating a new type that shares common details with an existing type. For example:

```
im copytype --name=Defect Requirement
```

copies the *Requirement* type's information to the *Defect* type.

When naming a copied type, you cannot use a name that already exists.

Note

Caution: A type can contain a maximum of 1000 fields. Exceeding the maximum number of fields results in exception errors when creating, editing, or viewing the issue type in the GUI.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--editPresentation=`*value*

  specifies the presentation template to use when editing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the .
- `--printPresentation=`*value*

  specifies the presentation template to use when printing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--printReport=`*report*

  specifies the administrator report that will provide the structure and format when a user prints a document using **Document ▸ Print** from the GUI. This must be specified with `--documentClass=`*segment*.
- `--viewPresentation=`*value*

  specifies the presentation template to use when viewing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--name=`*value*

  specifies the name of the type. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--image=[`*none*`|<`*path*`>]`

  specifies whether an image appears for the type.

  `--image=`*none* does not specify an image for the type.

  `--image=<`*path*`>` specifies the path and name of a custom image for the type, for example, `c:\images\type_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=`*value*

  specifies a description of the type.
- `--[no]enableRevision`

  specifies whether to enable item revisioning for the specified item type or enable document versioning for the specified document model type. For more information on item revisioning or document versioning, see the *PTC Integrity Server Administration Guide*.
- `--majorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the major revision. For example, if the major rule is `Type=Requirement`, then only items of type Requirement may have major revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--minorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the minor revision. For example, if the minor rule is `State=Open`, then only items in an open state may have minor revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyMajorRevisionRule=`*type*

  copies the major revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one being created.
- `--copyMinorRevisionRule=`*type*

  copies the minor revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one being created.
- `--majorRevisionRuleFile=`*filename*

  specifies a file that contains the major revision rule for the specified item type (used with item revisioning) or document model type (used with document

versioning). For the file format, see Specifying Rules on the options reference page.

- `--minorRevisionRuleFile=`*`filename`*

  specifies a file that contains the minor revision rule for the specified item type (used with item revisioning) or document model type (used with document versioning). For the file format, see Specifying Rules on the options reference page.

- `--documentClass=`*`[none|segment|node|shareditem]`*

  allows you to specify a document class on a type. Each document domain requires a segment, a node type associated with the segment, and a shared item type associated with the node type. Each document instance requires the segment root and a set of nodes. The options are as follows:

  - `none` specifies that this item type is not used in documents. The default for all non-document types.
  - `segment` specifies that this type is used in the segment root. For example, you need to specify segment if you are creating a document domain.
  - `node` specifies that this type is used in nodes (content). Node correlates to the content (i.e. Requirement, Input, Test, Specification) item type. To create content, you need a node and a corresponding shared item. Therefore, you would create two types: one with `documentClass=node`, and the other with `documentClass=shared item`. Each node in a document is a reference to a shared item or a subsegment, and all three types expose arbitrary fields. Nodes also expose FVA fields from the shared item.
  - `shared item` specifies that this type is a shared item. Shared items should be associated with a corresponding node type (content). Each node class on a type requires an associated shared item.

  Example:
  `im createtype --name=Document --description="Requirement Document" --documentClass="segment" --associatedType="Requirement(node name)"`

  creates a segment with an associate node type called `Requirement`.

- `--[no]duplicateDetectionMandatory`

  makes it mandatory for users to view potential duplicates before they create a new item.

- `--duplicateDetectionSearchField=`*`field`*

  specifies the name of the short text field to search when using duplicate detection. Setting an empty field means that duplicate detection will not be available to users. The short text field cannot be a computational field or Field Value Attribute to a short text field. Before enabling duplicate detection, the selected short text field must be visible on the type. In addition, to use duplicate detection searches on a short text field, the created field must have text search indexing enabled. When searching for potential duplicates, the default number of results is 20 items.

- `--testRole=[`*`none`*`|`*`testSession`*`|`*`testCase`*`|`*`testStep`*`|`*`testSuite`*`]`

  allows you to specify the test management role for the type. The options are as follows:

  - `none` specifies that this item type does not a have a specific test management role. However, it can still be related to test results using the `--[no]enableTestResultRelationship` option.
  - `testSession` specifies that this type is a test session. A test session is a concrete run or execution of a group of test cases. If a type is a test session, also specify `modifyTestResultPolicy`.
  - `testCase` specifies that this type is a test case. A test case describes the test conditions and provides a container for adding test results from the test session. A test case must have the `--documentClass` option set to node. If a type is a test case, also specify `--[no]enableTestSteps` and `--testCaseResultFields`.
  - `testStep` specifies that this type is a test step. A test step is a specific testing operation performed as part of executing a test case.
  - `testSuite` specifies that this type is a test suite. A test suite is a grouping of test cases. A test case must have the `--documentClass` option set to segment.

- `--associatedType=`*`type`*

  specifies an associated type for segments and nodes. If the document class is `segment`, the associated type is of type node. If it is `node`, the type is of shared item.

- `--defaultReferenceMode=`*`[Reuse|Share]`*

  specifies the reference mode to apply to nodes after they are branched. `Reuse` points to the same shared item until the original node changes and a new shared item is created.`Share` points to the same shared item as the original branched node. Editability will not be allowed on shared fields; this mode only allows observation of the changes made on the other item.

- `--significantFields=`*`field,field...`*

  allows you to specify additional fields by type that, when edited, results in an update to the content revision. This is reflected in a change to the revision date in the item history. Each type has default associated significant edit fields.

- `--position=`*`number`*

  specifies the position in the list of types.

- `--addLabelRule=`*`rule`*

  specifies the rules for when users are permitted to add a label rule. For the rule syntax, see Specifying Rules on the options reference page.

- `--addLabelRuleFile=`*`filename`*

  specifies a file that contains the add label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page..

- `--moveLabelRule=`*`rule`*

  specifies the rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--copyMoveLabelRule=`*`type`*

  specifies the type from which to copy rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--moveLabelRuleFile=`*`filename`*

  specifies a file that contains the move label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page.

- `--branchRule=`*`rule`*

  specifies the rules for branching by users. For the rule syntax, see Specifying Rules on the options reference page.

Note

Users or groups do not require editability of the issue or any given field to be able to branch an issue; however, they must have project or type visibility to branch an issue. If a user does not have visibility to a given field during a branch operation, the field will be copied unchanged.

- `--branchRuleFile=`*`filename`*

  specifies a file that contains the branch rule set for the specified issue type. For the file content format, see Specifying Rules on the options reference page.

- `--copyAddLabelRule=`*`type`*

  specifies the rules for when users or groups are permitted to copy an add label rule.

- `--copyBranchRule=`*`type`*

  specifies the rules for when users are permitted to copy a branch rule.

- `--copyCopyTreeRule=`*`type`*

  specifies the rules for when users or groups have the ability to copy the copy tree rule for the specified issue type.

- `--copyDeleteLabelRule=`*`type`*

  copies the delete label rule of an existing type to the one being copied.

- `--[no]enableBranch`

  specifies whether issues of this type can be branched. For more information on branching, see the *PTC Integrity User Guide*.

- `--[no]enableCopyTree`

specifies whether users or groups can copy a tree of issues belonging to the hierarchy for this type.

- `--[no]enableLabel`

  specifies whether users or groups can add labels to issues of this type.
- `--copyTreeRule=`*`rule`*

  copies the tree rule of an existing type to the one being copied.
- `--copyTreeRuleFile=`*`value`*

  specifies a file that contains the copy tree rule set for the specified issue type.
- `--deleteLabelRule=`*`type`*

  specifies the rule for when users or groups can delete label rules.
- `--deleteLabelRuleFile=`*`value`*

  specifies a file that contains the delete label rule set for the specified issue type.
- `--[no]enableDeleteItem`

  specifies whether items of the specified type can be deleted. If items of the specified type can be deleted, a rule set must specified with `--deleteItemRule=`*`value`* or `--deleteItemRuleFile=`*`value`*.
- `--deleteItemRule=`*`value`*

  allows specific users or groups the ability to delete items of the specified item type. To change the rule on who can delete items, users or groups require the `ModifyDeleteItemRule` permission.

Note

Important: The `ModifyDeleteItemRule` permission differs from the `DeleteItem` permission, which allows users or groups to delete items of any type without a defined rule. To define strict item deletion rules in your organization, PTC recommends defining a delete item rule and assigning the `ModifyDeleteItemRule` permission to the appropriate users and groups.

- `--deleteItemRuleFile=`*`value`*

  specifies a file that contains the rule set for deleting items of the specified item type.
- `--copyDeleteItemRule=`*`type`*

  specifies a type to copy an existing delete rule set from.
- `--[no]allowChangePackages`

  specifies whether configuration management change packages are permitted for the type.
- `--[no]groupDocument`

  specifies whether items of the specified type will be used to group content. For group documents, the `--documentClass` option must be `segment`.
- `--[no]enableTestSteps`

  specifies whether the test case type can use test steps. A test step is a specific test for a test case. A test case can contain an ordered list of steps to follow in order to complete the test. Test steps can be shared across test cases.
- `--[no]enableTestResultRelationship`

  specifies whether the type can be related to test results. For example, defects for failed test results.
- `--testCaseResultFields=`*`field,field,...`*

  specifies fields from the test case type that will display in the Test Result Details view.
- `--modifyTestResultPolicy=`*`value`*

  specifies which users and/or groups are allowed to modify test results for the test session item type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to the test session are permitted to create test results. Valid values are `userField=<`*`field`*`>`, `groupField=<`*`field`*`>`, *`anyone`*, *`groups=group1,group2,...`*.
- `--createCPPolicy=`*`value`*

  specifies which users and/or groups are allowed to create change packages against issues of this type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to issues within that type are permitted to create change packages. Valid values are `userField=<`*`field`*`>`, `groupField=<`*`field`*`>`, *`anyone`*, *`groups=group1,group2,...`*.
- `--visibleFields=`*`field:group,group...[;...]`*

  specifies which groups have visibility for which fields. By default, the everyone group is specified for each field you specify. All previous values are replaced with specified values.

Note

Special fields are always visible, even if they are not specified.

- `--notificationFields=`*`field,field,...`*

  specifies notification fields for including in every email notification related to this type.

Note

SI project and attachment fields cannot be specified.

- `--copyFields=`*`field,field,...`*

  specifies the list of fields to be copied by default for items of this type. This setting overrides `--copyCommonFields` if set. Users can override these default fields by adding and removing any fields that they can view and edit.
- `--stateTransitions=`*`state:state:group|dynamic group,group|dynamic group,...[;...]`*

  specifies a state transition from one state to another, and the groups/dynamic groups permitted to make that state transition. At least one group/dynamic group must be specified for each transition.
- `--properties=`*`name:value:description[;...]`*

  defines type properties. Type properties provide custom code (such as triggers and API) with attributes to read and act on based on their values. The following can be specified for each property:

  `name` specifies the name of the property.

  `value` specifies the value for the property.

  `description` specifies an optional description for the property.
- `--editabilityRule=`*`rule`*

  specifies the rules for when users are permitted to edit the type. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *`MM/dd/yyyy h:mm:ss [AM|PM]`* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *`today`*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *`now`*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *`none`*.

- When specifying a user, you can choose yourself by specifying "me". "me" is a symbolic user which refers to the currently logged in user. For example, you could create an editability rule that specifies a Project type issue can be edited if the currently logged in user is one of the users defined in the multi-valued Stakeholders field.
- SI project and attachment fields cannot be specified.
- If a group name contains blank spaces, a second pair of quotes around the group name is required. For example:

  im editfield --hostname=server --port=7001 --editabilityRule="((user is a member of "Project Management") or (user is a member of ""Project Management""))" fieldname

---

- --editabilityRuleFile=*filename*

  specifies a file that contains the issue editability rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- --copyEditabilityRule=*type*

  copies the issue editability rule of an existing type to the new one being created.
- --mandatoryFields=*state:field,field,...[;...]*

  specifies mandatory fields for a state in the type's workflow. All previously specified mandatory fields are replaced with the new values. Mandatory fields can also be set on type constraints. For more information on constraints, see the *PTC Integrity Server Administration Guide*.
- --addFieldRelationship=*constraint*

  specifies a single constraint (field relationship) to be added to a type.
- --removeFieldRelationship=*constraint*

  specifies a single constraint (field relationship) to be removed from a type.
- --fieldRelationships=*constraint[;constraint]*

  specifies constraints between fields. All previously existing constraints on this type are replaced with the new constraints specified here. The syntax specified below also applies for the --addFieldRelationship and --removeFieldRelationship options.

  There are four constraint methods: Basic, Field Relationship, Rule, and Item Backed Pick List (IBPL). For more information on constraints and constraint methods, see the *PTC Integrity Server Administration Guide*.

Note

Important: If you are creating or copying a type, you cannot reference the name of that new type when creating a basic constraint. Other rule-based constraints, such as Rule and IBPL, can be created; however, errors may occur if you attempt to create a constraint based on a rule that includes the new type name, for example, (Field[Type]=*new type name*).where *constraint* is one of:

a Basic or Field Relationship constraint:

sourceField=sourceValue1[,sourceValue2,...]:targetValue [:[all][,mandatory][,errInvalidated=*invalidMessage*] [,errMandatory=*mandatoryMessage*][,description=*descriptionText*]]

a Rule constraint:

constraintrule=(*rule*):*targetValue* [:[all][,mandatory][,errInvalidated=*invalidMessage*][,errMandatory=*mandatoryMessage*] [,description=*descriptionText*]]

an IBPL constraint:

rule=(*ibplRule*):*ibplField* [:[mandatory][,errInvalidated=*invalidMessage*][,errMandatory=*mandatoryMessage*] [,description=*descriptionText*]]

and where *sourceField* is any field with predefined values (Pick, User, Group, IBPL, Boolean, Project, State, Phase, Type). For the basic constraint method, *sourceField* must be the type you are editing, and the source value must be the specified type.

and where *targetValue* is:

targetField=[value1][,value2,...] where *targetField* is a field of type other than User/Group.

or

targetField=[value1][,value2,...][,hasProjectPermission] where targetField is a field of type Group.

or

targetField=[value1][,value2,...][,hasProjectPermission][memberOf*(dynamicGroup1[,dynamicGroup2,...])*][valueOf*(group)*] where *targetField* is a field of type User.

targetField can be any field that is visible on the type and that is not the *sourceField*. Only the following fields can have values specified: User, Group, Pick, State, Project, IBPL, Logical. All other fields (such as Attachment, Integer, Short Text, etc...) can only be made mandatory.

Specifying values (*value1, value2,* etc...) is optional; however, if not specified, the all and mandatory options must be used to indicate that the field is mandatory and all values are acceptable.

Note

For all *targetField* types, if a mandatory option is specified, then at least one value (*value1,value2, ...*) must be specified, or the all option must be present.

hasProjectPermission constrains the values of the field to only those users or groups that have permissions for the item's project. This option cannot be specified with any values (value1, value2...), or with memberOf or valueOf options.

memberOf constrains the values of the field only to those users that are a members of the listed dynamic groups (*dynamicGroup1, dynamicGroup2, ...*). Note that the everyone group can also be used as a value for the dynamic groups. This option cannot be specified with any values (*value1, value2...*), or with hasProjectPermission or valueOf options.

valueOf constrains the values of the field to only the specified group. This option cannot be specified with any values (value1, value2...), or with hasProjectPermission or memberOf options.

and where *rule* is a specified rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

and where *ibplRule* is a specified IPBL rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

The IBPL rule can be made up of:

- sub-rules on the field values of the items backing the IBPL target. In the CLI, add an apostrophe (') at the end of the field to indicate the Target Field option in the GUI.
- sub-rules on the item being edited. If no apostrophe is added, then the 'Editing Item Value' GUI option is selected.

In addition:

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

and where *ibplField* is an IBPL field that is visible on a type.

and where *invalidMessage* is any string up to 2000 characters.

and where *mandatoryMessage* is any string up to 2000 characters.

~~and where *descriptionText* is any string up to 200 characters.~~

Examples:

Basic Constraint Method:
`Type=AdminRequest:"Assigned User"=memberOf(manager,administrator):mandatory,errMandatory="The {ConstrainedField} cannot be empty."`

Specifies that all items of the `AdminRequest` type must be assigned to a user who is either in the manager or an administrator group, and that the "Assigned User" field is mandatory.

Field Relationship Constraint Method:
`Importance=High,Critical:Escalated=True:mandatory,description="Ensure that the Incident is escalated if Importance is either High or Critical".`

Specifies that an incident is escalated if `Importance` is either `High` or `Critical`.

Rule Constraint Method:
`constraintrule=(field[importance]>"8"):"Assigned User"=valueOf(manager):mandatory`

Specifies that if the item is very important (importance > 8), then assign the item to a manager.

IBPL Constraint Method:
`rule=(field["State"] = "Prioritized"):Priority`

Specifies that if the state is `Prioritized`, then the `Priority` IBPL is visible.
`rule=(field'["Graduated"] = true):Students`

Filters the list of students such that the only values displayed are those where the student's backing item has `Graduated` set to true.

Note

You can specify the options for `--fieldRelationships`, `--addFieldRelationship`, and `--removeFieldRelationship` at the same time. If all three options are specified at the same time, the constraints on the type are modified in the following order:

1. `--fieldRelationships` is executed first and all constraints are replaced by the ones specified here.
2. `--removeFieldRelationship` is executed next and the specified constraints are removed from the ones specified in the `--fieldRelationships` option.
3. `--addFieldRelationship` is executed last and the specified constraints are added to the type.

The options for `--removeFieldRelationship` and `--addFieldRelationship` can be specified multiple times on the command.

Note

If you need to specify a large number of field relationships, use the `--fieldRelationshipsFile` option.

- `--fieldRelationshipsFile=`*value*

specifies the name of the file containing the constraint (field relationship). Multiple files should use the same format as `--fieldRelationships`, with all constraints on the same line and separated by semi-colons. For information on constraints, see the *PTC Integrity Server Administration Guide*.

Note

If you specify both `--fieldRelationships` and `--fieldRelationshipsFile`, only `--fieldRelationships` is used.

- `--[no]showHistory`

specifies whether to display the item history for all items of the selected type. When you set `--noshowHistory` for an existing type, the item history is always hidden from users when they are viewing or editing items of that type. The setting applies in all Integrity Client interfaces. For example, in the Integrity Client GUI and Web interface, when you set the `--noshowHistory` option, the `History` tab is no longer displayed for items of the selected type. By default, the item history is displayed for newly created types.

- `--[no]showWorkflow`

specifies whether to display the `Workflow` tab to the user in the `Create Issue` view, `Edit Issue` view, and `Issue Detail` view. The `Workflow` tab contains a read-only display of the issue type workflow to the user. Users in the Web interfaces do not have a user preference to override the display of the `Workflow` tab. Users launching the GUI view from the CLI can override the display of the Workflow tab on a per command basis, if workflow for that type is enabled.

- `--phaseField=`*field*

specifies the phase field to display in the `Workflow` tab of the `Create Issue` view, `Edit Issue` view, and `Issue Detail` view. Only a phase field that is specified in the `--visibleFields` option may be specified. This option must be specified with the `--showWorkflow` option.

- `--[no]enableProjectBacking`

Enables the type to back a project, allowing you to create a link between an issue of this type and a project in the `Project` field. By creating a Project issue type and specifying this option, then creating a Project issue and linking it to a specific project (using the `im createissue` command), the power and workflow of projects as issues becomes available. Important metadata and metrics can be recorded in the Project issue, for example, the assigned Project Manager, estimated and actual budgets (using computed fields), and important milestone dates. Linking a Project issue to a project also reduces possible confusion about the details of projects in your database.

Note

- A link between a Project issue and a project is optional.
  - If you enable this option, creating issues of this type and linking them to projects is useful only to certain users. You may want to prevent most users from being able to create issues of this type, for example, you can allow only users in the `ProjectManagers` group to create Project issues. To restrict type visibility, see the `--permittedGroups` option.
  - Projects and Project issues can be used independent of one another; you do not have to create a Project type to use the `Project` field.
  - Multiple types can back projects; however, only one issue may back a given project.
  - To enable this option, you must be an administrator for the specified type. This option can be disabled at any time.
  - This option cannot be specified unless the `Project` field is specified as a visible field for this type.
  - To create the issue that backs a project, you must be an administrator for the specified project and belong to a group that has permission to create issues of that type.
  - When you create an issue to back a project, Integrity warns you if there is more than one type that can back a project, displaying a list of types that have this option enabled and that you have "view" and "modify" permissions for. Specify the type that you want to back the project.
  - For Admin Staging, you cannot stage issues. Issues that back projects created on a Staging server will not be staged; you must re-create the issues that back projects on the production server. Rules, queries, or computed fields that explicitly mention the issue number that backs a project will not work after they have been transferred from the staging server to the production server.

- `--[no]enableTimeTracking`

Allows one or more users to allocate time spent working on issues of this type. When enabled, users can specify time entries when they edit an issue. In addition, a `Time Entries` tab is available when you view an issue of this type. You can use time entries to develop metrics (in the form of queries, charts, and reports) that measure the amount of effort spent on projects.

Note

- To create, edit, and delete time entries on behalf of other users, the `TimeTrackingAdmin` ACL permission is required.
- The ability to create, edit, and delete time entries is governed by state-based capabilities. For more information, see the `im createstate` and `im editstate` commands.

- `--permittedAdministrators=u=user1,user2,...;g=group1,group2,...`

  specifies a comma delimited list of users and/or groups that can administer this type. Integrity Type Administrators are allowed to edit and view types. Type Administrators are not allowed to assign themselves as administrators to any other types, or to edit types that they don't administer. A Type Administrator can create fields, but can only edit fields that are referenced by a type they administer. For more information, see the *PTC Integrity Server Administration Guide*.
- `--permittedGroups=group,group,...`

  specifies a comma delimited list of groups that have visibility for this type.
- `--addWordTemplate=name=templateName,path=templatePath[,description=description][,defaultEdit]`

  specifies a template to add to the type for users to use when editing items in Microsoft® Word. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
  - *name*

    specifies the name of the template. The name is visible to users if more than one template is available for selection.
  - *path*

    specifies the path of the template to add to the type. The file must be DOTX format.
  - *description*

    specifies an optional description for the template.
  - *defaultEdit*

    specifies to use the template as the default template for users editing an item or document in Word. Specifying this option pre-selects the template for the user. However, the user may still select a different template if needed. This option can be used to streamline the edit process for users. Only one template per type can have this option enabled. Enabling this option on one template clears it from any other that has it specified.
- `--removeWordTemplate=value`

  specifies a Microsoft® Word template to remove from the type. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
- `--[no]allowDocumentLocks`

  specifies whether document locking is supported for the specified type.
- `--documentLockPolicy=value`

  specifies which users and/or groups are allowed to lock documents of the specified type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to documents of this type are allowed to lock them. Valid values are `userField=<field>`, `groupField=<field>`, `anyone`, `groups=group1,group2,...`.
- `--documentUnlockGroup=group`

  specifies the lock administration group for the type. Members of the specified group can unlock any locked document of the specified type.
- `--[no]allowAdditionalLockFields`

  specifies whether additional fields (beyond significant fields) can be locked for the specified type. By default, only significant fields are affected by document locking.
- `--additionalLockFieldsRule=rule`

  specifies a rule that determines when additional fields (if allowed) are affected by locking. For the rule syntax, see Specifying Rules on the options reference page.
- `--additionalLockFieldsRuleFile=filename`

  specifies a file that contains the additional locks field rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyAdditionalLockFieldsRule=type`

  copies the additional field locks rule of an existing type to the new one being created.
- `--additionalSegmentLockFields=field,field,...`

  specifies the fields on a segment root of the specified type that, in addition to the significant fields, are affected by document locking.
- `--additionalContentLockFields=field,field,...`

  specifies the fields on content nodes of the specified type that, in addition to the significant fields, are affected by document locking.
- `--[no]lockingRequired`

  specifies whether a document of the specified type must be locked before the significant fields (plus any defined additional fields) can be edited.
- `--lockingRequiredRule=rule`

  specifies a rule that defines when documents of the specified type must be locked before the appropriate fields can be edited. For the rule syntax, see Specifying Rules on the options reference page.
- `--lockingRequiredRuleFile=rule`

  specifies a file that contains the rule that defines when locking is required for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyLockingRequiredRule=rule`

  copies the rule that defines when locking is required for an existing type to the new type being created.
- `--hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]`

  specifies the menus and related dialog boxes where creation or insertion of the item type is to be hidden. When an item type is hidden, it can still be created or inserted from the command line. The options are as follows:
  - *none* specifies that the item type is not to be hidden from any menu. This is the default. No other option can be specified along with this option. When item types are to be hidden from menus, multiple options can be specified.
  - *CreateItem* specifies that this item type is to be hidden from menus and dialog boxes for creating items.
  - *CreateRelatedItem* specifies that this item type is to be hidden from menus and dialog boxes for creating related items.
  - *CreateDocument* specifies that this item type is to be hidden from menus and dialog boxes for creating documents.
  - *InsertContent* specifies that this item type is to be hidden from menus and dialog boxes for inserting content in documents.
- *type*

  specifies the name of the type you want to copy.

See Also

- Commands: im createtype, im edittype, im viewtype, im deletetype, im types
- Miscellaneous: options

im cps

displays the attribute information for a selected change package type

Synopsis

`im cps [--attributes=attribute1,attribute2...] [--filter=type:name] [--height=value] [--width=value] [-x value] [-y value] [--[no]batch] [--hostname=value] [--port=value][--password=value] [--user=value] [(-g|--gui)] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)][--cwd=value] [--forceConfirm=[yes|no]] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]issue|issue:change package id...`

Description

`im cps` allows you to display attribute information for the selected change package type. You can select the change package using an issue ID or change package ID. The selected change package does not have to be assigned to you.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--attributes =attribute1,attribute2...`

  specifies the change package attributes. Common change package attributes include `id`, `type`, `status`, `summary`, `createdby`, `entrycount`, and `createddate`.

- `--filter=type:name`

  specifies a change package type to refine the change package selection. Acceptable values are `si` (configuration management change package), `implementer` (Implementer change package), or a custom change package type.

- *issue...*
- *issue:change package id...*

  issue identifies a specific issue that contains all change packages that you want to view; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to view; use spaces to specify more than one change package.

See Also

- Commands: im viewcp
- Miscellaneous: ACL, options

im createchart

creates a new chart

Synopsis

```
im createchart [--chartType=[Distribution|Trend|Issue Fields|Issue Fields Trend]] [--bgColor=value] [--chartFootnote=value] [--chartTitle=value] [--dataColors=value] [--descriptionFont=value] [--[no]displayDescription] [--[no]displayLegend] [--[no]displayLabels] [--endDate=value] [--fieldFilter=field=[value,value,...] [--fieldValues=value] [--footnoteFont=value] [--graphStyle=[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]] [--groupingValues=value] [--[no]is3D] [--[no]isAutoColors] [--[no]isShowZeroFieldCount] [--[no]isShowZeroGroupingCount] [--legendBgColor=value] [--legendPosition=[Right|Bottom|Left|Top]] [--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]] [--legendTitle=value] [--outlineColor=value] [--projectedTrendExpressions=value] [--query=[user:]query] [--startDate=value] [--numberOfSteps=value] [--titleFont=value] [--trendStep=[Hour|Day|Week|Month|Quarter|Year]] [--[no]useIssueDefinedOrigin] [--[no]xReverse] [--[no]xShowGrid] [--[no]xShowTitle] [--yLabelRotation=[Horizontal|VerticalUp]] [--[no]yReverse] [--[no]yShowGrid] [--[no]yShowTitle] [--description=value] [--name=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--sharedAdmin] [--computations=value] [--startDateField=field] [--runDateIsEndDate] [--[no]deltasOnly] [--issueIdentifier=value] [--[no]displayShapesForLineGraphs] [--[no]swapRowsAndColumns] [--[no]displayRowTotals] [--[no]displayColumnTotals] [--rangeDefinitions=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

You can define and store any number of charts using Integrity. A chart is a summary of data presented in a graphical format.

For more information on charts, refer to the *PTC Integrity User Guide*.

For example,

```
im createchart --name="Release 2.0 Docs Issues By User" --chartType=Distribution --query="Release 2.0 Docs Issues" --fieldValues="Assigned User=jriley,mchang" --groupingValues="State=In Progress"
```

creates a new chart showing the distribution of Release 2.0 Docs issues among the specified users.

Note the following:

- A chart can be edited by the user who created it. Principals (users and groups) that a chart is shared to can edit it if they have edit permissions assigned to them by the chart creator. A chart can only be deleted by the user who created it or by an administrator.
- The minimum information required to create a distribution chart is a chart name, a field, and a query. The minimum information required to create a trend chart is a chart name, step type, start and end date, and a field. The minimum information required to create an issue fields chart is a chart name, query, and aggregate expression. The minimum information required to create an issue fields trend chart is a chart name, query, step type, start and end date, and numeric field. All other modifications and additional information are optional.
- Charts can do more than just display field information in a graphical format. You can also perform arithmetic calculations between numeric fields, displaying the values in the chart. For example, you can calculate the average for a group of field values or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- All charts are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or see your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.

For trend charts, the following are valid option combinations:

--startDate

--endDate

--startDate

--numberOfSteps (positive value)

--startDate

--runDateIsEndDate

--startDateField*

--numberOfSteps (positive value

--numberOfSteps (negative value)

--endDate

--numberOfSteps (negative value)

--runDateIsEndDate

*Only applies to Issue Fields Trend charts.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--chartType=[Distribution|Trend|Issue Fields|Issue Fields Trend]`

  specifies the chart type.
- `--chartFootnote=value`

  specifies the footnote text of the chart.
- `--chartTitle=value`

  specifies the title of the chart.
- `--titleFont=value`

  specifies the font to be used for the chart title. Use the following format: `name,style,size`, where `style` is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic, for example, `helvetica,1,10`. When the chart is run, if the specified font cannot be found, Integrity uses a substitute font.
- `--descriptionFont=value`

  specifies the font to be used for the description. Use the following format: `name,style,size` format, where `style` is 0 for plain, 1 for bold and 2 for italic, for example, `helvetica,1,10`.
- `--[no]displayDescription`

  specifies whether to display the chart description.
- `--trendStep=[Hour|Day|Week|Month|Quarter|Year]`

  specifies the interval for each point on a trend or issue fields trend chart graph.
- `--[no]useIssueDefinedOrigin`specifies whether to use the start date defined in an issue field for an issue fields trend chart.
- `--startDate=value`

  specifies the start date for trend or issue fields trend charts. To specify a date and time, type `MM/dd/yyyy h:mm:ss [AM|PM]`.

  Other acceptable date formats include:

  MM/dd/yyyy h:mm:ss a zMM/dd/yyyy h:mm:ss.SSS a zMM/dd/yyyy h:mm:ss aMM/dd/yyyy h:mm:ss.SSS aMM/dd/yyyy
- `--endDate=value`

  specifies the end date for trend or issue fields trend charts. To specify a date and time, type `MM/dd/yyyy h:mm:ss [AM|PM]`. See the `--startDate=value` option for additional date and time formats.
- `--description=value`

  specifies a short description for the chart.
- `--name=value`

  specifies the new name of the chart. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`

  specifies the users and groups that can use and modify the chart. Your administrator defines users and groups.
- `--fieldFilter=field=[value,value,...]`

  specifies how field filters can be applied to the chart when it is run. The first component of the value is the field name. Currently, only project field filters are supported. The second component specifies the project(s) that you want to filter the chart data by when it is run. For example, `--fieldFilter="Project=/Project1"` filters for issues that have a value of `Project1` in the `Project` field. If you do not specify a value, Integrity filters for issues with a value of `Unspecified` in the `Project` field.

  Note

  You can also define project filters for dashboards. Depending on how you design your dashboard, when a chart is run through a dashboard, the dashboard's project filter can override the chart's project filter.

- `--fieldValues=value`

  specifies the field, field values and aliases used by the chart. For example: `--fieldValues=Type=Documentation, Development[Feature Request, Bug]` would include issues that have a `Type` field with a value of `Documentation`, `Feature` or `Bug`, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

  Use * to include all field values, and + to automatically include all future field values. For example: `--fieldValues=Type=*, +, Development[Feature Request, Bug]` would include all current values and any future values for the `Type` field, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

  For more information on specifying chart values, see the *PTC Integrity User Guide*.
- `--footnoteFont=value`

  specifies the font to use for the footnote. Use the following format: `name,style,size` format, where `style` is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic, for example, `helvetica,1,10`.
- `--groupingValues=value`

  specifies the field, field values, and aliases to use to group the data in the chart. For example: `--groupingValues=State=Submit, In Work[In Progress, In Development]` would group chart data into separate components for `Submit` and `In Work`, with `In Work` being a combination of the `In Progress` and `In Development` states.

  Use * to include all field values, and + to automatically include all future field values. For example: `--groupingValues=State=*, +, In Work[In Progress, In Development]` would group chart data into separate components for all current values and any future values for the State field, with `In Work` being a combination of the `In Progress` and `In Development` states.

  For more information on specifying chart values, see the *PTC Integrity User Guide*.
- `--projectedTrendExpressions=value`

  When creating or editing an item fields trend chart, you can specify whether to display a projected trend for one of the displayed item field series, thus supporting the use of `burn-down charts` for planning work remaining within a fixed duration project. The projected trend displays a line between a starting value corresponding to the starting date and the end value corresponding to the end date. In addition, if you choose to display the projected trend, you can also display the actual trend as a line between the last actual series value to the end value corresponding to the end date.

  This option specifies attributes for a projected trend graph in the item fields trend chart, where *value* consists of the following attributes separated by a colon:

  chartedExpression is the expression (numeric field) to chart. This value is mandatory.

  startValueExpression is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the historic item field value. This value is mandatory.

  endValueExpression is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the "now" item field value. This value is mandatory.

projectedTrendLabel is a text string used for the chart legend and tooltips.

showUpdatedTrend displays the projected trend in the chart. Specify `true` or `false` (default).

updatedTrendLabel is a text string used for the chart legend and tooltips.

Tip: To clear the value for `--projectedTrendExpressions=`*value*, specify an empty string.

For example:

```
--projectedTrendExpressions=Effort:0:"Effort":"Planned Effort":true:"Unplanned Effort"
```

- `--query=`*[user:]query*
- specifies the name of the query that the chart is based on.

Note

If the chart is a shared admin object, an admin query is required.

- `--graphStyle=[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]`

specifies the graph style used of the chart.

- `--dataColors=`*value*

specifies the custom data colors to be used using the RGB color model. For example: 'R,G,B;R,G;R,G,B' where R,G and B are within the range 0-255.

If the chart has more data points than the data colors you specify, the colors are repeated. If the `--[no]isAutoColors` option is true, the colors specified here are ignored.

Note

This option is invalid for table style graphs.

- `--bgColor=`*value*

specifies the background color of the chart using the RGB color model. For example: 'R,G,B' where R,G and B are within the range 0-255.

Note

This option is invalid for table style graphs.

- `--[no]displayLegend`

specifies whether to display the chart legend.

Note

This option is invalid for table style graphs.

- `--[no]displayLabels`

specifies whether to display labels for values in the chart. If you select a pie graph style, this option is automatically selected. `--nodisplayLabels` is the default option.

Note

This option is invalid for table graphs.

- `--[no]is3D`

specifies whether to display bar and pie graphs in 3D.

Note

This option is invalid for table style graphs.

- `--[no]isAutoColors`

specifies whether to use the default chart colors. If false, you must provide colors through the data colors option.

Note

This option is invalid for table style graphs.

- `--[no]isShowZeroFieldCount`

specifies whether to include empty field values in the chart.

- `--[no]isShowZeroGroupingCount`

specifies whether to include empty grouping values in the chart.

- `--legendBgColor=`*value*

specifies the background color for the chart legend using the RGB color model. For example: 'R,G,B' where R,G and B are within the range 0-255

Note

This option is invalid for table style graphs.

- `--legendPosition=[Right|Bottom|Left|Top]`
- specifies the legend position in relation to the graph.

Note

This option is invalid for table style graphs.

- `--legendTitle=`*value*

specifies the title for the chart legend.

Note

This option is invalid for table style graphs.

- `--outlineColor=`*value*

specifies the outline color of the graph using the RGB color model. For example: 'R,G,B'

Note

This option is invalid for table style graphs.

- `--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]`

specifies the rotation of the horizontal axis labels for the chart.

Note

This option is invalid for table style graphs.

- `--[no]xReverse`

specifies whether the chart uses a horizontal axis with a reverse orientation (left).

Note

This option is invalid for table style graphs.

- `--[no]xShowGrid`

specifies whether to display horizontal grid lines.

Note

This option is invalid for table style graphs.

- `--[no]xShowTitle`

specifies whether to display the title for the horizontal axis.

Note

This option is invalid for table style graphs.

- `--yLabelRotation=[Horizontal|VerticalUp]`

specifies the rotation of the vertical axis labels for the chart.

Note

This option is invalid for table style graphs.

- `--[no]yReverse`

specifies whether the chart uses a vertical axis with a reverse orientation (down).

Note

This option is invalid for table style graphs.

- `--[no]yShowGrid`

specifies whether to display vertical grid lines.

Note

This option is invalid for table style graphs.

- `--[no]yShowTitle`

specifies whether to display the title for the vertical axis.

Note

This option is invalid for table style graphs.

- `--sharedAdmin`

specifies the chart as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

Important:

Once a user object is converted to a system provided object, you cannot revert it to a user object again.

- `--computations=`*`expression:name:pattern:axis name:minRangeValue:maxRangeValue:tickUnitValue`*

specifies an expression and numeric axes attributes.

Note the following about specifying numeric axes attributes:

- If you specify one set of numeric axes attributes (minimum range, maximum range, and tick unit), these attributes are specified for the X and Y axes. For XY (scatter) charts, PTC recommends against setting individual numeric axes attributes for the X and Y axes.
- For bubble charts, PTC recommends against specifying numeric axes attributes because they override the calculated values provided by the underlying expression and users will have to zoom in/out to properly view chart values.

`expression` specifies an aggregate expression for a distribution chart, a computed expression for an issue fields chart, or a numeric field for an issue fields trend chart. For information on creating expressions, see the *PTC Integrity Server Administration Guide*

`name` specifies the label name for the aggregate expression, computed expression, or numeric field as you want it to appear in the chart. If you do not define a label, the aggregate expression, computed expression, or numeric field name displays.

`pattern` specifies the display pattern for the value of the aggregate expression, computed expression, or numeric field value.

`axis name` specifies a name for the numeric axis as you want it to appear in the chart.

`minRangeValue` specifies the minimum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

`maxRangeValue` specifies specifies the maximum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

`tickUnitValue` specifies the units that display on the numeric axis. For example, if you specify a minimum range of 0, a maximum range of 100, and a tick unit of 10, the numeric axis displays 0, 10, 20, 30, 40, and so on up to 100.

Note

Field names in expressions and expression labels with colons must be enclosed by escaped double quotes, and the whole computation must be enclosed in double quotes, for example,

--computations="\"Actual Dev Time\":\"Hours: Development\":pattern:..."

- `--startDateField=`*`field`*

specifies the date field containing the date you want to use as the start date for each issue in an issue fields trend chart.

- `--numberOfSteps=`*`value`*

specifies the trend chart's time span. If this option is specified, the chart's end date is determined by the specified step type multiplied by the specified number of steps.

Note

You cannot have more than 500 steps in a trend chart.

To specify an interval in the past, use a negative value.

- `--runDateIsEndDate`

specifies that the chart's run date is the end date. This option replaces the `--endDate=`*`value`* option.

- `--[no]deltasOnly`

specifies whether to display only the differences between the current and previous values of the reported numeric fields in an issue fields trend chart.

- `--issueIdentifier=`*`value`*

specifies the field that you want to identify issues by in an issue field or issue fields trend chart. For example, if you specify `--issueIdentifier={Project}`, each issue in the chart is identified by the value of the `Project` field.

If you want to add text that precedes the specified field, type it before the field, for example, `--issueIdentifier=Project:{Summary}`. The chart then identifies each issue by displaying `Project:` *Summary field value*.

- `--[no]displayShapesForLineGraphs`

specifies whether to display shapes in a line graph chart. The shapes in the chart represent data, allowing you to more easily differentiate the data in the chart.

- `--[no]swapRowsAndColumns`

specifies whether to invert the appearance of columns and rows in a table chart.

- `--[no]displayRowTotals`

specifies whether to display row totals in a table chart.

- `--[no]displayColumnTotals`

specifies whether to display column totals in a table chart.

- `--rangeDefinitions=`*`value`*

specifies range definitions for computed expressions included in a table chart, where `value` consists of the following attributes: *`expression name;range field name;range label:lower limit:upper limit:icon:background color:text color:text style:display format; lower limit:upper limit:.....;extend to axis`*.

`expression name` specifies the name of the computed expression that the range definition applies to. An expression name is mandatory and must be a valid expression in the chart. For column or row totals, valid expression names are `-Column Totals-` and `-Row Totals-`. For distribution charts containing multiple computed expressions, row or column totals must be followed by the expression name.

`range field name` specifies a valid field name if you want to relate the range definitions to an existing range field. For one chart range, specify an empty string as the range field name. If a valid range field name is defined for each range, define a range label, background color, text color text style, and display format. For individual range definitions, define a range label, lower limit, upper limit, icon, background color, text color text style and display format for each range.

`range label` specifies a label for the range.

`lower limit` specifies the lower limit of the range. If a lower limit is not specified, `-Infinity` is automatically specified.

upper limit specifies the upper limit of the range. If an upper limit is not specified, `Infinity` is automatically specified.

Note

A numeric value must be specified for a defined range; range intersections are invalid. For example, the following ranges are invalid: `0 - 5` and `4 - 8`, or `0 - 5` and `5 - 10`. For an integer field, an acceptable range would be `0 - 5` and `6 - 10`. For a floating point field, an acceptable range would be `0 - 5` and `5.01 - 10`.

`icon` specifies an image file representing the range category. This is optional.

`background color` specifies the background color of the range using the RGB color model, for example, *`'R,G,B'`*, where *`R`*, *`G`*, and *`B`* are within the range 0-255.

`text color` specifies the text color of the range using the RGB color model, for example, *`'R,G,B'`*, where *`R`*, *`G`*, and *`B`* are within the range 0-255.

`text style` specifies the text style. Available text styles are `plain`, `bold`, `italic`, `bolditalic`, or `defaultplain`.

`display format` specifies how to display the range in the table chart. Available options are `value`, `iconvalue`, `icon`, `label`, `iconlabel`, or `blank`.

`extendToAxis` specifies whether to apply the range definition associated with a computed expression to all computed expressions in the chart. This option can be `false` or `true`. By default, `false` is specified.

Note the following:

- You cannot specify a range for the `Count` expression.
- You can specify a range for each computed expression; however, only one computed expression can specify the *`extendToAxis`* option.
- If a table cell contains a display definition that conflicts with the *`extendToAxis`* option of another table cell, both table cells display the *`background color`* option of the table cell with the enabled *`extendToAxis`* option.

See Also

- Commands: im copychart, im deletechart, im editchart, im viewchart, im runchart
- Miscellaneous: options

im createcolumnset

creates a single column set

Synopsis

```
im createcolumnset [--fields=field,field,...] [--name=value] [--[no]sortAscending] [--sortField=field] [--hostname=value] [--
port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=
[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

Note

Column sets are no longer supported for the Integrity Client. This command can only be used to create column sets for the MKS Worktray used in integrations. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

Column sets are a grouping of Issue fields into columns for viewing. The issue field names are used as the column headings and are referred to as *column types*. The order of the column headings and the rows may be sorted by field name. Column sets are individually saved for each user. A user may not view, modify, or delete another user's column sets.

A default column set (named default) has already been created for you.

Note the following:

- You cannot delete or rename the default column set.
- You cannot give the column set the same name as another column set you have already created. You cannot give the column set the name `default`.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field,field,...`

  specifies the issue fields to be included in the column set. Your administrator defines the fields in an issue type. Use commas to specify more than one field.
- `--name=value`

  specifies the name of the column set to create.

  Note

  You *must* specify a column set name. Do not use square brackets in column set names.

- `--[no]sortAscending`

  specifies whether to sort the specified field in ascending or descending order.
- `--sortField=field`

  specifies the field to sort issues by.

See Also

- Commands: im copycolumnset, im deletecolumnset, im editcolumnset, im viewcolumnset
- Miscellaneous: options

im createcontent

creates new content and adds it to a segment

Synopsis

```
im createcontent [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--addAttachments=value] [--type=type] [--field=value] [--richContentField=value] [--insertLocation=[first|last|before:ID|after:ID|[0,...]]] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)][--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--parentID=value] [--customFieldDefinition=value] [--customFieldValue=value]
```

Description

`im createcontent` creates new content and adds it to a parent segment. You can insert the content into a specified location in the parent segment's structural relationship list. For example:

```
im createcontent --parentID=23 --insertLocation=first
```

creates an empty issue and inserts it at the beginning of the structural relationship list for parent segment 23.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--field=`*value*

  specifies a field and its value for the issue, where *value* is of the form "*fieldName*=*fieldValue*", for example, `--field=Severity=Critical`.

  If the field is multi-valued, *value* is of the form "*fieldName*=*fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add.

  To specify a workflow and document project, project names must be preceded by a (/), for example, `--field=Project=/testProject`.

  To specify a configuration management project, use the following syntax: `--field=`*fieldname*, `server=`*server*, `project=`*projectname*, [`devpath=`*devpath*|`revision=`*checkpointrevision*].

- `--richContentField=`*value*

  specifies a rich content field and its value for the issue, where *value* is of the form "*richcontentfieldname=fieldValue*".

  To specify a link to an item by ID: `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547">Part Requirement</a>`

  To specify a link to an item by ID and label: `<a href="mks:///item?itemid=`*ID*`&label=LABEL">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>`

  To specify a link to an item by ID and revision: `<a href="mks:///item?itemid=`*ID*`&revision=REVISION">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>`

  To specify a link to an item by ID and date: `<a href="mks:///item?itemid=`*ID*`&asof=DATETIME">`*DisplayText*`</a>` where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

  For example: `<a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>`

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "`b>Feature Overview/b>`". In the Windows command line interface (cmd.exe), the ^ escape character must precede the and > characters in HTML tags, for example, `^b^>Feature Overview^/b^>`. You may need to escape nested " characters, for example, `--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"`

- `--addRelationships=`*value*

  adds related issues to the issue, where *value* is of the form [`FieldName`]:`IssueID`[`relationshipFlags`][,...]. If no field name is specified, the `Forward Relationships` field is used.

Note

Adding a related issue is only permitted if your administrator has allowed relationships for the segment's issue type.

- `--addSourceTrace=`*value*

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addSourceLink=`*value*
- adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addAttachment=`*value*

  adds attachments to the issue, where *value* is of the form "*fieldName*,`path=`*pathToFile*[,`name=`*nameOfAttachment*][,`summary=`*shortDescription*] " .

Note

Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

- `--insertLocation=[first|last|before:ID|after:ID|[0,...]]`
- determines where the new content should go in the parent segment's structural relationship list. The options are as follows:

  first inserts the content at the beginning of the list

  last inserts the content at the end of the list

  before:<ID> inserts the content before the specified ID

  after:<ID> inserts the content after the specified ID

  [0,...] inserts the content at the specified location. If a negative is specified, the content is inserted at the beginning of the list. If the number specified is too large, the content is inserted at the end of the list.

- `--parentID=`*value*

  the ID of the parent segment that the issue will be added to. This option is required.

- `--type=`*type*

  specifies the type of issue. Your administrator defines issue types.

See Also

- Commands: im removecontent, im copycontent, im movecontent
- Miscellaneous: diagnostics, options, preferences

im createdashboard

creates new dashboard

Synopsis

```
im createdashboard [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--description=value] [--name=value] [--fieldFilterConstraint=field:[Open[:value,value,...] | [Fixed[:value,value,...] | [Restricted[:value,value,...[:value,value,...]]] | [None][--layout=value ] [--layoutFile=value] [--sharedAdmin] [--hostname=value] [--port=value] [--password=value] [--user=value][(-?|--usage)] [(-g|--gui)] [(-Fvalue|--selectionFile=value)] [--quiet][--settingsUI=[gui|default]][--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch][--cwd=value] [--forceConfirm=[yes|no]]
```

Description

You can define and store any number of dashboards using Integrity. A dashboard is a static, user-definable view comprised of any combination of the following components: charts, reports, images, labels, links to reports, links to queries, and URLs. For more information on dashboards, refer to the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `im`commands, as well as some general options. See the options reference page for descriptions.

- `--shareWith=u=`*`user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`*

  specifies the users and groups that can use and modify the dashboard. Your administrator defines users and groups.
- `--name=`*`value`*

  specifies the name of the new dashboard. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--description=`*`value`*

  specifies a short description for the new dashboard.
- `--fieldFilterConstraint=`*`field`*`:[Open[:`*`value,value,...`*`]|[Fixed[:`*`value,value,...`*`] | [Restricted[:`*`value,value,...[:value,value,...`*`]]]| [None]`

  specifies how field filters can be applied to the dashboard at runtime. The first component of the value is the field name. Currently, only project field filters are supported. The second component is the filter type.

  `Open` specifies that all projects can be selected as filter values when the dashboard is run. You can also specify default filter values to apply.

  `Fixed` specifies that when the dashboard is run it will be filtered by the specified values. You cannot change this filter at runtime.

  `Restricted` specifies that when the dashboard is run you can select any of the specified filter values. You can also specify default filter values to apply.

  `None` specifies that when the dashboard is run, no project filter displays.

  Note

  Depending on how you design your dashboard layout, the dashboard filter may not be applied to chart, report, report link or query link dashboard components. If this option is not specified, the`Open` filter is used.

- `--sharedAdmin`

  specifies the dashboard as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.
- `--layoutFile=`*`value`*

  specifies the file that contains the complete definition of the dashboard layout.
- `--layout=`*`value`*

  the XML representation of the dashboard layout. The layout must conform to a specified format. For more information, see the *PTC Integrity User Guide*. This setting is optional.

See Also

- Commands: im copydashboard, im deletedashboard, im editdashboard, im viewdashboard, im dashboards, im rundashboard
- Miscellaneous: options

im createdynamicgroup

creates a dynamic group and sets its properties

Synopsis

```
im createdynamicgroup [--membership=proj1=u=user1,user2,..:g=group1,group2,..;proj2=...] [--
projectmembership=project=inherit|nomembers|per-project-membership] [--image=[none|default|<path>] [--description=value] [--
name=value] [(-Ffile|--selectionFile=file)] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)]
[(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--quiet] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]]
[--status=[none|gui|default]]
```

Description

`im createdynamicgroup` creates a dynamic group to control which users can work with issues in a specific project for workflows and documents. Dynamic groups can be used when you need to limit access to issues for a specified project and ultimately control state transitions, field relevance, and field editability. For example:

```
im createdynamicgroup --name=ServicesGroup --membership=/ServicesProject=u=jriley
```

creates the *ServicesGroup* as a dynamic group, assigning the *ServicesProject* as the root project and *jriley* as a member.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--membership=proj1=u=user1,user2,..:g=group1,group2,..;proj2=...`

  assigns the membership for the dynamic group.

Note

If you are a project administrator, the `im dynamicgroups --fields=membership` command displays a subset of the projects in your Integrity configuration. If a super administrator uses that list of projects when specifying `im editdynamicgroup --membership`, the membership for the projects that the project administrator does not have permission to view are removed.

Note

Specifying the `u=user1,user2`, `g=group1,group2` option implies that the dynamic group is assigned to a root project. If the dynamic group is assigned to a child project and you want the parent project permissions to apply, do not specify these options.

---

where: *proj1*specifies the project you want the dynamic group to apply to, for example, the `/SourceCode` project. This command processes only the projects that the project administrator is assigned to.

To indicate that a project does not have any groups and users as members of the dynamic group, specify the `nomembers` keyword, for example, `--membership=/Project=nomembers`.

To inherit the membership from the parent project to the dynamic group, specify the `inherit` keyword, for example, `--membership=/Project=inherit`.

Note

The nomembers project is a top level project, and so cannot inherit permissions from any other project. Furthermore, no other project can be a subproject of the nomembers project.

`u=user1,user2`

specifies the users in the dynamic group.

`g=group1,group2`

specifies the groups in the dynamic group.

- `--image=[none|default|<path>]`

  specifies whether an image appears for the dynamic group.

  `--image=none` does not specify an image for the dynamic group.

  `--image=default` specifies the default image for the dynamic group.

  `--image=<path>` specifies the path and name of a custom image for the dynamic group, for example, `c:\images\dynamic_group_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=value`

  specifies a description of the dynamic group.

- `--name=value`

  specifies the name of the dynamic group. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.

- `--projectmembership=project=inherit|nomembers|per-project-membership`

  specifies the membership for a project that the project administrator is assigned to.

  where:

  project

  specifies the project you want to assign membership to.

  `inherit`

  specifies to inherit the membership from the parent project to the dynamic group.

  `nomembers`

  specifies that the project does not have any groups and users as members of the dynamic group.

  per-project-membership

  specifies the group and user membership for the project, where, *per-project-membership* is in the form `user-list|group-list|user-list:group-list`.

Note

Specifying users, but no groups removes any existing groups. Similarly, specifying groups, but no users removes any existing users.

See Also

- Commands: im deletedynamicgroup, im editdynamicgroup, im viewdynamicgroup, im dynamicgroups
- Miscellaneous: options

im createfield

creates a field

Synopsis

```
im createfield [--name=value] [--[no]confirmMultiValued] [--[no]multiValued] [--defaultBrowseQuery=[user:]query] [--description=value] [--position=<number>] [--testResult] [--type=
[integer|pick|float|logical|date|shorttext|longtext|user|group|relationship|siproject|range|phase|qbr|ibpl|fva|attachment|sourcelink|ier]] [--associatedField=field] [--loggingText=
[none|mostRecentFirst|mostRecentLast]] [--default=value] [--defaultColumns=value] [--displayAs=[default|checkbox]] [--displayTrueAs=value] [--displayFalseAs=value] [--min=value] [--max=value]
[--backedBy=value] [--backingFilter=value] [--backingStates=value] [--backingTextField=field] [--backingIBPLTextFormat=value] [--backingType=type] [--[no]sortIBPLDescending] [--
sortIBPLField=field] [--backingfilter=<querydefinition>] [--computation=value] [--[no]allowComputationUpdatesOnVersion] [--[no]staticComputation] [--storeToHistoryFrequency=
[never|daily|weekly|monthly|delta] [--addEntry=value] [--phases=text:state,state,...:image;...] [--[no]displayAsProgress] [--[no]displayAsLink] [--[no]trace] [--query=[user:]query] [--
correlation=src-field:dest-field,...] [--displayRows=value] [--displayStyle=value] [--ranges=text:lowerValue;upperValue:image,...] [--associatedField=fieldName] [--picks=text:value:image,...]
[--suggestions=text1,text2,text3,...] [--maxLength=value] [--displayPattern=value] [--displayName=value] [--relevanceRule=rule] [--relevanceRuleFile=filename] [--editabilityRule=rule] [--
editabilityRuleFile=filename] [--copyRelevanceRule=field] [--copyEditabilityRule=field] [--allowedTypes=type:type,type,...[;...]] [--
addLinkFlags=name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];...] [--[no]cycleDetection] [--[no]showTallRows] [--[no]richContent] [--[no]textIndex] [--
[no]substituteParams] [--defaultAttachmentField=field] [--showDateTime] [--relationshipBrowseStyle=browseStyle=[query|finder],queryOption=
[allQueries|specificQueries],specificQueriesList=query::query::..., finderQuery=[user:]query,outlineFormat=value,relationshipstoFollow=relationship::relationship::...] [--hostname=server] [--
password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] [--user=value] [--ierDefaultColumns=[server,title,name...]] [--incomingTraceProvider=value]
```

Description

`im createfield` creates a field for workflows and documents. For example:

```
im createfield --hostname=abcFinancial --user=jriley --type=integer\u00a0 --storeToHistoryFrequency=daily --staticcomputation  --computation='DaysInPhase("RFC Phase","Submission")' --name="Days_In_Phase"
```

creates the *Days_In_Phase* integer field. This field calculates how long an item spends in the *RFC Phase* and *Submission* phases, storing the new values on a daily basis.

Computed Fields

In addition to recording information in fields, Integrity can also perform calculations between fields, storing the result as a read-only value in a *computed field*. For example, in a Feature type, Integrity can add the `QA Estimated Time` and `Development Estimated Time` fields to determine the value of the `Total Estimated Time` field (the computed field). Previously, this could only be accomplished by creating an event trigger.

To create a computed field, you choose a field type to configure as a computed field. To specify a field type, use the `--type` option. Date, floating point, integer, short text, and logical field types can be configured as computed fields. However, only date, floating point, integer, long text, and short text fields can be used in the underlying expression that performs the calculation between fields. Logical fields can only be the result of an expression.

An expression in a computed field is similar to any expression you would find in a programming language. To specify an expression, use the `--computation=value` option. There are two types of expressions you can create:

- *intra-issue* expressions perform calculations between fields in a single issue, for example, adding the `QA Estimated Time` and `Development Estimated Time` fields to produce a value in the `Total Estimate Time` field.

  intra-issue expressions can also retrieve information from an issue using external information functions, such as `CPECount()`, which counts the number of change package entries in an issue. Using external information functions in expressions allow you to retrieve metrics about your workflow. Once you define an intra-issue expression using external information functions, you can use queries, charts, reports, and dashboards to collect the metric data and report on it. For example, you can create a query that returns issues with a specific number of change packages and use that query in a report, chart, or dashboard.
- Inter-issue or aggregation expressions use aggregation functions to perform calculations against fields in a list of issues. For example, using the `sum(field)` aggregation function in a report you could add the `Estimated Cost` field in a list of Project issues to produce a total estimated cost.

Creating computed fields to perform calculations between fields is not the only available method, nor is it restricted to administrators. Users can also perform these calculations by creating computed expressions in reports and charts, provided they understand the supported syntax, operators, functions, and operations. For example, using the appropriate report tags and syntax, a user can total the number of Defect issues that have not changed states for 0-5 days, 6-10 days, and 11+ days. For more information on creating reports, see the *Integrity Server Administration Help*. For more information on creating charts, see the *Integrity Help*.

For a complete list of acceptable syntax, operators, functions, and operations, see the *Integrity Server Administration Help*.

After you create the expression, you choose how often the computed field calculates the value and select a computation type by using the `--storeToHistoryFrequency=value` and `--[no]staticComputation` options.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirmMultiValued`

  specifies whether to confirm the setting of the `--[no]multiValued` option.
- `--[no]multiValued`

  specifies whether to allow users to select multiple pick list values at one time or link to multiple issues in a relationship field.

Note

Caution: Creating a multi-valued pick list permanently changes the storage format and cannot be reverted to single-value storage. However, you can still revert the field to a single-value pick list, if needed.

- `--defaultBrowseQuery=[user:]query`

  specifies the user and name of the default Admin query to use when adding a related item by browsing. This option only applies to relationship fields.
- `--name=value`

  specifies the name of the field to create. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--description=value`

  specifies a description of the field.
- `--position=<number>`specifies the position in the list of fields.
- `--testResult`

  specifies whether the field is used for test results only. For more information on using fields for test results, see the *PTC Integrity Server Administration Guide*.
- `--type=integer|pick|float|logical|date|shorttext|longtext|user|group|relationship|siproject|range|phase|qbr|ibpl|fva|attachment|sourcelink|ier`

  specifies the field data type. Fields are categories of data that can be associated with issues. This option is mandatory.
- `--type=integer`

  specifies simple, countable items, such as call tracking numbers.
- `--type=pick`

  specifies items that should display in a drop-down list, such as predefined product codes.
- `--type=float`

  specifies numbers with decimals, such as performance data.
- `--type=logical`

  specifies Boolean items (ones that are either true or false), such as whether an issue has been tested. Optionally, you can customize how true and false values display by specifying the `--displayTrueAs=value` and `--displayFalseAs=value` options. For example, the logical field can display yes or no values.
- `--type=date`

  specifies dates, such as when a defect was fixed. For example, Feb 25, 2007. Optionally, you can include the time by specifying the `--showDateTime` option.

Note

- Displayed date fields do not change based on the time zone in which a user is operating. However, displayed date/time fields and time entries vary based on the time zone in which a user is operating.
- Date/time fields cannot be converted to date fields.

- `--type=shorttext`

  specifies miscellaneous information, such as a comment field. This data type allows you to define suggestions.
- `--type=longtext`

  specifies miscellaneous information. This data type includes an option for displaying rows.
- `--type=user`

  specifies users that display in a drop-down list, such as users in a specific group.
- `--type=group`

  specifies groups that display in a drop-down list, such as groups assigned to a specific project.
- `--type=relationship`

  specifies links to other issues.
- `--type=qbr`

  specifies a named query that displays issues meeting the query criteria as a read-only relationships field. This field extends the concept of the relationship field by displaying a large number of related issues. For example, the `Features` field in a Project issue would display all the Feature issues that are returned by the `Release_5_Features` query.
- `--type=range`

  specifies categorized numeric value ranges in an associated numeric field (integer or floating point). A range field is a computed pick list field associated with a numeric field: range limits and the associated field are stored in the computation expression and the range category name and icon are stored in the database as pick list items. When a value is entered in the associated numeric field, the appropriate range category displays in the range field. Range fields are useful for defining thresholds for numeric data and providing a broad overview of that data. For example, if a Project type has an integer field called `Critical Defects` that displays the number of Defect issues marked `Critical`,, you could add a range field called `Defect Status` that displays one of the following range categories based on the number of issues in the `Critical Defects` field:

  `Golden`: "Critical Defects" = 0

  `Acceptable`: 1 <= "Critical Defects" <= 6

  `Watch`: 7 <= "Critical Defects" <= 19

  `Trouble`: 20 <= "Critical Defects"

Note

At this time, you can only create ranges using the <= operator.

- `--type=phase`

  specifies categorized groups of states in a workflow, essentially creating states (represented by phases) and sub-states (represented by states) for an issue type. A phase field is a computed pick list field associated with states. When a value is entered in the state field, the appropriate phase category displays in the phase field. Phases are useful for organizing an issue type's workflow if it contains a large number of states and provide users with a broad overview of an issue's status independent of the workflow. For example, a Feature type could have the following phases (states are in brackets):
  - `Requirements (Draft 1; First Draft Signoff;...)`
  - `Design (Update Data Model; ...)`
  - `Development (In Development; Unit Testing;...)`
  - `Testing (White Box Testing; Black Box Testing; Regression Testing;...)`
  - `Implementation (Planning; Implementation;...)`
  - `Post-Implementation Maintenance (Patch 1 In Progress;...)`
- `--type=siproject`

  allows users to specify a related configuration management project, optionally including a checkpoint revision or development path. By creating a configuration management project field and a computed field that uses the `SIMetric()` external information function, you can retrieve metrics about the specified project, for example, how many lines of code are in the project. For more information on creating configuration management project metrics, external information functions, and computed fields, see the *PTC Integrity Server Administration Guide*.

Note

- Metrics are only maintained against project checkpoints; therefore, to generate metrics, users must specify a checkpoint when they specify the configuration management project.
- To allow users to specify a related configuration management project when creating or editing an issue, the workflow and document management, and configuration management integration must be enabled and properly configured. For more information, refer to the *PTC Integrity Server Administration Guide*.

- `--type=ibpl`

  specifies an issue backed pick (IBPL) list field that allows you to share information in one or more issues with other issues. By creating an issue type that acts as a table of information (known as the backing issue type), then creating issues of that type with a non-computed short text field containing the information you want to share, an IBPL field in another issue type can display the values of the short text field as pick text. Whenever a short text field is updated in a backing issue, all issues with IBPL fields that reference the short text field value update to reflect the new value.

  For example, if you create a Department issue type with a `Manager` field, then create several Department issues to record the different departments and managers in your organization, `Manager` field values in all Department issues appear as pick text in the `Department Manager` IBPL field in the Defect issue type. If a Department issue's `Manager` field has a value of `Fred` and it changes to `Ted`, all Defect issues with a value of `Fred` in the `Department Manager` field automatically update to display `Ted`.
- `--type=fva`

  specifies a field value attribute (FVA) that allows you to share field information in an IBPL field's backing issue with other issues, displaying the field information as a value in the FVA field. An FVA data type is useful for maintaining field information in one issue and sharing the field information with other issues.

  For example, if a Defect issue has an IBPL field named `Department Manager` and a value of `Jim`, the FVA field named `Extension to Call` displays `x626` because the backing Department issue displays `Jim` in the `Manager` field and `x626` in the `Phone Extension` field (the backing field for the FVA field). If the `Department Manager` field value changes, the `Extension to Call` field value updates to reflect the new field value.

Note

- The name of the IBPL field cannot be the same as the referenced field in the backing issue type.
- You cannot create an FVA field backed by a sourcelink field, or a range field that is based on a numerical FVA field.
- If you create an FVA rich content field, it can only access attachments from an FVA attachment field over the same relationship as the FVA rich content field. This keeps the text and image data together. If the rich content field is not an FVA field, it should use non-FVA attachment fields. If you create a type with visible FVA and non-FVA rich content and attachment fields, Integrity only displays visible attachment fields visible to the user in the GUI or Web interface. Fom the CLI, the user must know which attachment fields to use with rich content fields.

- `--type=attachment`

  allows users to add attachments to items, such as design documents.
- `--type=sourcelink`

  specifies a source link field that allows you to create links to configuration management source files.

  sourcelink fields can either be created with trace enabled or disabled.

  For example, you could use a sourcelink field with `--trace` turned on (specifying `--trace` when creating the field), on a requirement item to track the changes made in order to satisfy that requirement. Viewing the sourcelink traces for a requirement helps you analyze the impact of changing that requirement.

  If specified with `--notrace` (trace is not enabled) the sourcelink field contains one way links between items and source files that don't get updated when the source changes.
- `--type=ier`

  specifies an incoming external reference (IER) field, which indicates that a requirement has a trace relationship to an object in a product lifecycle management (PLM) system. This option is only valid when enabled by functionality to create traces between Integrity and a PLM system.

  When specifying this option, you must also specify the `--incomingTraceProvider` option. Optionally, you can set the `--ierDefaultColumns` option.
- `--default=`*`value`*

  specifies the default value for the field.

Note

- To include the time with date fields, specify the `--showDateTime` option and use the *`mm/dd/yyyy hh/mm/ss`* format, where *`hh/mm/ss`* is the hour, minutes, and seconds. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format. However, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. For a date field, you can specify the current date and a time of 00:00:00 (midnight) when the issue is submitted by typing *`today`*. For a date/time field, you can specify the current date and time when the issue is submitted by typing *`today`*. To specify an empty value for a date field, type " ".
- Integer fields allow a maximum of nine digits and floating point fields allow a maximum of 15 digits.
- When editing a floating point field to set the default, minimum, or maximum value, you can enter a negative exponent using the E number notation, for example, -123.1E-3.

- `--defaultColumns=`*`field1,field2,...,mks:virtualField1,mks:virtualField2...`*

  specifies the default ColumnSet for the field. This option is only valid for the following field data types: Relationship and Query Backed Relationship. This option sets default columns for the relationship field pair.
- `--displayAs=[`*`default|checkbox`*`]`

  specifies an alternate display for a logical field. By default, a logical field allows users to select true or false values. However, you can choose to display the field as a checkbox (a checkbox that is checked indicates a true value and a clear checkbox indicates a false value). To display a logical field as a checkbox, specify `--displayAs=checkbox`. To display a logical field with true or false values, specify `--displayAs=default` or do not specify the option. This option is only valid with the `--type=logical` option.
- `--displayTrueAs=`*`value`*

  specifies the custom value that represents how the true value displays in a logical field, for example, `--displayTrueAs=yes`. The maximum number of characters is 100 and this option is only valid with the `--type=logical` and `--displayFalseAs=`*`value`*options.

Note

Important: Specifying custom values for logical fields impacts existing custom scripts that use the default true and false values. PTC recommends reviewing the scripts and making necessary modifications to reflect new custom values.

- `--displayFalseAs=`*`value`*

  specifies the custom value that represents how the false value displays in a logical field, for example, `--displayFalseAs=no`. The maximum number of characters is 100 and this option is only valid with the `--type=logical` and `--displayTrueAs=`*`value`* options.
- `--min=`*`value`*

  specifies the minimum value for the integer, float, or date type field. See the `--default==`*`value`* option for rules.
- `--max=`*`value`*

  specifies the maximum value for the integer, float, or date type field. See the `--default=`*`value`* option for rules.

  For long text fields, the maximum number of characters is 4,000.
- `--backedBy=`*`value`*

  specifies the field in the related issue type whose value you want to display in the FVA field, where value uses the format *`relationship-name.field-name`*.

  `relationship-name` is the field that contains the backing issue for the FVA field. This field can be any of the following:

  - A relationship field.
  - An issue backed pick list field.
  - The project field, if backing projects are enabled for the issue type.

  `field-name` is the field in the backing issue type whose value you want to display in the FVA field.

  This option is used with the `--type=fva` option.

Note

- The relationship field must be single valued.
- The referenced field must be visible in the backing issue.
- You can set display options such as `--displayPattern=`*`value`* and `--[no]displayAsProgress`, if the field data type allows them.

- `--backingFilter=`*`value`*

  specifies a query as the query definition filter to populate the issue backed pick list with.
- `--backingStates=`*`value`*

  specifies active states to populate the issue backed pick list with. Populating an issue backed pick list with backing issues in specific states essentially allows you to "deactivate" entries in an IBPL. For example, if an Employee issue contains an `Active` and `Inactive` state, and you select `Active` as the active states, only Employee issues in a state of `Active` display pick text entries in the IBPL. This option is used with the `--type=ibpl` option.
- `--backingTextField=`*`field`*

  specifies the non-computed short text field in the backing issue type that you want to use as pick text. For example, if you select the `Manager` field and `James Riley`, `Sherry Robertson`, and `Dan Evans` are field values in some of the backing issues, those names display as pick text in the issue backed pick list field. This option is used with the `--type=ibpl` option.
- `--backingIBPLTextFormat=`*`value`*

  specifies multiple fields in the backing issue type that you want to link together to use as pick text. The format is similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). For example:

  `--backingIBPLTextFormat="{ID}:{Summary}"`

Note

This option overrides any value specified by the `--backingTextField` option.

- `--backingType=`*`type`*

  specifies the issue type containing the short text field that you want to reference. This option is used with the `--type=ibpl` option.
- `--[no]sortIBPLDescending`

  specifies the sort order of the field on the backing item type in the IBPL. `--[no]sortIBPLDescending` sorts the field in ascending order. `--sortIBPLDescending` sorts the field in descending order. This option is used with the `--type=ibpl` and `--sortIBPLField=`*`field`* options.
- `--sortIBPLField=`*`field`*

  specifies a visible field on the backing item type to sort by in the IBPL. For example, if an IBPL field uses a `Build` item as the backing item type, you could display the most recent build from the `Build Number` field at the top of the IBPL field. If this option is not specified, field values in the IBPL are sorted alphanumerically. This option is used with the `--type=ibpl` and `--[no]sortIBPLDescending` options.

Note

- The following field types on the backing item type cannot be sorted: type, attachment, IBPL, FVA, QBR, range, relationship, source link, and SI project fields.
- If a field is currently specified as the sorting field and you choose to configure that field as invisible in the item type, a warning message notifies you that this will affect the sort order, allowing you to save or cancel the change. Saving your changes causes field values in the IBPL to be sorted alphanumerically.

- `--backingfilter=<`*`querydefinition`*`>`

  specifies a string to define the picklist value constraints. This option is used with the `--type=ibpl` option. The `<querydefinition>` must be of the same format as the query definition for a query. For details of the query definition format, see the reference page for the `--im createquery` command.

Note

The constraints defined for this filter are applied in addition to the filtering specified by the `--backingType` and the `--backingStates` options.

- `--computation=value`

  specifies a computed expression. For a complete list of acceptable syntax, operators, functions, and operations, see the *Integrity Server Administration Help*.

Note

A computed field displays 50 characters. However, it can contain a maximum of 200 characters.

The following are examples of computed field expressions:

- To determine how many days a Defect issue has not been worked on, create a computed field called `Days Inactive` and type the following expression:

  `now() - "Modified Date"`

  The `Days Inactive` field displays the number of days elapsed since the last edit of the Defect issue.
- If the Project issue type contains `Actual Cost` and `Expected Cost` fields, and you want to determine if and how much the actual cost of a project exceeds the expected cost, create a computed field called `Cost Overrun` and type the following expression:

  `"Actual Cost" - "Expected Cost"`

  If the value of the `Actual Cost` field exceeds the value of the `Expected Cost` field, the `Cost Overrun` field appears after editing the issue, displaying how much the actual cost overran the expected cost.
- To build upon the previous example, if you wanted to determine whether the actual cost of a project exceeded 90% of what you budgeted for, type the following expression:

  `"Actual Cost" > .90 * "Expected Cost"`

  If the value of the `Actual Cost` field exceeds the value of the `Expected Cost` field by 90%, the `Cost Overrun` field appears after editing the issue, displaying the percentage that the actual cost overran the expected cost.

  As project manager, you could closely monitor project budgets by creating a query or email notification that identifies Project issues where the cost has exceeded 90% of the estimated budget.

- `--[no]allowComputationUpdatesOnVersion`

  specifies to record the computation value at the time of versioning and prevent further updates. By default, the computation value in versioned items continues to update based on the computed field definition. If you specify `--[no]allowComputationUpdatesOnVersion` on a non-computed field, an error message displays. If you specify `--allowComputationUpdatesOnVersion` on a non-computed field, the option is silently ignored.
- `--[no]staticComputation`

  specifies the computation type. `--staticComputation` performs the computation and stores it in the issue's history based on the value specified in `--storeToHistoryFrequency=value`. PTC recommends a static computation if your expression involves expensive external functions, such as query or aggregate functions. `--[no]staticComputation` performs the computation every time field values used in the expression change. `--[no]staticComputation` is a dynamic computation and is selected by default.

Note

Because dynamically computed fields are not stored in the database, dynamically computed short text fields cannot be located with an all text field search in the Integrity Client. To search for dynamically computed short text fields, create a query that includes a specific "field" comparision. If the query does not include additional filters, the query may not return optimal results.

- `--storeToHistoryFrequency=never|daily|weekly|monthly|delta`

  indicates how often the computed field should be calculated and stored in the issue's history. Acceptable values are *daily*, *weekly*, *monthly*, *delta*, or *never*. *delta* specifies to store the computed field's value to the item history only when a delta is detected. To specify a custom frequency, specify *delta* and create an event trigger that specifies the desired frequency. Selecting a frequency is useful for historical charting. *delta* is specified by default.

Note

Using the `im analytics --recomputeHistory` command, you can calculate a computed field within a specific time frame, storing the value in the issue history. This command is useful for historical charting and reporting, allowing you to calculate, then compare the stored historical values of the computed field between current and past projects.

- `--addEntry=value`

  adds a phase or range field value.

  For phases, *value* is specified as *text:state,state,...:image;...*, where *text* specifies the phase name, *state* specifies an included state, *image* specifies the file path and the name of a custom image, or *none*.

  Note the following about adding phases:

  - There is no minimum or maximum amount of phases you can create for a phase field.
  - States not grouped into a phase are referred to as out of phase states. When an issue is in an out of phase state, the phase field displays `Out of Phase`.
  - Images for phases must be GIF or JPEG format, and no larger than 16 by 24 pixels.
  - No two phases can use the same state.
  - You cannot edit the `Out of Phase` phase. This phase identifies the states that are not included in any user defined phase.

  For ranges, *value* is specified as *label:lowerValue;upperValue,...*, where *label* specifies the range category name, *lowerValue* specifies the lower range, *upperValue* specifies the upper range. For example, `--addEntry=Golden:-Infinity;0,Acceptable:1;6,Watch:7;19,Trouble:20;+Infinity`. Range category names can be 100 characters long.

  Note the following about adding range limits:

  - You cannot specify different range category names and icons for types. However, you can specify different range limits for types.
  - Range field values are automatically determined based on an associated numeric field. Range fields cannot be edited in an Issue Details view.
  - Range value limits can be overridden on a per type basis. However, range category names and icons cannot be overridden.
  - If *lowerValue* is not set, `-Infinity` is automatically entered. If *upperValue* is not set, `+Infinity` is automatically entered.
  - A numeric value must be contained in one defined range; range intersections are invalid. For example, the following ranges are invalid: 0;5 and 4;8, or 0;5 and 5;10. For an integer field, an acceptable range would be 0;5 and 6;10. For a floating point field, an acceptable range would be 0;5 and 5.01;10.
  - If a value is entered in the associated numeric field that is beyond the set range values, the range field displays `Out of Range` in the issue. If no value is entered in the associated numeric field, the range field is empty.
- `--phases=text:state,state,...:image;...`

  specifies the phase name and the included states, where `image` can be none, or the file path and the name of a custom image. This option is used with the `--type=phase` option.

  Note the following about creating phases:

  - States not grouped into a phase are referred to as out of phase states. When an issue is in an out of phase state, the phase field displays `Out of Phase`.
  - You cannot edit the `Out of Phase` phase.
  - No two phases can use the same state.
  - Images for phases must be GIF or JPEG format, and no larger than 16 by 24 pixels.
  - There is no minimum or maximum amount of phases you can create for a phase field.
- `--ranges=text:lowerValue;upperValue:image,...`

  specifies the name of the range category and the lower and upper range limits, for example, `--ranges=Golden:-Infinity;0,Acceptable:1;6,Watch:7;19,Trouble:20;+Infinity`. Range category names can be 100 characters long. This option is used with the `--type=range` option.

  Note the following about setting range limits:

  - Range field values are automatically determined based on an associated numeric field. Range fields cannot be edited in an Issue Details view.
  - Range value limits can be overridden on a per type basis. However, range category names and icons cannot be overridden.
  - If *lowerValue* is not set, `-Infinity` is automatically entered. If *upperValue* is not set, `+Infinity` is automatically entered.
  - A numeric value must be contained in one defined range; range intersections are invalid. For example, the following ranges are invalid: 0;5 and 4;8, or 0;5 and 5;10. For an integer field, an acceptable range would be 0;5 and 6;10. For a floating point field, an acceptable range would be 0;5 and 5.01;10.
  - If a value is entered in the associated numeric field that is beyond the set range values, the range field displays `Out of Range` in the issue. If no value is entered in the associated numeric field, the range field is empty.
- `--associatedField=fieldName`

  specifies the numeric field to associate with the range field. This option is used with the `--type` option.

Note

Range fields cannot contain an associated field that includes a computed expression with an external information function.

- `--[no]displayAsProgress`

  displays the integer value as progress bar in the GUI.
- `--[no]displayAsLink`

  displays the selected value in the item backed pick list as a hyperlink to the backing item. The hyperlink displays in the GUI and the Web UI.
- `--[no]trace`

  If `--trace` is used with `--type=relationship`, sets the field as a trace relationship. Trace relationships are defined via field pairs and are presented to the user in domain-specific language, for example, Test and Requirements. To learn more about trace relationships, see the *PTC Integrity User Guide*.

  If `--trace` is used with `--type=sourcelink`, sets the field as a source link field with trace enabled (stores a source trace). This can only be specified when creating the field, and cannot be changed using `im editfield`.

  If `--notrace` is used with `--type=sourcelink`, or `--type=sourcelink` is specified without `--trace`, sets the field as a source link field. This can only be specified when creating the field, and cannot be changed using `im editfield`.
- `--query=[user:]query`

  specifies an administrator query to use as the backing query for a query backed relationship field.
- `--correlation=src-field:dest-field,...`

  specifies a pair of fields to correlate between the type containing the query backed relationship field and the issues returned by the query. For example, if you create a query backed relationship field called `Defects` for the Feature type and specify `Project` as the source field and `Project` as the target field, the `Defects` field displays all Defect issues that have the same project as the one specified in the Feature type's `Project` field. This option is not mandatory. However, if it is not specified, the list of relationships returned does not change with different issues.
- `--displayStyle=value`

  specifies whether the attachment, relationship, sourcelink, or query backed relationship field displays in table format or in a comma separated values (CSV) format.

  If you specify `-g` or `--gui`, the table format allows you to sort the issues and manipulate the columns that display in the table. For attachment, relationship and query backed relationship fields, the CSV format only displays the IDs of the issues. For sourcelink fields, the CSV format only displays the source file name, revision number, server and project.
- `--picks=text:value:image,...`

  specifies the list of valid active values for a pick list field, where image can be `none`, or the file path and the name of a custom image. Pick text must be 100 characters or less in length and empty values are not supported.

Note

This setting requires the complete list of active pick list values to be specified. Unspecified pick list values are inactive. Do not specify duplicate pick field values. Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--suggestions=text1,text2,text3,...`

specifies a list of suggested values for a short text field.

Note

This setting requires the complete list of suggested values to be specified.

- `--maxLength=value`

specifies the maximum character length value for a short text, long text, or rich content field. The maximum character length is dependant on your database type and setup.

- `--displayName=value`

specifies the name assigned as the display name of the field.

- `--displayRows=value`

specifies the number of rows to display for a long text, rich content, sourcelink, or relationship field. There is a maximum of 80 rows permitted for a long text, sourcelink, or relationship field, and 15 for a rich content field.

- `--displayPattern=value`

assigns a format to floating point or integer field values, known as a display pattern. Display patterns allow you to quantify numeric field values, for example, as currency or percentages. You can define a display pattern by combining a currency symbol, text that represents a measurement, and/or one or more of the following characters:

  - 0 - Displays as a zero in the output. For example, a display pattern of `000.00` displays an input value of `12.14` as `012.14` in the numeric field.
  - # - Displays as a digit in the output. If the digit is a zero and it is leading or trailing the input value, it is left out of the value displayed in the numeric field. For example, a display pattern of `#0.00` displays an input value of `0.126` as `0.13` in the numeric field.
  - .- Locale specific decimal separator.
  - - - Minus sign.
  - ,- Locale specific grouping separator.
  - E - Scientific notation, in the format `aEb`, where `a` is any real number, and `b` is the exponent.
  - ; - Separates positive and negative patterns.
  - % - Multiplies by 100 and displays as a percentage.
  - ' - Escapes special characters. Use '' to create a single quote.

  For example, if you specified a display pattern of `$#,###` and a user types `12345.123` in the associated numeric field, the numeric field displays `$12,345`. Similarly, if you specified a display pattern of `# minutes` and a user types `123` in the associated numeric field, the numeric field displays `123 minutes`

Note

- If the display pattern is invalid or the field type is not an integer or floating point, an error message displays. If no display pattern is specified, the field displays the value in a localized form.
- Display patterns appear only when viewing one or more issues. Integrity stores the field value as an unformatted numeric value in the database.
- By default, a floating-point field value displays the same number of decimal places as when the value was entered. Previously, floating point values rounded to three decimal places.
- Display patterns are applied to numeric values only when shown in the context of an issue. This means that query filters, rules, and trigger assignments display the unformatted, localized version of the numeric field value.

- `--relevanceRule=rule`

specifies the rules that determine when users see the field. For the rule syntax, see Specifying Rules on the options reference page.

Note

- Relevance rules are evaluated on the Integrity Client's time zone.
- SI project and attachment fields cannot be specified.

- `--relevanceRuleFile=filename`

specifies a file that contains the field relevance rules. See `--relevanceRule` for notes on the option and obtaining rule syntx for the file format.

- `--editabilityRule=rule`

specifies the rules for when users are permitted to edit the field. For the rule format, see Specifying Rules on the options reference page.

To prevent the field from being edited, specify `--editabilityRule="(false)"`. This option is useful for fields that are updated by event triggers and are not meant to be edited by users. For example, you could create a date field where the date is automatically specified when the issue enters a certain state.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date or date/time field, type *today*. To specify an empty value for the date field, type *none*.
- When specifying a user, you can choose yourself by specifying "`me`". "`me`" is a symbolic user which refers to the currently logged in user. For example, you could create an editability rule that specifies the `Requirements` field can be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
- Editability rules are evaluated on the Integrity Client's time zone.
- By default, `--editabilityRule="(false)"` is specified for read-only custom fields (i.e. phase, range, computed) and cannot be unspecified.
- SI project and attachment fields cannot be specified.
- Editability rules cannot include a computed expression that requires an external information function.

- `--editabilityRuleFile=filename`

specifies a file that contains the field editability rules. For the file format, see Specifying Rules on the options reference page.

- `--copyRelevanceRule=field`

copies the relevance rules of an existing field to the one being created.

Note

A false relevance rule (`--relevanceRule="(false)"`) can be copied in the CLI. However, you cannot do this in the GUI.

- `--copyEditabilityRule=field`

copies the editability rules of an existing field to the one being created.

Note

A false editability rule (`--editabilityRule="(false)"`) can be copied in the CLI. However, you cannot do this in the GUI.

- `--allowedTypes=type:type,type,...[;...]`

specifies the types of issues that can be linked using the relationship field. Specify the issue type that will use the relationship field, then list the issue types that can be linked to using the reverse relationship field. For example, for a one-way relationship showing the relationship between documentation issues and bugs, specify *Docs:Bugs* for the forward field, and *Bugs:Docs* for the reverse field.

For a two-way relationship, you need to specify allowed types for both sides of the relationship. For example, to allow the field to be used to create relationships between documentation issues and bugs, specify *Docs:Bugs;Bugs:Docs*.

- `--addLinkFlags=name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];...`

defines the relationship flags that can be added to relationships in the relationship field.

- `--[no]cycleDetection`

specifies whether or not the system will prevent relationship loops from occurring in the relationship field. A relationship loop occurs when an issue has both a backward and forward relationship through an issue (or through multiple issues). For more information on relationship loops, see the *PTC Integrity User Guide*.

- `--[no]richContent`

specifies whether to configure the long text field as a rich content field. Rich content enhances the display of text in long text fields by adding formatted text, tables, background colors, images, and hyperlinks. This option is enabled by default and does not support logging text fields.

Note

Caution: You can convert rich content fields back to long text fields. However, any existing rich content is displayed as HTML tags and attributes.

Because rich content is expressed using a limited set of HTML elements and attributes, you can define screen and printer Cascading Style Sheets (CSS) that ensure a consistent look when viewing and printing rich content field data in different Web browsers. For more information, see the *PTC Integrity Server Administration Guide*.

- `--[no]substituteParams`

specifies whether parameter references in this text field are replaced with parameter values when you view the item through a view or report that supports parameter substitution. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.

- `--defaultAttachmentField=field`

specifies the default attachment field that images are retrieved from when inserting images into a rich content field via attachment field. The default is the default `Attachment` field

- `--showDateTime`

specifies to include the time with the date in a date field.

Note

Important: Once you include the time and save the date field, you cannot change the date field to display the date only.

- `--relationshipBrowseStyle=browseStyle=[query|finder],queryOption=[allQueries|specificQueries],specificQueriesList=query::query::..., finderQuery=[user:]query,outlineFormat=value,relationshipstoFollow=relationship::relationship::...`

specifies the relationship browse style that is used for adding a related item. `browseStyle` can be either set to `query` or `finder`.

After you set the `browseStyle` to `query` or `finder`, the following options can be selected:

  - `queryOption` is applicable for `browseStyle=query` and can be specified as either "allQueries" or "specificQueries". When the `browseStyle=query` is specified, the `queryOption=[allQueries|specificQueries]` is mandatory.
  - `specificQueriesList` is applicable for `queryOption=specificQueries`. This option specifies the list of the names of Admin queries, separated by double colons(for example, specificQueriesList="query1::query2::query3).
  - `finderQuery` is applicable for `browseStyle=finder`. This option specifies the name of the Admin query to be used for the content of the `Find Items to relate` window left pane.
  - `outlineFormat` is applicable for `browseStyle=finder`. This option specifies the outline for the right pane in the `Find Items to relate` window.
  - `relationshipsToFollow` is applicable for `browseStyle=finder`. This option specifies the list of relationships, separated by double colons(for example, relationshipsToFollow="relationship1::relationship2::...). In the `Find Items to relate` window, the right pane is populated with the list of related items derived from the selected items in the left pane. The `relationshipsToFollow` defines the related items with which to populate the right pane.

    For example:

    - When the `browseStyle` is set to `query`:

```
im createfield --name=queryOptionAsSpecificQueriesMentioned --type=relationship --
relationshipBrowseStyle=browseStyle=query,queryOption=specificQueries,specificQueriesList="query1::query2::query3"
```

- When the `browseStyle` is set to `finder`:

```
im createfield --name=finderWithAllOptions --type=relationship --
relationshipBrowseStyle=browseStyle=finder,finderQuery="query1",outlineFormat="value",relationshipsToFollow="relationship1::relationship2"
```

- `--[no]textIndex`

  specifies whether queries against the field will be treated as word searches.
- `--[no]showTallRows`

  specifies whether the relationship field should display variable height rows.
- `--loggingText=[none|mostRecentFirst|mostRecentLast]`specifies logging text field mode, including the order the entries are displayed in. Logging text field mode is only a valid option for longtext fields.
- `--ierDefaultColumns=[server,title,name...]`

  specifies the columns that appear for trace relationships. This option is only valid for a `type` of `ier`.

  Available values are:

  - `server`—Server where the object resides
  - `title`—Summary of the object
  - `name`—Name of the trace
  - `reversename`—Trace type, such as "Satisfied By"
  - `suspect`—Indicates whether the trace is suspect
  - `icon`—Icon for this object type
  - `largePreview`—Large preview of the object
  - `smallPreview`—Small preview of the object
  - `URI`—Uniform resource identifier of the object
  - `localURI`—Uniform resource identifier of the requirement
- `--incomingTraceProvider=value`

  The incoming trace provider that is used to retrieve external references. If you are using ThingWorx, this is the name of the ThingWorx thing that returns IER values. This option is only valid for a `type` of `ier`.

See Also

- Commands: im editfield, im viewfield,im fields, im analytics
- Miscellaneous: options

im creategroup

creates a group

Synopsis

```
im creategroup [--name=value] [--description=value] [--image=[none|default|<path>] [--email=value] [--notificationRule=rule] [--queryTimeout=value] [--sessionLimit=value] [--notificationRuleFile=value] [--copyNotification=[user|group]:<name>] [--[no]active] [--[no]allowNonRealm] [-Ffile|--selectionFile=file] [--name=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [-?|--usage] [-N|--no] [-Y|--yes] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im creategroup` creates a group for workflows and documents. By default, the group must exist in the realm. You can create groups in Integrity by importing groups from a user authentication realm. For example:

```
im creategroup --name=Services --description=Provides_services_to_customers --image=/admin/creategroup/services.jpg
```

creates the *Services* group with a description and custom image.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the name of the group to create. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.

Note

PTC recommends that you do not use commas, colons, or special characters in group names. If you use commas in the group name, you cannot select the group in a multi-valued user field. If you use colons in the group name, the group cannot edit shared system provided objects (charts, queries, reports, and dashboards).

- `--description=value`

  specifies a description of the group.
- `--image=[none|default|<path>]`

  specifies whether an image appears for the group.

  `--image=none` does not specify an image for the group.

  `--image=default` specifies the default "group" image for the group.

  `--image=<path>` specifies the path and name of a custom image for the group, for example, `c:\images\group_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--email=value`

  specifies the email address of the group.
- `--notificationRule=rule`

  specifies the notification rule for this principal. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date field, type *today*. This option can be specified for a date field or a date field configured to display the date and time. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
- E-mail notification is subject to project, type, and field visibility rules. Only users that have visibility for a given project and type receive e-mail notification for issues related to that project and type. In addition, e-mail notifications include only the fields they have permission to view.

- `--notificationRuleFile=value`

  specifies the file that contains the rules to read. For the file content format, see Specifying Rules on the options reference page.
- `--queryTimeout=value`

  specifies the total time in seconds allowed for queries performed by members of the specified group.
- `--sessionLimit=value`

  specifies the number of sessions each user in the group can open at one time. This option assists in managing resource allocation for the Integrity Server. For example, if you specify 5, a user can begin a session in the Web interface, and open 5 tabs in a browser to run a chart in each tab. You can specify a maximum of 100 connections. Default is 5; however, a value of 0 specifies the default value.

Note

If there are Web interface users with heartbeat and idle timeout, then an idle-timed-out user still continues to exist, and is not subject to this limit.

- `--copyNotification=[user|group]:<name>`

  copies the given user/group's notification rules. This command copies the given user/group's notification rule into this group's at the time that the command is run. The command does not create a pointer to another notification rule.
- `--[no]active`

  Specifies if the group is actively used in Integrity. This option is useful for removing groups that are no longer needed. Specifying `--noactive` prevents the group from being displayed in group drop-down lists; however, the group still appears in existing issue data.
- `--[no]allowNonRealm`

  confirms the non-realm allowance. By default, you cannot import a group into the Integrity database unless that group exists in the current realm. This option allows you to override the requirement that the group needs to be in the realm. `--noallowNonRealm` is specified by default.

See Also

- Commands: im editgroup, im viewgroup, im deletegroup, im groups
- Miscellaneous: options

im createissue

creates a new Integrity issue

Synopsis

```
im createissue [--[no]showWorkflow] [--addAttachment=value] [--addFieldValues=value] [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--type=type] [--customFieldDefinition=value] [--customFieldValue=value] [--field=value] [--richContentField=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

You create an issue by providing issue data. You can also link the issue you are creating to another issue, or add an attachment to it.

For example,

```
im createissue --type=Docs --field="Type=Docs" --field="Summary=Update release notes" --description="Update content of release notes" --field="Assigned User=jriley" --field="Project=/Release 2.0"
```

creates a new Docs issue with the specified field values.

Note the following:

- Your administrator can configure long text fields to support rich content. *Rich content* enhances the display of text in long text fields by adding formatted text, tables, background colors, images, and hyperlinks. From the CLI, rich content is applied using a limited set of HTML elements and attributes. For a complete list of supported elements and attributes, see the *PTC Integrity User Guide*.
- The types of issues that you can create depend on the types created by your administrator. If there are no issue types to select, contact your administrator.
- Text fields, such as Summary, support HTTP hyperlinks. This is useful if you want to include a link to a document on an internal or external Web site, such as a design document. You must include the `http://` prefix.
- If you can specify an issue type but not create it, contact your Integrity administrator. The issue may contain a custom field defined to be both mandatory and invisible for this issue type.
- If your administrator has set up electronic signatures, you may need to provide your user name and password when creating an issue.
- Your administrator may include the time in date fields. You can specify the time when you select a date from the calendar. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. If you do not specify a time, the current time displays in the date field.
- To retrieve metrics from a configuration management project related to the issue you are creating, your administrator may define a field that accepts a configuration management project as a value. Optionally, you can specify a checkpoint revision or development path. If you specify a configuration management project and a checkpoint, then save the issue, one or both of the following may occur when you view the issue in the GUI or Web interface:
- One or more computed expressions in the issue calculate specific metrics about the project, displaying the results as a read-only value in a computed field (the visibility of the computed field depends on the field's relevance rules). For example, once you specify a project for the `Source Code` field, a `Lines of Code` field could calculate and display the number of lines of code in that project. As lines of code are added or removed from the project, the `Lines of Code` field updates to display the new value.
- A `metrics` hyperlink displays in the configuration management project field. Clicking the hyperlink displays various configuration management metrics about the project.

  In addition, the server and project information display in the configuration management project field as a hyperlink. Clicking on the hyperlink displays the project in a Project view.

  To select a configuration management project, you require the `OpenProject` permission for the specified project. Once a configuration management project has been specified, metrics can be obtained by any user with permissions to view the configuration management project field. For more information on selecting configuration management projects and viewing configuration management metrics, refer to the *PTC Integrity User Guide*. For more information on creating configuration management metrics, refer to the *PTC Integrity Server Administration Guide*.

  Important:

  Metrics are only maintained against project checkpoints; therefore, to generate metrics, you must specify a checkpoint when you specify the configuration management project.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields vary based on the time zone that a user is operating in.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- Your administrator determines the types of issues that can have attachments. Attachment fields can display under Fields or Attachments, and can display in comma separated values or table display format, depending on how your administrator has defined the attachment field. This procedure is based on the following configuration: an attachment field in table display format under Attachments.
- By default, Integrity allows file attachments to a maximum size of 4 MB and a maximum of 255 characters for file names. Your administrator may define a higher or lower limit depending on the requirements of your system. You can also attach more than one file to a single issue.
- Relationship fields can display under Fields or Relationships, and can display in comma separated values or table display format, depending on how your administrator has defined the relationship field. This procedure is based on the following configuration: a relationship field in table display format under Relationships.
- Adding a related issue is only permitted if your Integrity administrator has allowed relationships for the issue type.
- To quantify numeric field values, your administrator may define display patterns for numeric fields, for example, as currency or percentages. In addition, a display pattern may format the value you initially enter in the numeric field, for example, an input value of `0.126` may display as `0.13` after you create the issue. For more information on display patterns, contact your administrator.
- If a floating point field does not contain a display pattern, the field in the created issue displays the same number of decimal places as when the value was typed in the numeric field.
- Integer fields allow a maximum of nine digits and floating point fields allow a maximum of 15 digits. Your administrator can define default, minimum, and maximum values.
- Setting a user, group or project field to an inactive value results in an error message.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showWorkflow`

  specifies whether to display the workflow for the issue type, if your administrator has enabled it. This option can only be specified with `-g` or `--gui`. Viewing the Workflow panel is useful for determining where you can progress in the workflow. The Workflow panel displays the complete workflow for the issue type, unvisited states, visited states, the current state, other state transitions, and phases, as indicated by the Legend.
- `--addAttachment=value`

  adds attachments, where `value` is of the form *"field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription] "* .

  Note

  The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

  For example,

  ```
  addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
  ```

  adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

  If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. An issue cannot have more than one attachment with the same name.

  Note

  Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

`--addFieldValues=value`

specifies new field values to add, where *value* is of the form "*fieldName=fieldValue*."

Note

This option is only valid for the following multi-valued field types: pick, user, group, item backed picklist (IBPL), and relationship

---

`--addRelationships=value`

adds related issues, where *value* is of the form [*FieldName*]:`IssueID`[`relationshipFlags`][,...]. If no field name is specified, the `Forward Relationships` field is used. Adding a related issue is only permitted if your administrator has allowed relationships for the issue type.

Note

This option is deprecated. Instead, use the `--addFieldValues=`*value*option.

`--addSourceTrace=value`

adds a trace to a source file, where `value` is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName] [,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

Note the following:

- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

`--addSourceLink=value`

adds a link to a source file, where `value` is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*".

Note the following:

- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

`--type=type`

specifies the issue type to create. Your administrator defines issue types. This option is mandatory.

`--field=value`

specifies a field and its value for the new issue, where `value` is of the form "`fieldName=fieldValue`", for example,

--field=Severity=Critical

. If the field is multi-valued, value is of the form "fieldName=fieldValue,...".

To specify more than one field, specify this option for each field you want to add to the issue.

To specify an Integrity project, project names must be preceded by a (/), for example, .

--field=Project=/testProject

To specify a configuration management project, use the following syntax: --field=*field-name*=server=*server:port*:: project=*projectname/project.pj*::(devpath=*devpath*|revision=*checkpoint-revision*).

To specify a Parameters field, use the following format:

<Parameter Name =*<Parameter Type <Parameter Name* ;values=*<Parameter Values <Parameter Name* ;description=*<Description*

To specify a Parameter Values field, use the following format:

<Parameter Name =*<Parameter Value <Parameter Name* [;locked]=true|false *<Parameter Name* [;description]=A textual description

The *<Parameter Type*

can be "String" or "Pick List". The "String" type does not have any parameter values. The "Pick List" type has a comma-separated list of values. The locked attribute is optional. If it is not specified the parameter value is not locked. If a parameter name contains a ":", ";" or "=" the character must be escaped. An entry with a given parameter name can only appear once. For more information on parameter fields, see the *PTC Integrity User Guide*.

--richContentField=*value*

specifies a rich content field and its value for the new issue, where value is of the form "richcontentfieldname=fieldValue".

To specify a link to an item by ID: <a href="mks:///item?itemid=

ID">*DisplayText*

For example: Part Requirement

To specify a link to an item by ID and label: *DisplayText*

For example: Part Requirement, Label EDF-343

To specify a link to an item by ID and revision: *DisplayText*

For example: Part Requirement, Revision 1.2

To specify a link to an item by ID and date: *DisplayText* where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

For example: Part Requirement, March 25, 2012 12:04am

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "

b>Feature Overview

/b>"

. In the Windows command line interface (cmd.exe), the ^ escape character must precede the and > characters in HTML tags, for example, ^

b^>Feature Overview^

/b^>

. You may need to escape nested " characters, for example,

--richContentField=Description="Part Requirement"

See Also

- Commands: im copyissue, im editissue, im extractattachments, im viewissue
- Miscellaneous: options

im createproject

creates a project

Synopsis

```
im createproject [--name=value] [--parent=project] [--description=value] [--
permittedAdministrators=u=user1,user2,...;g=group1,group2] [--permittedGroups=group,group,...] [--[no]inheritPermittedGroups] [--
openImage=[none|default|<path>] [--closedImage=[none|default|<path>] [--[no]active] [--user=name] [--hostname=server] [--
password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [(-Ffile|--selectionFile=file)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im createproject` creates a project for workflows and documents. For example:

```
im createproject --name=SavingsManager --inheritPermittedGroups --parent=/FinancialToolkit
```

creates the */FinancialToolkit/SavingsManager* project, inheriting the permitted groups from the parent project *FinancialToolkit*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=`*value*

  specifies the name of the project to create. Names can be a maximum of 100 characters and cannot contain a forward slash (/) or square brackets. This option is mandatory.
- `--parent=`*project*

  specifies the name of the parent project. If specified, then the new project becomes subproject of the parent project. The specified parent project must exist before creating a subproject. When specifying a parent project, you must include a forward slash and the full project name, for example, `/AuroraProject`.
- `--description=`*value*

  specifies a description of the project.
- `--permittedAdministrators=u=`*user1*`,`*user2*`,...;g=`*group1*`,`*group2*`,...`

  Specifies a comma delimited list of users and/or groups that can administer this project. Project Administrators are allowed to edit and view projects. Project Administrators can also manage all child projects of the project they are approved for, but they cannot edit projects assigned to another Project Administrator. Project Administrators can only create and delete subprojects, they cannot create top level projects unless they have been granted the CreateProject ACL permission. Project Administrators can view all users, groups, and dynamic groups. They can also edit dynamic group membership for projects they are assigned to. If not specified, the project creator is the assigned Administrator. For more information, see the *PTC Integrity Server Administration Guide*.
- `--permittedGroups=`*group*`,`*group*`,...`

  specifies a comma delimited list of groups that have visibility for this project.
- `--[no]inheritPermittedGroups`

  controls whether or not permissions are inherited from the parent project. Project settings cannot be inherited unless there is a parent project. Inherited permissions are mutually exclusive with `--permittedGroups`.
- `--openImage=[`*none*`|`*default*`|<`*path*`>]`

  specifies whether an image appears for an open project.

  `--image=`*none* does not specify an image for the project.

  `--image=`*default* specifies the default "open" image for the project.

  `--image=<`*path*`>` specifies the path and name of a custom image for the project, for example, `c:\images\project_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--closedImage=[`*none*`|`*default*`|<`*path*`>]`

  specifies file path for an image icon file used for a closed project when displayed in the graphical user interface. See`--openImage` for details. The default image is a closed folder.
- `--[no]active`

  Specifies if the project is actively used in Integrity. This option is useful for removing inactive projects in your organization. Specifying `--noactive` prevents the project from being displayed in project drop-down lists; however, the project still appears in existing issue data.

See Also

- Commands: im editproject, im viewproject, im deleteproject, im projects
- Miscellaneous: options

im createquery

creates a new Integrity query

Synopsis

```
im createquery [--copyColumnsFromQuery[=[user:]query] [--fields=field,field,...] [--[no]sortAscending][--sortField= field] [--image=[none|default|path][--description=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--name=value] [--hostname=value] [--queryDefinitionFile=value] [--queryDefinition=query] [--sharedAdmin] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

You can define and store any number of queries using Integrity. A query is a request to select and list the issues that meet specific selection criteria. A selection criterion is a logical expression of specific values, or ranges of values, of the fields of the issue. For more information about queries and creating query constraints, refer to the *PTC Integrity User Guide*.

For example, the following creates a new query that displays all Docs type issues that are part of the Release 2.0 project:

- `im createquery --name="Release 2.0 Docs Isssues" --queryDefinition='((field[Type]="Docs")and(field[Project]="Release 2.0"))'`

Note the following:

- When creating a query that includes a project name in a query constraint, the project name must include the forward slash (/).
- You cannot query on configuration management project fields.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.
- Symbolic dates are evaluated on the Integrity Client's time zone.
- When specifying a date range in a query, all dates are treated as timestamps, converted to the time zone on the Integrity Server, and then truncated to a date-only format. The resulting date-only format is then converted to a Structured Query Language (SQL) statement format on the Integrity Server, and the query is run based on the time zone of the server. If the user defining the date range is not in the same time zone as the Integrity Server, a day can be lost or gained at either end of the defined range. The rollback of dates can cause the query results to vary.

Caution

The date range conversion does not cause any problems when the user is in the same time zone as the Integrity Server. For example, if the Integrity Server is in the America/New_York time zone and the following query date range is defined by a user in Germany: Jan 1, 2007 to Jan 31, 2007. The final conversion of the date range is: Dec 31, 2005 06:00:00 AM GMT+01:00 to Jan 31, 2007 06:00:00 AM GMT+01:00. To avoid any date rollbacks when working with query date ranges in Integrity, PTC recommends that users specify the time zone that is used by the Integrity Server to which they are connecting.

- Because dynamically computed fields are not stored in the database, dynamically computed short text fields cannot be located with an all text field search in the Integrity Client. To search for dynamically computed short text fields, create a query that includes a specific \u201cfield contains\u201d comparison. For more information about computed fields, see your administrator.

Options

This command takes the universal options available to all *im* commands, as well as some general options. See the options reference page for descriptions.

- `--copyColumnsFromQuery[=[user:]query]`
- specifies an existing query to copy the column configuration from to use as the default for the new query. If you do not specify a query to copy from, the new query uses the server default column configuration which consists of `ID`, `Type`, `Summary`, `State`, `Assigned User`, and `Project` columns, sorted by ID in ascending order.

Note

`--fields`=*field, field,...*, `--[no]sortAscending`, and `--sortField`=*field* are not mandatory, but if specified, they overwrite the current default column configuration.

- `--fields`=*field,field,...*

  specifies the fields to use as the default columns for the query. This overrides the current default columns.
- `--[no]sortAscending`

  specifies the sort direction that issues are displayed in. This overrides the current default sort direction.
- `--sortField`=*field* specifies the field that issues are sorted by. This overrides the current default sort field.
- `--image=[none|default|<path>]`

  specifies whether an image appears for the new query.

  `--image=none`

  does not specify an image for the query.

  `--image=default`

  specifies the default funnel image for the query.

  `--image`<=*path*> specifies the path and name of a custom image for the query, for example,.

  `c:\images\defect_icon.gif`

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`

  specifies the users and groups that can use and modify the query. Your administrator defines users and groups.
- `--description=value`

  specifies a short description for the new query.
- `--name=value`

  specifies the name of the new query. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--queryDefinition=query`

  specifies a string to define the query constraints. The query must be of the following format:

  <rule> is defined as (*<filtergroup>*)

  <filtergroup> is defined as one of the following:

  (*<filtergroup>* and *<filtergroup>* and ...) (*<filtergroup>* or *<filtergroup>* or ...)((*<filter>*) and (*<filter>*) and ...)((*<filter>*) or (*<filter>*) or ...)

  where

  <filter> is defined as disabled (*<filter>*)*<filter>* is defined as not (*<filter>*)<filter> is defined as *<fields>*| *<subquery>*| *<attachment>*| *<relationship>*| *<branch>*| *<genericcp>*| *<histval>*| *<histdate>*| *<histuser>*| *<timeentry>*| *<testresult>*| *<item>*

  <histuser> is defined as *histuser[Summary|State|..]* was changed by *<user>*

  <histuser> is defined as *histuser.any* field was changed by *<user>**<histdate>* is defined as *histdate[Summary|State|..]* was changed *<datevalue>*

<histdate> is defined as *histdate.any* field was changed *<datevalue><histval>* is defined as *histval[Summary|State|..]<value>*

<genericcp> is defined as *genericcp:<cptype>:<attrfieldidentifier>:[fieldname]*

where *<attrfieldidentifier>* is *attribute* or *entryattribute* and *[fieldname]* is the real name, not the display name, of the attribute. Use the `im viewcptype` command to find the attribute name.

<genericcp> is defined as not *(genericcp:si:attribute[resolutionlist]is empty)*

<genericcp> is defined as *genericcp:<cptype>.exists*

<relationship> is defined as *relationship[ID|Created User|..]* using *[Relationship Field]=<value>*

<relationship> is defined as *relationship.exists backward|forward* using *[Relationship Field]*. To restrict your query to either backward or forward relationships, *backward* must be specified if you specify the Backward Relationships field, and the *forward* option must be specified if you specify the Forward Relationships field or a custom relationship field.

<relationship> is defined as *relationshipFlag [Relationship Flag Name] backward|forward* using *[Relationship Field]*

<attachment> is defined as *attachment[file size|file name|mime type] <value><attachment>* is defined as *attachment.exists*

<timeentry> is defined as *timeentry[issue ID|user|entry date|source|duration|notes|created by|created date|modified by|modified date]*

<timeentry> is defined as *timeentry.exists*

<testresult> is defined as *testresult.isrelatedto <testresult>* is defined as *testresult.exists <testresult>*is defined as *testresult.hasattachment <testresult>* is defined as *testresult.hasrelateditem <testresult>* is defined as *[ID]comparison* where *ID* can be one of *Verdict*, *Verdict Type*, *Modified By*, *Modified Date*, *Session ID*

<branch> is defined as *branch.isBranch*

<branch> is defined as *branch.hasBranch*

<branch> is defined as *branch[X]* in parent contains *Y*. This filter finds items where field X of the branched from item has a specific value. In other words, an item is a child (is a branch), and the parent item has field X with value Y. Examples follow:

<branch> is defined as *branch[X]* in child contains *Y*. This filter finds items where field X of the branched item has a specific value. In other words, an item is a parent (has a branch), and the child item has field X with value Y. Examples follow:

<item> is defined as *item.node*

<item> is defined as *item.content*

<item> is defined as *item.segment*

<item> is defined as *item.subsegment*

<item> is defined as *item.lockeddocument*

<item> is defined as *item.meaningful*

<item> is defined as *item.nonmeaningful*

<item> is defined as *item.shareditem*

<item> is defined as *item.group*

<item> is defined as *item.teststep*

<item> is defined as *item.testcase*

<item> is defined as *item.testsession*

<item> is defined as *item.testsuite*

<item> is defined as *item.live*

<item> is defined as *item.versioned*

Note

If document versioning is enabled, queries return all items (live and versioned) by default. To specify live or versioned items, include the `item.live` or `item.versioned` filter. For more information on document versioning, see the *PTC Integrity User Guide*.

<subquery> is defined as *subquery[Query1|Query2|...]*

<fields> is defined as *field[ID|Created User|Created Date|..] <value>*

<fields> is defined as *field.anyTextField*

If document versioning is enabled, you can specify the following conditions to display live and versioned items by ID:

Note: You cannot display a range of versioned items, for example, "`(field[ID]>123-1.0 and 128-1.0)`".

For more information on live and versioned items, see the *PTC Integrity User Guide*.

<value> is defined as *<value>* or *"is empty"*

<value> is defined as *"is empty"*

<value> is defined as *<leftrangeop> num*and *<rightrangeop> num*

<value> is defined as contains *<text>*

<value> is defined as *<operator> num*and *<operator> num*

<value> is defined as *<operator> num*

<value> is defined as hasSourceLinks

<rightrangeop> is defined as < | <=

<leftrangeop> is defined as | >=

<value> is defined as = <uservalue>, <uservalue>, ..

`<uservalue>` is defined as "`me`" | "`unspecified`" or "`is empty`" | `user1` | `user2` | ...

<value> is defined as <datevalue>

<datevalue> is defined as between mm/dd/yyyy and mm/dd/yyyy <datevalue> is defined as between mm/dd/yyyy hh/mm/ss and mm/dd/yyyy hh/mm/ss (Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM.)
<datevalue> is defined as in the last|next num days|months|years hours|minutes|seconds (Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM.) <datevalue> is defined as today|yesterday|tomorrow

num is defined as .. | -1 | 0 | 1 | ..

<operator> is defined as = | > | >= | <= | < | <>

For example:

```
(((field[Summary]contains"Hello")or(field[Assigned Group]="everyone"))
```

and

```
(attachment.exists))
((field[Summary]contains"Requirement")or(field[Assigned Group]="Project Managers"))
((field[Category]="Technical Requirement") or (field[Category]="System Requirement"))
```

Note

To reference the original query in the new query so that any changes to the original query are reflected in the new query, define <subquery> as subquery[OriginalQuery].

- branch[ALM_Text] in parent contains "Project Description"
- branch[ALM_Actual Budget] in parent = 12
- branch[ALM_Actual Effort] in parent > 20
- branch[Project] in child=/Projects/Release2
- branch[Summary] in child contains "Some issue Summary"
- branch[ALM_Content] in child 10
- display a specific live item only. For example,"`(field[ID]=123)`" returns`123`.
- display a live item and all versions of the item. For example,

  "`(field[ID]=123 include versions)`"

  returns

  `123`,`123-1.0`,`123-1.1`, and`123-1.2`.
- display a range of live items. For example, "`(field[ID]>123 and 128)`" returns `124`, `125`, `126`, and `127`.
- display a specific versioned item. For example,"`(field[ID]=123-1.0)`" returns`123-1.0`.
- `--sharedAdmin`specifies the query as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.
- `--queryDefinitionFile=`*value*specifies a file that contains the complete definition of the query. See `--queryDefinition` for the file format.

See Also

- Commands: im copyquery, im deletequery, im editquery, im viewquery, im queries
- Miscellaneous: options

im createreport

creates a new Integrity report

Synopsis

```
im createreport[--query=[user:]query] [--reportTemplate=value] [--reportTemplateFile=value] [--recipeParam=value] [--recipeParams=value] [--recipeParamsFile=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--name=value] [--description=value] [--sharedAdmin] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F file|--selectionFile=file)][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

Integrity allows you to create, edit, and generate various reports on the data in your projects. You can share your report with selected user groups, or you can keep it private. Only a user who created a report can modify it, delete it, and decide which user groups can see the report. Reports can be printed, displayed on screen, or exported as an HTML file for viewing on the Web.

For example,

```
im createreport --name="Docs Daily Status" --query="Docs Items Worked On This Week" --reportTemplateFile=c:/Program Files/Integrity/IntegrityServer10/data/reports/recipes/One column.html
```

creates a new report based on a query and template.

For more information on reports, see the *PTC Integrity User Guide*.

Note the following:

- Reports can do more than just display field information. You can also perform arithmetic calculations between numeric fields, displaying the values in the report. For example, you can add up column totals or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- You cannot create or edit a query while creating a report.
- A report can be edited by the user who created it. Principals (users and groups) that a report is shared to can edit it if they have edit permissions assigned to them by the report creator. A report can only be deleted by the user who created it or by the administrator.
- Because reports are based on queries, reports are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or contact your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Creating deeply nested reports with a large number of inter-related issues can create extremely large reports and/or cause the Integrity Server to stop responding. When creating a report, take into consideration that the average number of links per issue and the number of levels in the report multiply the size of the report.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- Although the electronic signature fields Signed By and Signature Comment are only visible in an issue's history (if enabled by your administrator), you can report on the historical values by specifying the fields in the report.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--query=`*`[user:]query`*

  specifies the name of the query that defines the selection criteria for the report, and the user who created the query.
- `--reportTemplate`*`value`*

  specifies the report template on which the report is based. For information on the report template format, see the *PTC Integrity Server Administration Guide*.
- `--reportTemplateFile=`*`value`*

  specifies the file name that contains a report template. For information on the report template file format, see the *PTC Integrity Server Administration Guide*.
- `--recipeParam=`*`value`*specifies a single parameter *key:value* pair.

  The report parameters (or attributes) determine what is displayed to users when creating and editing reports in the Report Wizard.

  The available list of report recipe parameters and attributes can be viewed for each report type using the `im viewreport` and `im reports` commands. For more detailed information see the knowledge base in the Support section of the PTC Web site (http://www.ptc.com).
- `--recipeParams=`*`value`*specifies a semicolon separated list of report recipe parameters or attributes in `key:value` pairs.
- `--recipeParamsFile=`*`value`*

  specifies the file that contains the recipe parameters in the report.
- `--shareWith=u=`*`user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`*

  specifies the users and groups that can use and modify the report. Your administrator defines users and groups.
- `--sharedAdmin`

  specifies the report as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

⚠ Caution

Once a user object is converted to a system provided object, you cannot revert it to a user object again.

- `--name=value` specifies the name of the report. This field is mandatory. Names may be a maximum of 100 characters, and cannot contain square brackets.
- `--description=`*`value`* specifies a description for the report.

See Also

- Commands: im editreport, im copyreport, im viewreport, im runreport, im deletereport, im reports
- Miscellaneous: options

im createsegment

creates a new segment

Synopsis

```
im createsegment[--insertMode=[reference:include]] [--insertLocation=number[first|last|before:name|after:name] [--parentID=value] [--type=type] [--addAttachment=value] [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--customFieldDefinition=value] [--customFieldValue=value] [--field=value] [--richContentField=value] [--project=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [--quiet] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im createsegment` creates a new segment. A segment can be a document root or a subsegment. For example:

```
im createsegment --field='Project=/Project2/Release 1.0' --type='Test Document' --field='Category=Document'
```

creates a test document

```
im createsegment --parentID=123 --field='Project=/Project2' --type='Requirement Document'
```

creates a requirement document under document root 123.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--insertMode=[reference:include]` indicates how the segment is to be inserted into a segment root. This option is required if you specify a `--parentID`. The options are as follows:

  reference references related issues but does not include them.

  include includes related issues as children of the parent issue.
- `--field=value`

  specifies a field and its value, where *value* is of the form "*fieldName=fieldValue*", for example, `--field=Severity=Critical`. If the field is multi-valued, *value* is of the form "*fieldName=fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add.

  To specify an Integrity project, project names must be preceded by a (/), for example, `--field=Project=/testProject`.

  To specify a configuration management project, use the following syntax: `--field=`*fieldname*`,server=`*server*, `project=`*projectname*, [`devpath=`*devpath*`|revision=`*checkpointrevision*]
- `--addRelationships=value`adds related issues to the segment, where `value` is of the form [*FieldName*]:*IssueID*[*relationshipFlags*][,...]. If no field name is specified, the Forward Relationships field is used.

Note

Adding a related issue is only permitted if your administrator has allowed relationships for the segment's issue type.

- `--addSourceTrace=value`

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName] [,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
- `--addSourceLink=value`

  adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber,[file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName] [,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*".

  Note the following:
- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
- `--richContentField=value`

  specifies a rich content field and its value for the segment, where *value* is of the form "*richcontentfieldname=fieldValue*".

  To specify a link to an item by ID:

  ```
  <a href="mks:///item?itemid=ID">DisplayText</a>
  ```

  For example:

  ```
  <a href="mks:///item?itemid=4547">Part Requirement</a>
  ```

  To specify a link to an item by ID and label:

  ```
  <a href="mks:///item?itemid=ID&label=LABEL">DisplayText</a>
  ```

  For example:

  ```
  <a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>
  ```

  To specify a link to an item by ID and revision:

  ```
  <a href="mks:///item?itemid=ID&revision=REVISION">DisplayText</a>
  ```

  For example:

  ```
  <a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>
  ```

  To specify a link to an item by ID and date:

  ```
  <a href="mks:///item?itemid=ID&asof=DATETIME">DisplayText</a>
  ```

  where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

  For example:

  ```
  <a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>
  ```

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example,

```
""<b>Feature Overview/b>"
```

In the Windows command line interface (cmd.exe), the ^ escape character must precede the and > characters in HTML tags, for example,

```
^^<b^>Feature Overview^/b^>
```

You may need to escape nested " characters, for example,

```
--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"
```

- `--addAttachment=`*value*

  adds attachments to the segment, where `value` is of the form "*fieldName*`,path=`*pathToFile*`[,name=`*nameOfAttachment*`][,summary=`*shortDescription*`]" .

Note

The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

For example,

- `--type=`*type*

  specifies the type of the segment. Your administrator defines segment types. This option is required if more than one segment type is allowed and the type field is not specified.
- `--parentID=`*value*

  the ID of the parent segment or node that will contain the reference to the segment being created. This option is required.
- `--insertLocation=`*number[first|last|before:name|after:name]*

  determines where the segment should go in the parent segment's structural relationship list. You must specify a `--parentID`. The options are as follows:

  first inserts the segment at the beginning of the list

  last inserts the segment at the end of the list

  before:<ID> inserts the segment before the specified ID

  after:<ID> inserts the segment after the specified ID

  [0,...] inserts the segment at the specified location. If a negative is specified, the segment is inserted at the beginning of the list. If the number specified is too large, the segment is inserted at the end of the list.
- `--insertMode=`*[reference:include]*

  indicates how the segment is to be inserted into a segment root. This option is required if you specify a `--parentID`. The options are as follows:

  reference references related issues but does not include them.

  include includes related issues as children of the parent issue.

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
```

adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. A segment cannot have more than one attachment with the same name.

Note

Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

See Also

- Commands: im viewsegment, im branchsegment, im insertsegment, im importsegment
- Miscellaneous:diagnostics, options, preferences

im createsolution

exports an existing workflow into a solution file

Synopsis

```
im createsolution [--buil=value] [--defaultPrefix=value] [--description=value] [--[no]includeIssues] [--major=value] [--minor=value]
[--siProjectDir=value] [--solutionFile=value] [--verboseDescription=value] [--hostname=server] [--user=value] [--password=password]
[--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

To create a solution for installation on other Integrity Servers, you can use the Admin Migration Wizard to migrate specific workflow objects, such as an item type and its fields, to an empty production server (an Integrity Server containing a new database and only the system provided objects that come with Integrity). From the production server, you can use the `im createsolution` command to export the workflow to a solution file (`.imt`). For example, if you have a server containing test workflows and you create one that you want to use in your organization, you can export that specific workflow into a solution file and then install it on the production server.

Tip

Exporting a workflow to a solution file also allows you to create a snapshot of a workflow on physically separate server, without having to copy all of the system provided objects and user data (as would be the case if you performed a database backup/restore operation).

Note the following about creating and installing a new solution:

- PTC recommends excluding existing items and user data from the new solution. Creating a solution from a workflow on an Integrity Server containing several hundred or more items and user data may cause the server to run out of memory. If you create a solution from a server containing hundreds or thousands of users, the solution installer prompts you to map each user to an existing user on the destination server.
- PTC recommends excluding configuration management projects from the new solution. Configuration management projects in PTC provided solutions demonstrate how a configuration management project with a simple project tree works with Integrity; however, the admin migration wizard does not allow you to migrate configuration management projects, preventing you from selecting specific projects for a destination server.
- If the Integrity data you are creating a solution from references non-existent or broken Integrity objects (for example, admin queries referencing non-existent objects or broken item presentation templates), you may encounter problems when running the create solution command. PTC recommends verifying the integrity of your data before creating a solution.
- In previous releases, you could only install a solution onto a server that was the same version of Integrity as the one you created the solution with. With MKS Integrity 2009, you can create solutions and install them on later versions of Integrity.
- When installing the solution, some Integrity object names may conflict with existing object names. This is most likely to occur if you are installing the solution without a prefix. To resolve this issue during the installation, you can optionally specify a conflict prefix in the solution installation wizard which prepends the prefix to the name of the Integrity object.
- If you create a solution from an existing server, you must install the solution without a prefix for it to install successfully. Solutions provided with Integrity can be installed with a prefix.
- If you create a solution containing a workflow that uses source code projects, the projects must be installed as sample data for the solution to work properly. If you do not install the projects as sample data, the workflow references non-existing projects, which can cause errors.
- You cannot uninstall a solution, or update an earlier version of a solution with a newer one.

To create a solution from an existing workflow:

1. Start the empty production server that you want to migrate the existing workflow to.
2. On the staging server containing the workflow you want to migrate, set the properties specified in the *PTC Integrity Server Administration Guide*, and add the following property in the *installdir*`/config/properties/is.properties` file:

   ```
   mksis.im.stageToEmptyServer=true
   ```

   This removes the restriction that the staging and production servers require identical data.
3. Start the staging server.
4. On the staging server, migrate an existing workflow to the empty production server, specifying only the objects you want to include in the solution.
5. On the empty production server, make changes to the workflow if necessary.
6. Once you are satisfied with the workflow you want to export, use the `im createsolution` command.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--build=`*value*

  specifies the build number of the solution.
- `--defaultPrefix=`*value*

  specifies the default prefix to use during installation, for example, `RM_`.
- `--description=`*value*

  specifies the name of the solution being created.
- `--[no]includeIssues`

  specifies whether to include all items in the database with the solution.
- `--major=`*value*

  specifies the major version number of the solution.
- `--minor=`*value*

  specifies the minor version number of the solution.
- `--siProjectDir=`*value*

  specifies the directory containing the configuration management project file (`.pj`) to include in the solution.
- `--solutionFile=`*value*

  specifies the name of the file to store the solution template to.
- `--verboseDescription=`*value*

  specifies a multi-line description of the solution that displays in the solution installation wizard.

See Also

- Commands: im installsolution, im releaseadminlock, im viewadminlock, im obtainadminlock
- Miscellaneous: options

im createstate

creates a state

Synopsis

```
im createstate [--name=value] [--description=value] [--image=[none|default|<path>] [--position=<number>|first|last|before:
<name>|after:<name>] [--
capabilities=MKSSI:OpenChangePackages,MKSSI:ChangePackagesUnderReview,MKSIM:TimeTracking,MKSTM:ModifyTestResult] [--user=name] [--
hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--
status=[none|gui|default]]
```

DESCRIPTION

`im createstate` creates a state for workflows and documents. For example:

```
im createstate --description="Initial state" --name=Submit
```

creates the *Submit* state with a description.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the name of the state to create. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--description=value`

  specifies a description of the state.
- `--image=none|default|<path>`

  specifies whether an image appears for the state.

  `--image=none`
  does not specify an image for the state.

  `--image=default` specifies the default image for the state.

  `--image=<path>` specifies the path and name of a custom image for the state, for example, `c:\images\state_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--position=<number>|first|last|before:<name>|after:<name>`

  specifies the position in the order of states. The first position is `0`.
- `--capabilities=MKSSI:OpenChangePackages,MKSSI:ChangePackagesUnderReview,MKSIM:TimeTracking,MKSTM:ModifyTestResult`

  sets the given state capabilities to allow open change packages, change packages under review, time entries, or modifications to test results.

See Also

- Commands: im editstate, im viewstate, im deletestate, im states
- Miscellaneous: options

im createtrigger

creates an event trigger

Synopsis

```
im createtrigger [--name=name] [--type=[scheduled|rule|timeentry|copytree|branch|label|testresult|lock|unlock]] [--runAs=user] [--query=[user:]query] [--position=<number>|first|last|before:<name>|after:<name>] [--description=value] [--frequency=[manual|hourly|daily|monthly]] [--script=filename] [--scriptParams=[arg=value[;arg2=value2...]]] [--scriptTiming=pre|post|pre,post|none] [--assign=[field=value[;field2=value2...]]] [--rule=value] [--copyRule=trigger] [--ruleFile=filename] [--hostname=server] [--user=value] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im createtrigger` creates an event trigger for workflows and documents. An event trigger for workflows and documents contains a list of script files that Integrity runs based on either a defined rule, or a scheduled query. Event triggers must reside on the server machine, where the Integrity Server executes them. Integrity does not support client-side event triggers for workflows and documents. As a result, all references must be relative to the server. For more information on Integrity event triggers, see the *PTC Integrity Server Administration Guide*. For example:

```
im createtrigger --type=scheduled --name=sendmail --script=javamail.js --runAs=jriley --query=UrgentDefects
```

creates the scheduled trigger *sendmail*, using the *javamail.js* script and *UrgentDefects* query.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=`*name*

  specifies the name of the trigger. May be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--type=[`*scheduled*`|`*rule*`|`*timeentry*`|`*copytree*`|`*branch*`|`*label*`|`*testresult*`|`*lock*`|`*unlock*`]`

  specifies the type of trigger. If you do not specify a type, `rule` is specified by default. If you specify `timeentry`, the values for `--rule`, `--ruleFile`, `--runAs`, `--frequency`, `--assign`, and `--query` are ignored.

  Note

  Scheduled event triggers are evaluated on the Integrity Server's time zone.

- `--runAs=`*user*

  specifies the user for a scheduled trigger. All modifications appear as though they were made by the specified user.
- `--query=`*[user:]query*

  specifies the name of the query and to use for a scheduled trigger, and the username of the user who created the query.
- `--position=`*<number>*`|`*first*`|`*last*`|`*before:<name>*`|`*after:<name>*

  specifies the position of trigger in the run order.
- `--description=`*value*

  specifies description of what the trigger does.
- `--frequency=[`*manual*`|`*hourly*`|`*daily*`|`*monthly*`]`

  specifies the frequency of a scheduled trigger. By default, `manual` is specified. May be the following:

  ```
  [manual]
  [hourly [start=[00:]mm]            [hours=00,01,...,23]]
  [daily [start=hh:mm]               [days=mon,tue,...]]
  [monthly [start=hh:mm]             [day=1-31] [months=jan,feb,...]]
  ```
- `--script=`*filename*

  specifies the filename of the script, for example, `scriptfile.js`. The script must be located in the following directory: `<installdir>/data/triggers/scripts`.
- `--scriptParams=`*[arg=value[;arg2=value2...]]*

  specifies the list of script arguments.
- `--scriptTiming=[`*pre*`|`*post*`|`*pre,post*`|`*none*`]`

  specifies if the script should be run pre, post, both pre and post on issue commit to the database, or no timing is associated with the script. If you do not specify an option, `none` is specified by default.
- `--assign=`*[field=value[;field2=value2...]]*

  specifies the list of field assignments.
- `--rule=`*<rule>*

  specifies the rule to associate with the trigger. For the rule syntax, see Specifying Rules on the options reference page.

  Note

  - To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *today*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can choose yourself by specifying "`me`". "`me`" is a symbolic user which refers to the currently logged in user. For example, you could create an event trigger that specifies Project issues can only be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
  - Configuration management project fields are invalid in event trigger rules.

- `--copyRule=`*trigger*

  copies the given trigger's rule as it exists at the time of the copy. This does not create a pointer to another rule.
- `--ruleFile=`*filename*

  specifies a file containing the rule text. See `--rule` for notes on the command use and where to obtain file format information.

See Also

- Commands: im copytrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers, im echo
- Miscellaneous: options

im createtype

creates an issue type

Synopsis

```
im createtype [--addWordTemplate=name=templateName,path=templatePath[,description=description][,defaultEdit]] [--testResult] [--
removeWordTemplate=value] [--editPresentation=value [--printPresentation=value [--printReport=report [--viewPresentation=value [--
name=value] [--image=[none|<path>] [--description=value] [--[no]enableRevision] [--majorRevisionRule=rulename] [--
minorRevisionRule=rulename] [--copyMajorRevisionRule=type] [--copyMinorRevisionRule=type] [--majorRevisionRuleFile=filename] [--
minorRevisionRuleFile=filename] [--position=<number>] [--[no]allowChangePackages] [--createCPPolicy=value] [--
[no]duplicateDetectionMandatory] [--duplicateDetectionSearchField=field] [--documentClass=[none|segment|node|shareditem]] [--
associatedType=type] [--significantFields=field,field,...] [--defaultReferenceMode=[Reuse|Share]] [--
versionEditFields=field,field,...] [--visibleFields=field:group,group,...[;...]] [--notificationFields=field,field,...] [--
stateTransitions=state:state:group|dynamic group,group|dynamic group,...[;...]] [--addLabelRule=rule] [--moveLabelRule=rule] [--
copyMoveLabelRule=type] [--moveLabelRuleFile=filename] [--properties=name:value:description[;...]] [--addLabelRuleFile=value] [--
branchRule=rule] [--branchRuleFile=value] [--branchFields=field,field,...] [--copyAddLabelRule=type] [--copyBranchRule=type] [--
copyCopyTreeRule=type] [--copyDeleteLabelRule=type] [--copyTreeRule=type] [--copyFields=field,field,...] [--copyTreeRuleFile=value]
[--deleteLabelRule=type] [--defaultReferenceMode=[reuse|share]] [--deleteLabelRuleFile=filename] [--[no]enableDeleteItem] [--
deleteItemRule=value] [--deleteItemRuleFile=value] [--copyDeleteItemRule=type] [--[no]enableBranch] [--[no]enableCopyTree] [--
[no]enableLabel] [--testRole=[none|testSession|testCase|testStep|testSuite]] [--[no]enableTestSteps] [--
[no]enableTestResultRelationship] [--testCaseResultFields=field,field,...] [--testSessionDateField=value] [--
modifyTestResultPolicy=value] [--[no]groupDocument] [--editabilityRule=rule] [--editabilityRuleFile=filename] [--
copyEditabilityRule=type] [--mandatoryFields=state:field,field,...[;...]] [--addFieldRelationship=constraint] [--
fieldRelationships=constraint[;constraint]] [--removefieldRelationship=constraint] [--fieldRelationshipsFile=value] [--
[no]showHistory] [--[no]showWorkflow] [--phaseField=field] [--[no]enableProjectBacking] [--[no]enableTimeTracking] [--
permittedAdministrators=u=user1,user2,...;g=group1,group2] [--permittedGroups=group,group,...] [--[no]allowDocumentLocks] [--
documentLockPolicy=value] [--documentUnlockGroup=group] [--[no]allowAdditionalLockFields] [--additionalLockFieldsRule=rule] [--
additionalLockFieldsRuleFile=filename] [--copyAdditionalLockFieldsRule=type] [--additionalSegmentLockFields=field,field,...] [--
additionalContentLockFields=field,field,...] [--[no]lockingRequired] [--lockingRequiredRule=rule] [--
lockingRequiredRuleFile=filename] [--copyLockingRequiredRule=type] [--user=name] [--hostname=server] [--password=password] [--
port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--
hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]] [--status=[none|gui|default]]
```

Description

`im createtype` creates an issue type. For example:

```
im createtype --name=Defect --description="Captures software defects" --allowAttachments --allowChangePackages
```

creates the *Defect* type with a description, allowing attachments and change packages.

Caution

A type can contain a maximum of 1000 fields. Exceeding the maximum number of fields results in exception errors when creating, editing, or viewing the issue type in the GUI.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--testResult`

  specifies whether the type is used for test results only. For more information on using types for test results, see the *PTC Integrity Server Administration Guide*.
- `--editPresentation=`*value*

  specifies the presentation template to use when editing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--printPresentation=`*value*

  specifies the presentation template to use when printing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--printReport=`*report*

  specifies the administrator report that will provide the structure and format when a user prints a document using **Document ▸ Print** from the GUI. This must be specified with `--documentClass=`*segment*.
- `--viewPresentation=`*value*

  specifies the presentation template to use when viewing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--name=`*value*

  specifies the name of the type. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--image=[`*none*`|<`*path*`>]`

  specifies whether an image appears for the type.

  `--image=`*none* does not specify an image for the type.

  `--image=<`*path*`>` specifies the path and name of a custom image for the type, for example, `c:\images\type_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=`*value*

  specifies a description of the type.
- `--[no]enableRevision`

  specifies whether to enable item revisioning for the specified item type or enable document versioning for the specified document model type. For more information on item revisioning or document versioning, see the *Integrity Server Administration Guide*.
- `--majorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the major revision. For example, if the major rule is `Type=Requirement`, then only items of type Requirement may have major revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--minorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the minor revision. For example, if the minor rule is `State=Open`, then only items in an open state may have minor revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyMajorRevisionRule=`*type*

  copies the major revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one being created.
- `--copyMinorRevisionRule=`*type*

  copies the minor revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one

being created.

- `--majorRevisionRuleFile=filename`

  specifies a file that contains the major revision rule for the specified item type (used with item revisioning) or document model type (used with document versioning). For the file format, see Specifying Rules on the options reference page.

- `--minorRevisionRuleFile=filename`

  specifies a file that contains the minor revision rule for the specified item type (used with item revisioning) or document model type (used with document versioning). For the file format, see Specifying Rules on the options reference page.

- `--documentClass=[none|segment|node|shareditem]`

  allows you to specify a document class on a type. Each document domain requires a segment, a node type associated with the segment, and a shared item type associated with the node type. Each document instance requires the segment root and a set of nodes. The options are as follows:

  - `none` specifies that this item type is not used in documents. The default for all non-document types.
  - `segment` specifies that this type is used in the segment root. For example, you need to specify segment if you are creating a document domain.
  - `node` specifies that this type is used in nodes (content). Node correlates to the content (i.e. Requirement, Input, Test, Specification) item type. To create content, you need a node and a corresponding shared item. Therefore, you would create two types: one with `documentClass=node`, and the other with `documentClass=shared item`. Each node in a document is a reference to a shared item or a subsegment, and all three types expose arbitrary fields. Nodes also expose FVA fields from the shared item.
  - `shared item` specifies that this type is a shared item. Shared items should be associated with a corresponding node type (content). Each node class on a type requires an associated shared item.

  Example:

  ```
  im createtype --name=Document --description="Requirement Document"
  --documentClass="segment" --associatedType="Requirement(node name)"
  ```

  creates a segment with an associate node type called `Requirement`.

- `--[no]duplicateDetectionMandatory`

  makes it mandatory for users to view potential duplicates before they create a new item.

- `--duplicateDetectionSearchField=field`

  specifies the name of the short text field to search when using duplicate detection. Setting an empty field means that duplicate detection will not be available to users. The short text field cannot be a computational field or Field Value Attribute to a short text field. Before enabling duplicate detection, the selected short text field must be visible on the type. In addition, to use duplicate detection searches on a short text field, the created field must have text search indexing enabled. When searching for potential duplicates, the default number of results is 20 items.

- `--testRole=[none|testSession|testCase|testStep|testSuite]`

  allows you to specify the test management role for the type. The options are as follows:

  - `none` specifies that this item type does not a have a specific test management role. However, it can still be related to test results using the `--[no]enableTestResultRelationship` option.
  - `testSession` specifies that this type is a test session. A test session is a concrete run or execution of a group of test cases. If a type is a test session, also specify `modifyTestResultPolicy`.
  - `testCase` specifies that this type is a test case. A test case describes the test conditions and provides a container for adding test results from the test session. A test case must have the `--documentClass` option set to node. If a type is a test case, also specify `--[no]enableTestSteps` and `--testCaseResultFields`.
  - `testStep` specifies that this type is a test step. A test step is a specific testing operation performed as part of executing a test case.
  - `testSuite` specifies that this type is a test suite. A test suite is a grouping of test cases. A test case must have the `--documentClass` option set to `segment`.

- `--associatedType=type`

  specifies an associated type for segments and nodes. If the document class is `segment`, the associated type is of type node. If it is `node`, the type is of shared item.

- `--defaultReferenceMode=[Reuse|Share]`

  specifies the reference mode to apply to nodes after they are branched. `Reuse` points to the same shared item until the original node changes and a new shared item is created.`Share` points to the same shared item as the original branched node. Editability will not be allowed on shared fields; this mode only allows observation of the changes made on the other item.

- `--significantFields=field,field,...`

  allows you to specify additional fields by type that, when edited, results in an update to the content revision. This is reflected in a change to the revision date in the item history. Each type has default associated significant edit fields.

- `--position=<number>`

  specifies the position in the list of types.

- `--addLabelRule=rule`

  specifies the rules for when users are permitted to add a label rule. For the rule syntax, see Specifying Rules on the options reference page.

- `--addLabelRuleFile=filename`

  specifies a file that contains the add label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page.

- `--moveLabelRule=rule`

  specifies the rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--copyMoveLabelRule=type`

  specifies the type from which to copy rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--moveLabelRuleFile=filename`

  specifies a file that contains the move label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page.

- `--branchRule=rule`

  specifies the rules for branching by users. For the rule syntax, see Specifying Rules on the options reference page.

Note

Users or groups do not require editability of the issue or any given field to be able to branch an issue; however, they must have project or type visibility to branch an issue. If a user does not have visibility to a given field during a branch operation, the field will be copied unchanged.

- `--branchRuleFile=filename`

  specifies a file that contains the branch rule set for the specified issue type. For the file content format, see Specifying Rules on the options reference page.

- `--branchFields=field,field,...`

  specifies the list of fields to make visible on the Branches tab for viewing item branches.

Note

The branched fields that can be configured are only visible fields on the item type. This means that field visibility permission is checked when adding a branched field. The `--branchFields` option can be used with `--visibleFields=field:group,group,...[;...]` for each command.

- `--copyAddLabelRule=type`

specifies the rules for when users or groups are permitted to copy an add label rule.

- `--copyBranchRule=`*`type`*

  specifies the rules for when users are permitted to copy a branch rule.
- `--copyCopyTreeRule=`*`type`*

  specifies the rules for when users or groups have the ability to copy the copy tree rule for the specified issue type.
- `--copyDeleteLabelRule=`*`type`*

  copies the delete label rule of an existing type to the one being edited.
- `--[no]enableBranch`

  specifies whether issues of this type can be branched. For more information on branching, see the *PTC Integrity User Guide*.
- `--[no]enableCopyTree`

  specifies whether users or groups can copy a tree of issues belonging to the hierarchy for this type.
- `--[no]enableLabel`

  specifies whether users or groups can add labels to issues of this type.
- `--copyTreeRule=`*`rule`*

  copies the tree rule of an existing type to the one being created.
- `--copyTreeRuleFile=`*`value`*

  specifies a file that contains the copy tree rule set for the specified issue type.
- `--deleteLabelRule=`*`type`*

  specifies the rule for when users or groups can delete label rules.
- `--deleteLabelRuleFile=`*`value`*

  specifies a file that contains the delete label rule set for the specified issue type.
- `--[no]enableDeleteItem`

  specifies whether items of the specified type can be deleted. If items of the specified type can be deleted, a rule set must specified with `--deleteItemRule=value` or `--deleteItemRuleFile=value`.
- `--deleteItemRule=`*`value`*

  allows specific users or groups the ability to delete items of the specified item type. To change the rule on who can delete items, users or groups require the `ModifyDeleteItemRule` permission.

Note

Important: The `ModifyDeleteItemRule` permission differs from the `DeleteItem` permission, which allows users or groups to delete items of any type without a defined rule. To define strict item deletion rules in your organization, PTC recommends defining a delete item rule and assigning the `ModifyDeleteItemRule` permission to the appropriate users and groups.

- `--deleteItemRuleFile=`*`value`*

  specifies a file that contains the rule set for deleting items of the specified item type.
- `--copyDeleteItemRule=`*`type`*

  specifies a type to copy an existing delete rule set from.
- `--[no]allowChangePackages`

  specifies whether configuration management change packages are permitted for the type.
- `--[no]groupDocument`

  specifies whether items of the specified type will be used to group content. For group documents, the `--documentClass` option must be `segment`.
- `--[no]enableTestSteps`

  specifies whether the test case type can use test steps. A test step is a specific test for a test case. A test case can contain an ordered list of steps to follow in order to complete the test. Test steps can be shared across test cases.
- `--[no]enableTestResultRelationship`

  specifies whether the type can be related to test results. For example, defects for failed test results.
- `--testCaseResultFields=`*`field,field,...`*

  specifies fields from the test case type that will display in the Test Result Details view.
- `--modifyTestResultPolicy=`*`value`*

  specifies which users and/or groups are allowed to modify test results for the test session item type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to the test session are permitted to create test results. Valid values are `userField=<`*`field`*`>`, `groupField=<`*`field`*`>`, *`anyone`*, `groups=`*`group1,group2,...`*.
- `--createCPPolicy=`*`value`*

  specifies which users and/or groups are allowed to create change packages against issues of this type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to issues within that type are permitted to create change packages. Valid values are `userField=<`*`field`*`>`, `groupField=<`*`field`*`>`, `anyone`, `groups=`*`group1,group2,...`*.
- `--versionEditFields=`*`field,field,...`*

  specifies the fields that users will be able to edit on document and content versions. For document and content versions, you can allow editing only on pick fields and relationship fields. Fields that are included in the `--significantFields` list may not also be included in `--versionEditFields` list. Conversely, fields that are included in the `--versionEditFields` list may not also be included in `--significantFields` list. For more information on configuring editable fields for document versions, see the *PTC Integrity Server Administration Guide*.
- `--visibleFields=`*`field:group,group,... [;...]`*

  specifies which groups have visibility for which fields. By default, the everyone group is specified for each field you specify. All previous values are replaced with specified values.

Note

Special fields are always visible, even if they are not specified.

- `--notificationFields=`*`field,field,...`*

  specifies notification fields for including in every email notification related to this type.

Note

SI project and attachment fields cannot be specified.

- `--copyFields=`*`field,field,...`*

  specifies the list of fields to be copied by default for items of this type. This setting overrides `--copyCommonFields` if set. Users can override these default fields by adding and removing any fields that they can view and edit.
- `--stateTransitions=`*`state:state:group|dynamic group,group|dynamic group,...[;...]`*

  specifies a state transition from one state to another, and the groups/dynamic groups permitted to make that state transition. At least one group/dynamic group must be specified for each transition.

- `--properties=name:value:description[;...]`

  defines type properties. Type properties provide custom code (such as triggers and API) with attributes to read and act on based on their values. The following can be specified for each property:

  - *name* specifies the name of the property.
  - *value* specifies the value for the property.
  - *description* specifies an optional description for the property.
- `--editabilityRule=rule`

  specifies the rules for when users are permitted to edit the type. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *today*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
- When specifying a user, you can choose yourself by specifying "me". "me" is a symbolic user which refers to the currently logged in user. For example, you could create an editability rule that specifies a Project type issue can be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
- SI project and attachment fields cannot be specified.
- If a group name contains blank spaces, a second pair of quotes around “”group name”” is required. For example:

  ```
  im editfield --hostname=server --port=7001 --editabilityRule="((user is a member of "Project Management" or (user is a member of ""Project Management""))" fieldname
  ```

- `--editabilityRuleFile=filename`

  specifies a file that contains the issue editability rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyEditabilityRule=type`

  copies the issue editability rule of an existing type to the new one being created.
- `--mandatoryFields=state:field,field,...[;...]`

  specifies mandatory fields for a state in the type's workflow. All previously specified mandatory fields are replaced with the new values. Mandatory fields can also be set on type constraints. For more information on constraints, see the *PTC Integrity Server Administration Guide*.
- `--addFieldRelationship=constraint`

  specifies a single constraint (field relationship) to be added to a type.
- `--removeFieldRelationship=constraint`

  specifies a single constraint (field relationship) to be removed from a type.
- `--fieldRelationships=constraint[;constraint]`

  specifies constraints between fields. All previously existing constraints on this type are replaced with the new constraints specified here. The syntax specified below also applies for the `--addFieldRelationship` and `--removeFieldRelationship` options.

  There are four constraint methods: Basic, Field Relationship, Rule, and Item Backed Pick List (IBPL). For more information on constraints and constraint methods, see the *PTC Integrity Server Administration Guide*.

Note

Important: If you are creating or copying a type, you cannot reference the name of that new type when creating a basic constraint. Other rule-based constraints, such as Rule and IBPL, can be created; however, errors may occur if you attempt to create a constraint based on a rule that includes the new type name, for example, `(Field[Type]=new type name)`

where *constraint* is one of:

a Basic or Field Relationship constraint:

sourceField=sourceValue1[,sourceValue2,...]:targetValue `[:[all][,mandatory][,errInvalidated=`*invalidMessage*`][,errMandatory=`*mandatoryMessage*`][,description=`*descriptionText*`]]`

a Rule constraint:

`constraintrule=`*(rule) :targetValue*`[:[all][,mandatory][,errInvalidated=`*invalidMessage*`][,errMandatory=`*mandatoryMessage*`][,description=`*descriptionText*`]]`

an IBPL constraint:

`rule=(`*ibplRule*`):`*ibplField* `[:[mandatory][,errInvalidated=`*invalidMessage*`][,errMandatory=`*mandatoryMessage*`][,description=`*descriptionText*`]]`

and where *sourceField* is any field with predefined values (Pick, User, Group, IBPL, Boolean, Project, State, Phase, Type). For the basic constraint method, *sourceField* must be the type you are editing, and the source value must be the specified type.

and where *targetValue* is:

targetField=[value1][,value2,...] where *targetField* is a field of type other than User/Group.

or

targetField=[value1][,value2,...]`[,hasProjectPermission]` where *targetField* is a field of type Group.

or

targetField=[value1][,value2,...]`[,hasProjectPermission][memberOf(`*dynamicGroup1[,dynamicGroup2,...]*`)][valueOf(`*group*`)]`

where *targetField* is a field of type User.

targetField can be any field that is visible on the type and that is not the *sourceField*. Only the following fields can have values specified: User, Group, Pick, State, Project, IBPL, Logical. All other fields (such as Attachment, Integer, Short Text, etc...) can only be made mandatory.

Specifying values (*value1, value2,* etc...) is optional; however, if not specified, the `all` and `mandatory` options must be used to indicate that the field is mandatory and all values are acceptable.

Note

For all *targetField* types, if a mandatory option is specified, then at least one value (*value1,value2, ...*) must be specified, or the `all` option must be present.

`hasProjectPermission` constrains the values of the field to only those users or groups that have permissions for the item's project. This option cannot be specified with any values (`value1, value2...`), or with `memberOf` or `valueOf` options.

`memberOf` constrains the values of the field only to those users that are a members of the listed dynamic groups (*(dynamicGroup1, dynamicGroup2, ...)*. Note that the `everyone` group can also be used as a value for the dynamic groups. This option cannot be specified with any values (*value1, value2...*), or with `hasProjectPermission` or `valueOf` options.

`valueOf` constrains the values of the field to only the specified group. This option cannot be specified with any values (*value1, value2...*), or with `hasProjectPermission` or `memberOf` options.

and where *rule* is a specified rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

and where *ibplRule* is a specified IPBL rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

The IBPL rule can be made up of:

- sub-rules on the field values of the items backing the IBPL target. In the CLI, add an apostrophe (') at the end of the field to indicate the Target Field option in the GUI.
- sub-rules on the item being edited. If no apostrophe is added, then the 'Editing Item Value' GUI option is selected.

In addition:

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

and where *ibplField* is an IBPL field that is visible on a type.

and where *invalidMessage* is any string up to 2000 characters.

and where *mandatoryMessage* is any string up to 2000 characters.

and where *descriptionText* is any string up to 200 characters.

Examples:

Basic Constraint Method:`Type=AdminRequest:"Assigned User"=memberOf(manager,administrator):mandatory,errMandatory="The {ConstrainedField} cannot be empty."`

Specifies that all items of the `AdminRequest` type must be assigned to a user who is either in the manager or an administrator group, and that the "Assigned User" field is mandatory.

Field Relationship Constraint Method:

`Importance=High,Critical:Escalated=True:mandatory,description="Ensure that the Incident is escalated if Importance is either High or Critical".`

Specifies that an incident is escalated if `Importance` is either `High` or `Critical`.

Rule Constraint Method:

`constraintrule=(field[importance]>"8"):"Assigned User"=valueOf(manager):mandatory`

Specifies that if the item is very important (importance > 8), then assign the item to a manager.

IBPL Constraint Method:

`rule=(field["State"] = "Prioritized"):Priority`

Specifies that if the state is `Prioritized`, then the `Priority` IBPL is visible.

`rule=(field'["Graduated"] = true):Students`

Filters the list of students such that the only values displayed are those where the student's backing item has `Graduated` set to true.

Note

You can specify the options for `--fieldRelationships`, `--addFieldRelationship`, and `--removeFieldRelationship` at the same time. If all three options are specified at the same time, the constraints on the type are modified in the following order:

1. `--fieldRelationships` is executed first and all constraints are replaced by the ones specified here.
2. `--removeFieldRelationship` is executed next and the specified constraints are removed from the ones specified in the `--fieldRelationships` option.
3. `--addFieldRelationship` is executed last and the specified constraints are added to the type.

The options for `--removeFieldRelationship` and `--addFieldRelationship` can be specified multiple times on the command.

Note

If you need to specify a large number of constraints (field relationships), use the `--fieldRelationshipsFile` option.

- `--fieldRelationshipsFile=`*value*

specifies the name of the file containing the constraint. The file should use the same format as the `--fieldRelationships` option, with all constraints on the same line and separated by semi-colons. For information on constraints, see the *PTC Integrity Server Administration Guide*.

Note

If you specify both `--fieldRelationships` and `--fieldRelationshipsFile`, only `--fieldRelationships` is used.

- `--[no]showHistory`

specifies whether to display the item history for all items of the selected type. When you set `--noshowHistory` for an existing type, the item history is always hidden from users when they are viewing or editing items of that type. The setting applies in all Integrity Client interfaces. For example, in the Integrity Client GUI and Web interface, when you set the `--noshowHistory` option, the **History** tab is no longer displayed for items of the selected type. By default, the item history is displayed for newly created types.

- `--[no]showWorkflow`

specifies whether to display the **Workflow**tab to the user in the **Create Issue** view, **Edit Issue** view, and **Issue Detail** view. The **Workflow** tab contains a read-only display of the issue type workflow to the user. Users in the Web interfaces do not have a user preference to override the display of the **Workflow** tab. Users launching the GUI view from the CLI can override the display of the Workflow tab on a per command basis, if workflow for that type is enabled.

- `--phaseField=`*field*

specifies the phase field to display in the **Workflow** tab of the **Create Issue** view, **Edit Issue** view, and **Issue Detail** view. Only a phase field that is specified in the `--visibleFields` option may be specified. This option must be specified with the `--showWorkflow` option.

- `--[no]enableProjectBacking`

Enables the type to back a project, allowing you to create a link between an issue of this type and a project in the `Project` field. By creating a Project issue type and specifying this option, then creating a Project issue and linking it to a specific project (using the `im createissue` command), the power and workflow of projects as issues becomes available. Important metadata and metrics can be recorded in the Project issue, for example, the assigned Project Manager, estimated and actual budgets (using computed fields), and important milestone dates. Linking a Project issue to a project also reduces possible confusion about the details of projects in your database.

Note

- A link between a Project issue and a project is optional.
- If you enable this option, creating issues of this type and linking them to projects is useful only to certain users. You may want to prevent most users from being able to create issues of this type, for example, you can allow only users in the **ProjectManagers** group to create Project issues. To restrict type visibility, see the `--permittedGroups` option.
- Projects and Project issues can be used independent of one another; you do not have to create a Project type to use the `Project` field.
- Multiple types can back projects; however, only one issue may back a given project.
- To enable this option, you must be an administrator for the specified type. This option can be disabled at any time.
- This option cannot be specified unless the `Project` field is specified as a visible field for this type.
- To create the issue that backs a project, you must be an administrator for the specified project and belong to a group that has permission to create issues of that type.
- When you create an issue to back a project, Integrity warns you if there is more than one type that can back a project, displaying a list of types that have this option enabled and that you have "view" and "modify" permissions for. Specify the type that you want to back the project.
- For Admin Staging, you cannot stage issues. Issues that back projects created on a Staging server will not be staged; you must re-create the issues that back projects on the production server. Rules, queries, or computed fields that explicitly mention the issue number that backs a project will not work after they have been transferred from the staging server to the production server.

- `--[no]enableTimeTracking`

  Allows one or more users to allocate time spent working on issues of this type. When enabled, users can specify time entries when they edit an issue. In addition, a `Time Entries` tab is available when you view an issue of this type. You can use time entries to develop metrics (in the form of queries, charts, and reports) that measure the amount of effort spent on projects.

Note

- To create, edit, and delete time entries on behalf of other users, the `TimeTrackingAdmin` ACL permission is required.
- The ability to create, edit, and delete time entries is governed by state-based capabilities. For more information, see the `im createstate` and `im editstate` commands.

- `--permittedAdministrators=u=user1,user2,...;g=group1,group2,...`

  specifies a comma delimited list of users and/or groups that can administer this type. Integrity Type Administrators are allowed to edit and view types. Type Administrators are not allowed to assign themselves as administrators to any other types, or to edit types that they don't administer. A Type Administrator can create fields, but can only edit fields that are referenced by a type they administer. For more information, see the *PTC Integrity Server Administration Guide*.
- `--permittedGroups=group,group,...`

  specifies a comma delimited list of groups that have visibility for this type.
- `--addWordTemplate=name=templateName,path=templatePath[,description=description][,defaultEdit]`

  specifies a template to add to the type for users to use when editing items in Microsoft® Word. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
  - *name*

    specifies the name of the template. The name is visible to users if more than one template is available for selection.
  - *path*

    specifies the path of the template to add to the type. The file must be DOTX format.
  - *description*

    specifies an optional description for the template.
  - *defaultEdit*

    specifies to use the template as the default template for users editing an item or document in Word. Specifying this option pre-selects the template for the user. However, the user may still select a different template if needed. This option can be used to streamline the edit process for users. Only one template per type can have this option enabled. Enabling this option on one template clears it from any other that has it specified.
- `--removeWordTemplate=value`

  specifies a Microsoft® Word template to remove from the type. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
- `--[no]allowDocumentLocks`

  specifies whether document locking is supported for the specified type.
- `--documentLockPolicy=value`

  specifies which users and/or groups are allowed to lock documents of the specified type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to documents of this type are allowed to lock them. Valid values are `userField=<field>`, `groupField=<field>`, *anyone*, `groups=group1,group2,...`.
- `--documentUnlockGroup=group`

  specifies the lock administration group for the type. Members of the specified group can unlock any locked document of the specified type.
- `--[no]allowAdditionalLockFields`

  specifies whether additional fields (beyond significant fields) can be locked for the specified type. By default, only significant fields are affected by document locking.
- `--additionalLockFieldsRule=rule`

  specifies a rule that determines when additional fields (if allowed) are affected by locking. For the rule syntax, see Specifying Rules on the options reference page.
- `--additionalLockFieldsRuleFile=filename`

  specifies a file that contains the additional locks field rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyAdditionalLockFieldsRule=type`

  copies the additional field locks rule of an existing type to the new one being created.
- `--additionalSegmentLockFields=field,field,...`

  specifies the fields on a segment root of the specified type that, in addition to the significant fields, are affected by document locking.
- `--additionalContentLockFields=field,field,...`

  specifies the fields on content nodes of the specified type that, in addition to the significant fields, are affected by document locking.
- `--[no]lockingRequired`

  specifies whether a document of the specified type must be locked before the significant fields (plus any defined additional fields) can be edited.
- `--lockingRequiredRule=rule`

  specifies a rule that defines when documents of the specified type must be locked before the appropriate fields can be edited. For the rule syntax, see Specifying Rules on the options reference page.
- `--lockingRequiredRuleFile=rule`

  specifies a file that contains the rule that defines when locking is required for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyLockingRequiredRule=rule`

  copies the rule that defines when locking is required for an existing type to the new type being created.

- `--hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]`

  specifies the menus and related dialog boxes where creation or insertion of the item type is to be hidden. When an item type is hidden, it can still be created or inserted from the command line. The options are as follows:

  - *none* specifies that the item type is not to be hidden from any menu. This is the default. No other option can be specified along with this option. When item types are to be hidden from menus, multiple options can be specified.
  - *CreateItem* specifies that this item type is to be hidden from menus and dialog boxes for creating items.
  - *CreateRelatedItem* specifies that this item type is to be hidden from menus and dialog boxes for creating related items.
  - *CreateDocument* specifies that this item type is to be hidden from menus and dialog boxes for creating documents.
  - *InsertContent* specifies that this item type is to be hidden from menus and dialog boxes for inserting content in documents.
- *type*

  specifies the name of the type you want to create.

See Also

- Commands: im edittype, im viewtype, im deletetype, im types, im copytype
- Miscellaneous: options

im createuser

creates a user

Synopsis

`im createuser [--name=value] [--description=value] [--image=[none|default|<path>] [--fullName=value] [--email=value] [--notificationRule=rule] [--notificationRuleFile=value] [--copyNotification=[user|group]:<name>] [--[no]active] [--[no]allowNonRealm] [-Ffile|--selectionFile=file] [--user=name] [--hostname=server] [--password=password] [--port=number] [-?|--usage] [-N|--no] [-Y|--yes] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`im createuser` creates a user for workflows and documents. By default, the user must exist in the realm. The create users feature is helpful for assigning work to users who have not yet auto-registered by logging in. For example:

```
im createuser --name="jriley" --description=ServicesManager
```

creates the user *jriley* with a description.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=`*value*specifies the name of the user to create. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.

Note

PTC recommends you do not use commas, colons, or special characters in user names. If you use commas in the user name, you cannot select the user in a multi-valued user field. If you use colons in the user name, the user cannot edit shared system provided objects (charts, queries, reports, and dashboards).

- `--description=`*value*

  specifies a description of the user.
- `--image=[`*none*`|`*default*`|<`*path*`>]`

  specifies whether an image appears for the user.

  `--image=`*none* does not specify an image for the user.

  `--image=`*default* specifies the default "person" image for the user.

  `--image=<`*path*`>` specifies the path and name of a custom image for the user, for example, `c:\images\defect_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--fullName=`*value*

  specifies the full name of the user.
- `--email=`*value*

  specifies the email address of the user.
- `--notificationRule=`*rule*

  specifies the notification rule for this principal. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date field, type *today*. This option can be specified for a date field or a date field configured to display the date and time. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can specify the user's name or "`me`". "`me`" is a symbolic user which refers to the user that the notification rule is created for. This is useful if you want to create a common notification rule and share it with other users.
  - E-mail notification is subject to project, type, and field visibility rules. Only users that have visibility for a given project and type receive e-mail notification for issues related to that project and type. In addition, e-mail notifications include only the fields they have permission to view.

- `--notificationRuleFile=`*value*

  specifies the file that contains the rules to read. For the file content format, see Specifying Rules on the options reference page.
- `--copyNotification=[`*user*`|`*group*`]:<`*name*`>`

  copies the given user/group's notification rules. This command copies the given user/group's notification rule into this user's at the time that the command is run.
- `--[no]active`

  Specifies if the user is actively used in Integrity. This option is useful for removing users that no longer exist in your organization. Specifying `--noactive` prevents the user from being displayed in user drop-down lists; however, the user still appears in existing issue data.
- `--[no]allowNonRealm`

  confirms the non-realm allowance. By default, a user cannot be imported into the Integrity database unless that user exists in the current realm. This option allows you to override the requirement that the user needs to be in the realm. `--noallowNonRealm` is specified by default.

See Also

- Commands: im edituser, im viewuser, im deleteuser, im users
- Miscellaneous: options

im dashboards

displays the list of dashboards

Synopsis

im dashboards [--fields=*field1[:width1],field2[:width2]...*] [--fieldsDelim=*value*] [--[no]showHistory] [--[no]showReferences] [--height=*value*] [--width=*value*] [-x *value*] [-y *value*] [--user=*value*] [--hostname=*value*] [--password=*value*] [--port=*value*] [(-?|--usage)] [(-g|--gui)] [(-F *value* |--selectionFile=*value*)][--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)][(-Y|--yes)] [--[no]batch] [--cwd=*value*] [--forceConfirm=[yes|no]]*[user:]dashboard*

Description

`im dashboards` displays the list of Integrity dashboards. By default, the command displays all dashboards that are currently shared to you.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fields=`*`field1[:width1],field2[:width2]...`*

  specifies the dashboard fields to display and the width of each field in characters. If the output is directed to the GUI, the width is specified in pixels.

  For example,

  ```
  im dashboards --fields=name,description,lastModified --fieldsDelim=, jriley:"Release 2 Overview"
  ```

  displays the name, description, and date of the last modification for the Release 2 Overview dashboard create by `jriley`.

  The dashboard fields you can specify are:

  - `createdBy`

    displays the name of the user who created the dashboard.
  - `description`

    displays a description of the dashboard.
  - `id`

    displays the database ID of the dashboard. This is for PTC - Integrity Support only.
  - `layout`displays the XML representation of the dashboard layout. The layout must conform to a specified format. For more information, see the *PTC Integrity User Guide*.
  - `lastModified`

    displays the date the dashboard was last modified.
  - `name`

    displays the name of the dashboard.
  - `references`

    displays all system provided and user objects that reference the dashboard.
  - `shareWith`

    displays the users and groups that the dashboard is shared with.
  - `isAdmin`

    displays whether the dashboard is a system provided object.
  - `--fieldsDelim=`*`value`*

    specifies the string to be used as a delimiter between fields.
  - *[username:]dashboard*

    specifies the name of the dashboard to view, and the user who created it, for example, `jhoyt:ProjectOverview`. If you are viewing a dashboard you created, you do not have to specify the user name, but you must specify the dashboard name.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the dashboard named `ProjectOverview` created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the dashboard named `jhoyt:ProjectOverview` created by any user.

See Also

- Commands: im copydashboard, im createdashboard, im deletedashboard, im editdashboard, im viewdashboard im rundashboard
- Miscellaneous: options

im deletechart

deletes an existing Integrity chart

Synopsis

```
im deletechart [--[no]confirm] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [user:]chart...
```

Description

Eventually, you may want to delete charts that you no longer use.

You can only delete charts that you created, unless you are an administrator and the chart has been shared to you. If you are not authorized to delete the chart, you are presented with an error message stating that you may not delete the selected chart.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the chart. By default, you are prompted to confirm the deletion of the chart.
- *[user:]chart...*

  specifies the name of the user who the chart belongs to and the chart name, for example, `jhoyt:"Cosmos Critical Defects"`. If you are deleting your own chart and if no other charts exist with the same name, you do not have to specify the user name, but you must specify the chart name. Use spaces to specify more than one user and chart name. This option is useful for administrators who want to delete charts that belong to users who no longer exist.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a chart name. If Integrity fails to find a matching user name and chart name, it searches for a chart name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` chart created by *jhoyt*. If Integrity cannot find the chart and/or user, it searches for the `jhoyt:CosmosDefects` chart created by any user.

See Also

- Commands: im copychart, im createchart, im editchart, im viewchart, im charts
- Miscellaneous: options

im deletecolumnset

deletes a column set

Synopsis

```
im deletecolumnset [--[no]confirm] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-g|--gui)] [(-?|--usage)]
[(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] columnset...
```

Description

Note: Column sets are no longer supported for the Integrity Client. This command can only be used to delete column sets for the MKS Worktray used in integrations. You cannot use the `-g` option with this command. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

`im deletecolumnset` deletes column sets that are no longer needed. You cannot delete or rename the `default` column set. You cannot undo a column set deletion and you cannot delete another user's column set.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm`

  specifies whether to confirm the deletion of the specified column set.
- *columnset...*

  specifies the column set to delete. Use spaces to specify more than one column set.

See Also

- Commands: im copycolumnset, im createcolumnset, im viewcolumnset, im editcolumnset
- Miscellaneous: options

im deletedashboard

deletes an existing Integrity dashboard

Synopsis

```
im deletedashboard [--[no]confirm] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)]
[(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [user:]dashboard...
```

Description

Eventually, you may want to delete dashboards that you no longer use, or if you have too many to manage.

You can only delete dashboards that you created unless you are an administrator and the dashboard has been shared to you. If you are not authorized to delete the dashboard, you are presented with an error message stating that you may not delete the selected dashboard.

Options

This command takes the universal options available to all `im`commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the dashboard. By default, you are prompted to confirm the deletion of the dashboard.
- *[user:]dashboard...*

  specifies the name of the user who the dashboard belongs to and the dashboard name, for example, `jhoyt:"Project Overview"`. If you are deleting your own dashboard and if no other dashboards exist with the same name, you do not have to specify the user name, but you must specify the dashboard name. Use spaces to specify more than one user and dashboard name. This option is useful for administrators who want to delete dashboards that belong to users who no longer exist.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the dashboard named `ProjectOverview` created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the dashboard named `jhoyt:ProjectOverview` created by any user.

See Also

- Commands: im copydashboard, im createdashboard, im editdashboard, im viewdashboard
- Miscellaneous: options

im deletedynamicgroup

deletes the specified dynamic group if it is no longer required

Synopsis

`im deletedynamicgroup [--[no]confirm] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--quiet] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`dynamic group...`*

Description

`im deletedynamicgroup` deletes the specified dynamic group if it is no longer required.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the dynamic group. By default, you are prompted to confirm the deletion of the dynamic group.
- *`dynamic group...`*

  specifies the names of dynamic groups you want to delete.

See Also

- Commands: im createdynamicgroup, im editgroup, im viewdynamicgroup, im dynamicgroups
- Miscellaneous: options

im deletefield

deletes a field

Synopsis

```
im deletefield [--[no]confirm] [--[no]confirmRelationshipFieldPairDeletion] [--user=name] [--hostname=server] [--password=password]
[--port=number] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] field...
```

Description

`im deletefield` deletes a field for workflows and documents. Before deleting the field, any objects referencing the field must first be purged or modified so they no longer reference the field. For a list of objects referencing a field, see im viewfield.

You cannot delete a field that has historical references to that field in the history of one or more items. You must first delete the items that have entries for the field in their histories.

Caution

Deleting a field is permanent. That data cannot be restored after command completion. Ensure that you back up your database before deleting a field.

Caution

Deleting a field of Relationship data type, also deletes its partner field. By default, a confirmer is presented to you before the server deletes the partner field. For more information on such fields, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies if to display a confirmer when deleting a field.
- `--[no]confirmRelationshipFieldPairDeletion`

  specifies if to display a confirmer when deleting a relationship field pair.
- *field...*

  specifies the name of the field to delete.

See Also

- Commands: im createfield, im editfield, im viewfield
- Miscellaneous: options

im deletegroup

deletes a group

Synopsis

```
im deletegroup [--[no]confirm] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] group...
```

Description

`im deletegroup` deletes a group for workflows and documents, if it is safe to do so. You can delete a group only if it has not appeared in the history, that is, there are no issues assigned to this group and no member of this group has submitted or edited an issue.

Note

Deleting a group only removes that group from the Integrity database, it does not remove the group from the realm.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the group. By default, you are prompted to confirm the deletion of the group.
- *`group...`*

  specifies the names of groups you want to delete.

See Also

- Commands: im createGroup, im editGroup, im viewGroup, im groups
- Miscellaneous: options

im deleteissue

deletes an issue from the Integrity database

Synopsis

`im deleteissue [--no[confirm] [--[no]confirmRQ] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--n)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [--quiet] [-g|--gui] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] issue id...`

Description

`im deleteissue` deletes an item from the Integrity database. Only a user with the `DeleteItem` permission can delete an item of any type; however, your administrator can assign the `ModifyDeleteItemRule` permission, allowing a type administrator to set a type rule that specifies which users can delete items of that type. Deleting an item removes the item history and any links to attachments, change packages (change package members associated with the item are not deleted from the database), relationships (including both forward and backward relationships), and some history records from related items. If you delete an item containing relationships, the history of each related items denotes the ID of the deleted items, the user that deleted the items, and the date of the deletion.

Note

- Deleting an item is irreversible.
- The ID of a deleted item is never reused.
- If you are using configuration management with workflows and documents, do not confuse an item's change package members with project or Sandbox members.
- Do not delete document model items. Items that play a role in the document data model are linked in a document's history, so when you delete an item, you change the history of that document as if the item never existed.

  If you do delete a document model item, the following is true:

  - If a document item (segment) is being referenced by a node that exists in another document, the segment item cannot be deleted. For example, document A contains a node that references subdocument B. You cannot delete segment B until you first perform a remove content operation on the node from document A.
  - If a document item contains content (the Contains relationship is not empty), the document item cannot be deleted. Remove the content before attempting to delete the document item.
  - If a node is part of a document (node has a Document ID value), the node item cannot be deleted until it is removed from the document.
  - If the shared item is referenced by any nodes that are still contained by a document, the shared item cannot be deleted.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to confirm the deletion of the item.
- `--[no]confirmRQ`

  specifies whether to separately confirm the deletion of the requirements components of the item, for example, shared items. For more information on requirements, see the *Integrity for Requirements Solution Guide*.

  Note

  The `--forceConfirm` option does not apply to this prompt.

- *`issue id...`*

  specifies the id of the item that you want to delete.

See Also

- Commands: im importissue
- Miscellaneous: options

im deletelabel

removes a label from an issue

Synopsis

`im deletelabel [(-L label|--label=label)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue id...`

Description

`im deletelabel` removes a label from one or more issues.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `-L` *label*
- `--label=`*label*

  identifies a label to delete. Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.

  Note

  Labels that include spaces must be enclosed by quotes.

- *issue id...*

  identifies a specific issue to delete the label from; use spaces to specify more than one issue.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

See Also

- Commands: im addlabel
- Miscellaneous: ACL, diagnostics, options, preferences

im deleteproject

deletes a project

Synopsis

```
im deleteproject [--[no]confirm] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] project...
```

Description

`im deleteproject` deletes a project for workflows and documents. You can delete a project only if neither the project nor any of its children has been referenced in any issue. To delete an idle project that has subprojects, delete its subprojects first.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the project. By default, you are prompted to confirm the deletion of the project.
- *`project...`*

  specifies the projects you want to delete. Include a forward slash (/) when specifying a high level project. For subprojects, include the full path to the subprojects, for example, `/AuroraProject/BorealisProject`.

See Also

- Commands: im createproject, im editproject, im viewproject, im projects
- Miscellaneous: options

im deletequery

deletes an existing Integrity query

Synopsis

```
im deletequery [--[no]confirm] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [user:]query...
```

Description

Eventually, you may want to delete queries that you no longer use, or if you have too many to manage.

You can only delete named queries that you created unless you are an administrator and the query has been shared to you. If you are not authorized to delete the query, you are presented with an error message stating that you may not delete the selected query.

Important: You cannot delete the Quick Query; however, you can clear it in the graphical user interface or Web interface, resetting the default fields to empty values.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the query. By default, you are prompted to confirm the deletion of the query.
- *[user:]query...*
- specifies the name of the user who the query belongs to and the query name, for example, `jhoyt:"Cosmos Critical Defects"`. If you are deleting your own query and if no other queries exist with the same name, you do not have to specify the user name, but you must specify the query name. Use spaces to specify more than one user and query name. This option is useful for administrators who want to delete queries that belong to users who no longer exist.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` query created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the `jhoyt:CosmosDefects` query created by any user.

See Also

- Commands: im copyquery, im createquery, im editquery, im viewquery, im queries
- Miscellaneous: options

im deletereport

deletes an Integrity report

Synopsis

```
im deletereport [--[no]confirm=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [username:]report
```

Description

`im deletereport` deletes an Integrity report.

You can only delete reports that you created unless you are an administrator and the report has been shared to you. If you are not authorized to delete the report, you are presented with an error message stating that you may not delete the selected report.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm=`*value*

  specifies whether to be prompted to confirm the deletion of the report. By default, you are prompted to confirm the deletion of the report.

- *[username:]report*

  specifies the name of the user the report belongs to. If you are deleting a report you created, you do not have to specify the user name, but you must specify the report name. Only the creator of a report or an administrator can delete a report.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a report name. If Integrity fails to find a matching user name and report name, it searches for a report name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` report created by `jhoyt`. If Integrity cannot find the report and/or user, it searches for the `jhoyt:CosmosDefects` report created by any user.

See Also

- Commands: im createreport, im editreport, im copyreport, im viewreport, im runreport, im reports
- Miscellaneous: options

im deletestate

deletes a state

Synopsis

```
im deletestate [--[no]confirm] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] state...
```

Description

`im deletestate` deletes a state for workflows and documents. You can only delete a state if it is idle, that is, if no one has used this state in any issue.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the state. By default, you are prompted to confirm the deletion of the state.
- *`state...`*

  specifies the names of states you want to delete.

See Also

- Commands: im createstate, im editstate, im viewstate, im states
- Miscellaneous: options

im deletetrigger

deletes an event trigger

Synopsis

`im deletetrigger [--[no]confirm] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`trigger...`*

Description

`im deletetrigger` deletes an event trigger for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the trigger. By default, you are prompted to confirm the deletion of the trigger.
- *`trigger...`*

  specifies the names of the event triggers you want to delete.

See Also

- Commands: im createtrigger, im copytrigger, im edittrigger, im viewtrigger, im runtrigger, im triggers, im echo
- Miscellaneous: options

im deletetype

deletes an item type

Synopsis

`im deletetype [--[no]confirm] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`type...`*

Description

`im deletetype` deletes an item type. You can delete a type as long as no issues of that type have been created.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the type. By default, you are prompted to confirm the deletion of the type.
- *`type...`*

  specifies the names of the types you want to delete.

See Also

- Commands: im createtype, im edittype, im viewtype, im types, im copytype
- Miscellaneous: options

im deleteuser

deletes a user

Synopsis

`im deleteuser [--[no]confirm] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`user...`*

Description

`im deleteuser` deletes a user for workflows and documents, if it is safe to do so. You can delete a user only if that user has not appeared in the history, that is, there are no issues assigned to the user and the user has not submitted or edited an issue, change package, trigger, or any other administrative object.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether to be prompted to confirm the deletion of the user. By default, you are prompted to confirm the deletion of the user.
- *`user...`*

  specifies the users you want to delete.

See Also

- Commands: im createuser, im edituser, im viewuser, im users
- Miscellaneous: options

im diffsegments

compares the differences between document selections

Synopsis

`im diffsegments [(-?|--usage)][--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...`*`)] [--[no]hideItemsWithoutDifferences][--[no]hideFieldsWithoutDifferences] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--hostname=`*`server`*`] [--password=`*`password`*`] [--[no]persist] [--port=`*`number`*`] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|gui.actions|default]`*`] [--user=`*`name`*`] `*`DocumentID DocumentID`*

Description

`im diffsegments` compares the differences between one document at two different points in time, two versions of the same document (and optionally at different points in time), or two documents that share a common branch ancestor ID (and optionally at different points in time), such as a versioned document compared with a document that was branched.

For example:

```
im diffsegments 851@label:Draft1 851@label:Draft2
```

returns all differences between a single document at two points in time based on labels.

The following are key considerations when using this command:

- This command is only supported with the `-g` or `--gui` option. When you use the `-g` or `--gui` option, Integrity displays the View Document Differences window where you can specify a source and target document to compare. The Document Difference view displays all differences between the document selections side-by-side.
- This command accepts two document IDs only. A single document ID or more than two document IDs are not supported.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- *DocumentID DocumentID*

  identifies the two documents selections you want to compare. Each document ID must correspond to a segment (e.g., a root node of a document). Use spaces between the two IDs, for example `34 23`. Can be one of:

  - an identifier (i.e., `851`) which corresponds to the current document as of now.
  - a historical document ID which corresponds to a baseline document with a given revision, label, branch time and date, edit date, or time stamp. For example: `851@label:labelname`.
  - an identifier for a branched document.
  - a versioned document when document versioning is enabled. To type the ID of a versioned document, use the format *Live Item ID-major.minor*. For example `184-1.2`.

- [`--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...`*`)]`

  specifies the fields and their respective widths to be compared and displayed in the Document Difference view. Your administrator defines the available fields. Fields can include `Contains`,`Summary`, and others. Use commas to specify more than one field.

- `[--[no]hideItemsWithoutDifferences]`

  filters the Document Difference view to hide items that have no field or document structure differences.

- `[--[no]hideFieldsWithoutDifferences]`filters the Document Difference view to hide additional comparison fields that contain no differences.

Diagnostics

See the `diagnostics` reference page for possible exit status values.

Preferences

Using `im setprefs` or `im viewprefs`, you are able to set or view the preference keys for this command.

See Also

- Commands: si diff, si difffiles, im viewsegment
- Miscellaneous: ACL, diagnostics, options, preferences

im disconnect

disconnects from the Integrity Server

Synopsis

```
im disconnect [--[no]confirm] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

`im disconnect` disconnects the client connection to the host Integrity Server.

Note

When disconnecting a connection that is the current connection, all open client views close. All new views use the connection specified in the Integrity preferences, or an existing connection, as the new current connection.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]confirm`

  controls whether to implement the Integrity Server disconnection confirmation policy.

See Also

- Commands: im connect, im servers
- Miscellaneous: options

im dynamicgroups

displays the list of dynamic groups

Synopsis

`im dynamicgroups [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--[no]confirm] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--quiet] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] dynamic group...`

Description

`im dynamicgroups` displays the list of dynamic groups. By default, all dynamic groups are selected to be displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  specifies the dynamic group fields to display and the width of each field in pixels.

  The dynamic group fields you can specify are:

  - `membership`

    displays only the projects the project administrator is assigned to.

    To display projects that do not have any groups and users as members of the dynamic group, specify the `nomembers` keyword, for example, `--membership=/Project=nomembers`.

    Note

    Caution: If you are a project administrator, the `im dynamicgroups --fields=membership` command displays a subset of the projects in your Integrity configuration. If a super administrator uses that list of projects when specifying `im editdynamicgroup --membership`, the membership for the projects that the project administrator does not have permission to view are removed.

  - `id`

    displays the ID of the dynamic group in the database. This option is for PTC - Integrity Support only.

  - `name`

    displays the name of the dynamic group.

  - `description`

    displays a description of the dynamic group.

  - `image`

    displays whether there is an image, and if so, whether it uses the default dynamic group image or a custom image.

  - `references`

    displays a list of object references.

- `--fieldsDelim=value`

  specifies the character to use to separate dynamic group fields, for example, "*" or ",". The following is an example of dynamic groups output using * as a field delimiter value:

  QA Scripts Group*Groups and individuals permitted to work in the QA Scripts project

- *dynamic group...*

  specifies the names of the dynamic groups.

See Also

- Commands: im createdynamicgroup, im editdynamicgroup, im viewdynamicgroup, im deletedynamicgroup
- Miscellaneous: options

im echo

echoes a string to UI

Synopsis

```
im echo [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...
```

Description

`im echo` echoes a string to the appropriate user interface when used in an event trigger for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- *`string...`*

  specifies a string to display in the appropriate user interface.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers
- Miscellaneous: options

im editchart

edits an existing Integrity chart

Synopsis

```
im editchart [--bgColor=value] [--chartFootnote=value] [--chartTitle=value] [--dataColors=value] [--descriptionFont=value] [--
[no]displayDescription] [--[no]displayLegend] [--[no]displayLabels] [--endDate=value] [--fieldFilter=field=[value,value,...] [--
fieldValues=value] [--footnoteFont=value] [--graphStyle=
[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]] [--groupingValues=value] [--[no]is3D]
[--[no]isAutoColors] [--[no]isShowZeroFieldCount] [--[no]isShowZeroGroupingCount] [--legendBgColor=value] [--legendPosition=
[Right|Bottom|Left|Top]] [--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]] [--legendTitle=value] [--
outlineColor=value] [--projectedTrendExpressions=value] [--query=[user:]query] [--startDate=value] [--numberOfSteps=value] [--
titleFont=value] [--trendStep=[Hour|Day|Week|Month|Quarter|Year]] [--[no]useIssueDefinedOrigin] [--[no]xReverse] [--[no]xShowGrid] [-
-[no]xShowTitle] [--yLabelRotation=[Horizontal|VerticalUp]] [--[no]yReverse] [--[no]yShowGrid] [--[no]yShowTitle] [--
description=value] [--name=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--
sharedAdmin] [--[no]confirmSharedAdmin] [--computations=value] [--startDateField=field] [--runDateIsEndDate] [--[no]deltasOnly] [--
issueIdentifier=value] [--[no]displayShapesForLineGraphs] [--[no]swapRowsAndColumns] [--[no]displayRowTotals] [--
[no]displayColumnTotals] [--rangeDefinitions=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--
usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--
no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [user:]chart
```

Description

`im editchart` edits the properties of an Integrity chart. Integrity displays a chart selection dialog box when you use the `-g` or `--gui` option.

For more information on charts, refer to the *PTC Integrity User Guide*.

For example,

```
im editchart --name="Release 3.0 Docs Issues By User" --query="Release 3.0 Docs Issues" "Release 2.0 Docs Issues By User"
```

edits the existing Release 2.0 Docs Issues By User chart to change its name and the query that it is based on.

Note the following:

- A chart can be edited by the user who created it. Principals (users and groups) that a chart is shared to can edit it if they have edit permissions assigned to them by the chart creator. A chart can only be deleted by the user who created it or by an administrator.
- You cannot create or edit a query while creating a chart.
- Charts can do more than just display field information in a graphical format. You can also perform arithmetic calculations between numeric fields, displaying the values in the chart. For example, you can calculate the average for a group of field values or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- All charts are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or see your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.

For trend charts, the following are valid option combinations:

--startDate

--endDate

--startDate

--numberOfSteps (positive value)

--startDate

--runDateIsEndDate

--startDateField*

--numberOfSteps (positive value

--numberOfSteps (negative value)

--endDate

--numberOfSteps (negative value)

--runDateIsEndDate

*Only applies to Issue Fields Trend charts.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--chartFootnote=`*value*

  specifies the footnote text of the chart.
- `--chartTitle=`*value*

  specifies the title of the chart.
- `--titleFont=value`

  specifies the font to be used for the chart title. Use the following format: *name,style,size*, where *style* is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic, for example, `helvetica,1,10`. When the chart is run, if the specified font cannot be found, Integrity uses a substitute font.
- `--descriptionFont=`*value*

  specifies the font to be used for the description. Use the following format: *name,style,size* format, where *style* is 0 for plain, 1 for bold and 2 for italic, for example, `helvetica,1,10`.
- `--[no]displayDescription`

  specifies whether to display the chart description.
- `--trendStep=[Hour|Day|Week|Month|Quarter|Year]`

  specifies the interval for each point on a trend or issue fields trend chart graph.
- `--[no]useIssueDefinedOrigin`

  specifies whether to use the start date defined in an issue field for an issue fields trend chart.
- `--startDate=`*value*

  specifies the start date for trend or issue fields trend charts. To specify a date and time, type *MM/dd/yyyy h:mm:ss [AM|PM]*.

  Other acceptable date formats include:
  MM/dd/yyyy h:mm:ss a zMM/dd/yyyy h:mm:ss.SSS a zMM/dd/yyyy h:mm:ss aMM/dd/yyyy h:mm:ss.SSS aMM/dd/yyyy
- `--endDate=`*value*

  specifies the end date for trend or issue fields trend charts. To specify a date and time, type *MM/dd/yyyy h:mm:ss [AM|PM]*. See the `--startDate`**=***value* option for additional date and time formats.
- `--description=`*value*

  specifies a short description for the chart.
- `--name=`*value*

specifies the new name of the chart. Names may be a maximum of 100 characters and cannot contain square brackets.

- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`

  specifies the users and groups that can use and modify the chart. Your administrator defines users and groups.

- `--fieldFilter=field=[value,value,...]`

  specifies how field filters can be applied to the chart when it is run. The first component of the value is the field name. Currently, only project field filters are supported. The second component specifies the project(s) that you want to filter the chart data by when it is run. For example, `--fieldFilter="Project=/Project1"` filters for issues that have a value of `Project1` in the `Project` field. If you do not specify a value, Integrity filters for issues with a value of `Unspecified` in the `Project` field.

Note

You can also define project filters for dashboards. Depending on how you design your dashboard, when a chart is run through a dashboard, the dashboard's project filter can override the chart's project filter.

- `--fieldValues=value`

  specifies the field, field values and aliases used by the chart. For example: `--fieldValues=Type=Documentation, Development[Feature Request, Bug]` would include issues that have a `Type` field with a value of `Documentation`, `Feature` or `Bug`, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

  Use * to include all field values, and + to automatically include all future field values. For example: `--fieldValues=Type=*, +, Development[Feature Request, Bug]` would include all current values and any future values for the `Type` field, with `Feature` and `Bug` types combined on the chart under the alias `Development`.

  For more information on specifying chart values, see the *PTC Integrity User Guide*.

- `--footnoteFont=value`

  specifies the font to use for the footnote. Use the following format: `name,style,size` format, where `style` is 0 for plain, 1 for bold, 2 for italic, and 3 for bold italic, for example, `helvetica,1,10`.

- `--groupingValues=value`

  specifies the field, field values, and aliases to use to group the data in the chart. For example: `--groupingValues=State=Submit, In Work[In Progress, In Development]` would group chart data into separate components for `Submit` and `In Work`, with `In Work` being a combination of the `In Progress` and `In Development` states.

  Use * to include all field values, and + to automatically include all future field values. For example: `--groupingValues=State=*, +, In Work[In Progress, In Development]` would group chart data into separate components for all current values and any future values for the State field, with `In Work` being a combination of the `In Progress` and `In Development` states.

  For more information on specifying chart values, see the *PTC Integrity User Guide*.

- `--projectedTrendExpressions=value`

  When creating or editing an item fields trend chart, you can specify whether to display a projected trend for one of the displayed item field series, thus supporting the use of `burn-down charts` for planning work remaining within a fixed duration project. The projected trend displays a line between a starting value corresponding to the starting date and the end value corresponding to the end date. In addition, if you choose to display the projected trend, you can also display the actual trend as a line between the last actual series value to the end value corresponding to the end date.

  This option specifies attributes for a projected trend graph in the item fields trend chart, where *value* consists of the following attributes separated by a colon:

  chartedExpression is the expression (numeric field) to chart. This value is mandatory.

  startValueExpression is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the historic item field value. This value is mandatory.

  endValueExpression is a field expression (field name) or numeric constant (numeric value). For a field expression, the numeric value is the "now" item field value. This value is mandatory.

  projectedTrendLabel is a text string used for the chart legend and tooltips.

  showUpdatedTrend displays the projected trend in the chart. Specify `true` or `false` (default).

  updatedTrendLabel is a text string used for the chart legend and tooltips.

  Tip: To clear the value for `--projectedTrendExpressions=value`, specify an empty string.

  For example:

  `--projectedTrendExpressions=Effort:0:"Effort":"Planned Effort":true:"Unplanned Effort"`

- `--query=[user:]query` specifies the name of the query that the chart is based on.

Note

If the chart is a shared admin object, an admin query is required.

- `--graphStyle=[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]`

  specifies the graph style used of the chart.

- `--dataColors=value`

  specifies the custom data colors to be used using the RGB color model. For example:

  'R,G,B;R,G;R,G,B'

  where R,G and B are within the range 0-255.

  If the chart has more data points than the data colors you specify, the colors are repeated. If the

  --[no]isAutoColors

  option is true, the colors specified here are ignored.

Note

This option is invalid for table style graphs.

- `--bgColor=value`

  specifies the background color of the chart using the RGB color model. For example:

  'R,G,B'

  where R,G and B are within the range 0-255.

Note

This option is invalid for table style graphs.

- `--[no]displayLegend`

  specifies whether to display the chart legend.

Note

This option is invalid for table style graphs.

- `--[no]displayLabels`

  specifies whether to display labels for values in the chart. If you select a pie graph style, this option is automatically selected. `--nodisplayLabels` is the default option.

Note

This option is invalid for table graphs.

- `--[no]is3D`

  specifies whether to display bar and pie graphs in 3D.

Note

This option is invalid for table style graphs

- `--[no]isAutoColors`

  specifies whether to use the default chart colors. If false, you must provide colors through the data colors option.

Note

This option is invalid for table style graphs.

- `--[no]isShowZeroFieldCount`

  specifies whether to include empty field values in the chart.
- `--[no]isShowZeroGroupingCount`

  specifies whether to include empty grouping values in the chart.
- `--legendBgColor=`*`value`*

  specifies the background color for the chart legend using the RGB color model. For example:

  'R,G,B'

  where R,G and B are within the range 0-255

Note

This option is invalid for table style graphs.

- `--legendPosition=[Right|Bottom|Left|Top]`

  specifies the legend position in relation to the graph.

Note

This option is invalid for table style graphs.

- `--legendTitle=`*`value`*

  specifies the title for the chart legend.

Note

This option is invalid for table style graphs.

- `--outlineColor=`*`value`*

  specifies the outline color of the graph using the RGB color model. For example:

  'R,G,B'

Note

This option is invalid for table style graphs.

- `--xLabelRotation=[Horizontal|VerticalDown|VerticalUp|45Down|45Up]`

  specifies the rotation of the horizontal axis labels for the chart.

Note

This option is invalid for table style graphs.

- `--[no]xReverse`

  specifies whether the chart uses a horizontal axis with a reverse orientation (left).

Note

This option is invalid for table style graphs.

- `--[no]xShowGrid`

  specifies whether to display horizontal grid lines.

Note

This option is invalid for table style graphs.

- `--[no]xShowTitle`

  specifies whether to display the title for the horizontal axis.

Note

This option is invalid for table style graphs.

- `--yLabelRotation=[Horizontal|VerticalUp]`

  specifies the rotation of the vertical axis labels for the chart.

  Note

  This option is invalid for table style graphs.

- `--[no]yReverse`

  specifies whether the chart uses a vertical axis with a reverse orientation (down).

  Note

  This option is invalid for table style graphs.

- `--[no]yShowGrid`

  specifies whether to display vertical grid lines.

  Note

  This option is invalid for table style graphs.

- `--[no]yShowTitle`

  specifies whether to display the title for the vertical axis.

  Note

  This option is invalid for table style graphs.

- `--sharedAdmin`

  specifies the chart as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.

- `--[no]confirmSharedAdmin`

  specifies whether to confirm the conversion of the chart to a system provided object.

- `--computations=`*expression:name:pattern:axis name:minRangeValue:maxRangeValue:tickUnitValue*

  specifies an expression and numeric axes attributes.

  Note the following about specifying numeric axes attributes:

  - If you specify one set of numeric axes attributes (minimum range, maximum range, and tick unit), these attributes are specified for the X and Y axes. For XY (scatter) charts, PTC recommends against setting individual numeric axes attributes for the X and Y axes.
  - For bubble charts, PTC recommends against specifying numeric axes attributes because they override the calculated values provided by the underlying expression and users will have to zoom in/out to properly view chart values.

  expression specifies an aggregate expression for a distribution chart, a computed expression for an issue fields chart, or a numeric field for an issue fields trend chart. For information on creating expressions, see the *PTC Integrity Server Administration Guide*.

  name specifies the label name for the aggregate expression, computed expression, or numeric field as you want it to appear in the chart. If you do not define a label, the aggregate expression, computed expression, or numeric field name displays.

  pattern specifies the display pattern for the value of the aggregate expression, computed expression, or numeric field value.

  axis name specifies a name for the numeric axis as you want it to appear in the chart.

  minRangeValue specifies the minimum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

  maxRangeValue specifies specifies the maximum range to display numeric field values in the chart. If you do not specify a range, a default range displays in the chart.

  tickUnitValue specifies the units that display on the numeric axis. For example, if you specify a minimum range of 0, a maximum range of 100, and a tick unit of 10, the numeric axis displays 0, 10, 20, 30, 40, and so on up to 100.

  Note

  Field names in expressions and expression labels with colons must be enclosed by escaped double quotes, and the whole computation must be enclosed in double quotes, for example,

  ```
  --computations="\"Actual Dev Time\":\"Hours: Development\":pattern:..."
  ```

- `--startDateField=`*field*

  specifies the date field containing the date you want to use as the start date for each issue in an issue fields trend chart.

- `--numberOfSteps=`*value*

  specifies the trend chart's time span. If this option is specified, the chart's end date is determined by the specified step type multiplied by the specified number of steps.

  Note

  You cannot have more than 500 steps in a trend chart.

  To specify an interval in the past, use a negative value.

- `--runDateIsEndDate`

  specifies that the chart's run date is the end date. This option replaces the `--endDate`=value option.

- `--[no]deltasOnly`

  specifies whether to display only the differences between the current and previous values of the reported numeric fields in an issue fields trend chart.

- `--issueIdentifier=`*value*

  specifies the field that you want to identify issues by in an issue field or issue fields trend chart. For example, if you specify `--issueIdentifier={Project}`, each issue in the chart is identified by the value of the `Project` field.

  If you want to add text that precedes the specified field, type it before the field, for example,`--issueIdentifier=Project:{Summary}`. The chart then identifies each issue by displaying

  Project:

  Summary field value

- `--[no]displayShapesForLineGraphs`

  specifies whether to display shapes in a line graph chart. The shapes in the chart represent data, allowing you to more easily differentiate the data in the chart.
- `--[no]swapRowsAndColumns`

  specifies whether to invert the appearance of columns and rows in a table chart.
- `--[no]displayRowTotals`

  specifies whether to display row totals in a table chart.
- `--[no]displayColumnTotals`

  specifies whether to display column totals in a table chart.
- `--rangeDefinitions=`*value*

  specifies range definitions for computed expressions included in a table chart, where *value* consists of the following attributes: *expression name;range field name;range label:lower limit:upper limit:icon:background color:text color:text style:display format; lower limit:upper limit:.....;extend to axis* .

  expression name specifies the name of the computed expression that the range definition applies to. An expression name is mandatory and must be a valid expression in the chart. For column or row totals, valid expression names are `-Column Totals-` and `-Row Totals-`. For distribution charts containing multiple computed expressions, row or column totals must be followed by the expression name.

  range field name specifies a valid field name if you want to relate the range definitions to an existing range field. For one chart range, specify an empty string as the range field name. If a valid range field name is defined for each range, define a range label, background color, text color text style, and display format. For individual range definitions, define a range label, lower limit, upper limit, icon, background color, text color text style and display format for each range.

  range label specifies a label for the range.

  lower limit specifies the lower limit of the range. If a lower limit is not specified, `-Infinity` is automatically specified.

  upper limit specifies the upper limit of the range. If an upper limit is not specified, `Infinity` is automatically specified.

  Note

  A numeric value must be specified for a defined range; range intersections are invalid. For example, the following ranges are invalid: `0 - 5` and `4 - 8`, or `0 - 5` and `5 - 10`. For an integer field, an acceptable range would be `0 - 5` and `6 - 10`. For a floating point field, an acceptable range would be `0 - 5` and `5.01 - 10`.

  ---

  icon specifies an image file representing the range category. This is optional.

  background color specifies the background color of the range using the RGB color model, for example, 'R,G,B', where R, G, and B are within the range 0-255.

  text color specifies the text color of the range using the RGB color model, for example, *'R,G,B*', where *R,G* , and *B* are within the range 0-255.

  text style specifies the text style. Available text styles are `plain`, `bold`, `italic`, `bolditalic`, or `defaultplain`.

  display format specifies how to display the range in the table chart. Available options are `value`, `iconvalue`, `icon`, `label`, `iconlabel`, or `blank`.

  extendToAxis specifies whether to apply the range definition associated with a computed expression to all computed expressions in the chart. This option can be `false` or `true`. By default, `false` is specified.

  Note the following:

  - You cannot specify a range for the `Count` expression.
  - You can specify a range for each computed expression; however, only one computed expression can specify the *extendToAxis* option.
  - If a table cell contains a display definition that conflicts with the *extendToAxis* option of another table cell, both table cells display the *background color* option of the table cell with the enabled *extendToAxis* option.
  - *[user:]chart*specifies the name of the chart to edit, and the user who edited that chart. This is useful when multiple users have the same name for a chart.

    Note

    Integrity initially assumes that text before the colon (:) is a user name and text after it is a chart name. If Integrity fails to find a matching user name and chart name, it searches for a chart name matching the exact text. For example, if you type *jhoyt:CosmosDefects*, Integrity searches for the *CosmosDefects* chart created by *jhoyt*. If Integrity cannot find the chart and/or user, it searches for the *jhoyt:CosmosDefects* chart created by any user.

See Also

- Commands: im copychart, im deletechart, im createchart, im viewchart, im runchart
- Miscellaneous: options

im editcolumnset

edits the properties of a column set

Synopsis

```
im editcolumnset [--fields=field,field,...] [--name=value] [--[no]sortAscending] [--sortField=field] [--hostname=value] [--
port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=
[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] columnset
```

Description

Note

Column sets are no longer supported for the Integrity client. This command can only be used to edit column sets for the MKS Worktray used in integrations. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

`im editcolumnset` make changes to the settings of existing column sets. You cannot delete or rename the `default` column set. Editing the default column set does not affect the default column set of other users.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fields=field,field,...`

  specifies the issue fields to be included in the column set, for example, `ID`, `Type`, `Summary`, `State`, `Project`. Your administrator defines the fields in an issue type. Use commas to specify more than one field.
- `--name=value`

  specifies the new name of the column set. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--[no]sortAscending`

  specifies whether to sort the specified field in ascending or descending order.
- `--sortField=field`

  specifies the field to sort issues by.
- *columnset*

  specifies the column set to edit.

See Also

- Commands: im copycolumnset, im createcolumnset, im viewcolumnset, im deletecolumnset im columnsets
- Miscellaneous: options

im editdashboard

edits an Integrity dashboard

Synopsis

`im editdashboard[--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--description=value] [--name=value] [--fieldFilterConstraint=field:[Open[:value,value,...] | [Fixed[:value,value,...]|[Restricted[:value,value,...[:value,value,...]]] | [None] [--layout=value] [--layoutFile=file] [--sharedAdmin] [--[no]confirmSharedAdmin] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]][user:]dashboard`

Description

`im editdashboard` edits the properties of an Integrity dashboard. You may only edit one dashboard at a time, and only dashboards created by you are available for editing. For more information on dashboards, refer to the *PTC Integrity User Guide*.

For example,

```
im editdashboard --name="Docs Weekly Status" --fieldFilterConstraint=project:Open "Docs Weekly Status Release 2.0"
```

edits the `Docs Weekly Status Release 2.0` dashboard to change its name to `Docs Weekly Status` and enable it to be filtered by any project value.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`

  specifies the users and groups that can use and modify the dashboard. Your administrator defines users and groups.
- `--name=value`

  specifies the name of the dashboard. Names may be a maximum of 100 characters and cannot contain square brackets.

  `Note`: If you do not specify a different name for the new dashboard, Integrity adds "Copy of" as a prefix to the query name, for example, `Copy of Project Overview`.
- `--description=value`

  specifies a short description for the dashboard, for example, "`Overview of current project status`".
- `--fieldFilterConstraint=field:[Open[:value,value,...] | [Fixed[:value,value,...] | [Restricted[:value,value,...[:value,value,...]]] | [None]`

  specifies how field filters can be applied to the dashboard at runtime. The first component of the value is the field name. Currently, only project field filters are supported. The second component is the filter type.

  `Open`specifies that all projects can be selected as filter values when the dashboard is run. You can also specify default filter values to apply.

  `Fixed`specifies that when the dashboard is run it will be filtered by the specified values. You cannot change this filter at runtime.

  `Restricted` specifies that when the dashboard is run you can select any of the specified filter values. You can also specify default filter values to apply.

  `None` specifies that when the dashboard is run, no project filter displays.

Note

Depending on how you design your dashboard layout, the dashboard filter may not be applied to chart, report, report link or query link dashboard components. If this option is not specified, the `Open` filter is used.

- `--layoutFile=value`

  specifies the file that contains the complete definition of the dashboard layout.
- `--layout=value`

  the XML representation of the dashboard layout. The layout must conform to a specified format. For more information, see the *PTC Integrity User Guide*. This setting is optional.
- `--sharedAdmin`

  specifies the dashboard as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

Note

Once a user object is converted to a system provided object, you cannot revert it to a user object again.

If the dashboard you are copying is an admin dashboard, the `--sharedAdmin` option is not set in the copy.

- `--[no]confirmSharedAdmin`

  specifies whether to confirm the conversion of the dashboard to a system provided object.
- *[user:]dashboard*

  specifies the name of the dashboard to edit, and the user who edited that dashboard. This is useful when multiple users have the same name for a dashboard.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the `ProjectOverview` dashboard created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the `jhoyt:ProjectOverview` dashboard created by any user.

See Also

- Commands: im copydashboard, im deletedashboard, im createdashboard, im viewdashboard, im dashboards, im rundashboard
- Miscellaneous: options

im editdynamicgroup

edits a dynamic group

Synopsis

`im editdynamicgroup [--membership=proj1=u=user1,user2,..:g=group1,group2,..;proj2=...] [--projectmembership=project=inherit|nomembers|per-project-membership] [--description=value] [--image=[none|default|<path>] [--name=value] [--user=name] [--projectmembership=project=inherit|nomembers|per-project-membership] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] dynamic group`

DESCRIPTION

`im editdynamicgroup` edits a specified dynamic group. You can edit the following dynamic group details: name, image, description, and membership. For example:

```
im editdynamicgroup --membership=/Support Services=g=Services
```

changes the membership of the *Services* dynamic group to the *Support* project.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--membership=proj1=u=user1,user2,..:g=group1,group2,..;proj2=...`

  assigns the membership for the dynamic group.

Caution

If you are a project administrator, the `im dynamicgroups --fields=membership` command displays a subset of the projects in your Integrity configuration. If a super administrator uses that list of projects when specifying `im editdynamicgroup --membership`, the membership for the projects that the project administrator does not have permission to view are removed.

Note

Specifying the `u=user1,user2`, or `g=group1,group2`, option implies that the dynamic group is assigned to a root project. If the dynamic group is assigned to a child project and you want the parent project permissions to apply, do not specify these options.

where:*proj1*

specifies the project you want the dynamic group to apply to, for example, the `/SourceCode` project. This command processes only the projects that the project administrator is assigned to.

To indicate that a project does not have any groups and users as members of the dynamic group, specify the `nomembers` keyword, for example, `--membership=/Project=nomembers`.

To inherit the membership from the parent project to the dynamic group, specify the `inherit` keyword, for example, `--membership=/Project=inherit`.

Note

The nonmembers project is a top level project, and so cannot inherit permissions from any other project. Furthermore, no other project can be a subproject of the nomembers project.

- `u=user1,user2`

  specifies the users in the dynamic group.
- `g=group1,group2`

  specifies the groups in the dynamic group.
- `--image=[none|default|<path>]`

  specifies whether an image appears for the dynamic group.
  - `--image=none`

    does not specify an image for the dynamic group.
  - `--image=default`

    specifies the default image for the dynamic group.
  - `--image=<path>`

    specifies the path and name of a custom image for the dynamic group, for example, `c:\images\dynamic_group_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=value`

  specifies a description of the dynamic group.
- `--name=value`

  specifies the name of the dynamic group you want to edit. Names can be a maximum of 100 characters and cannot contain square brackets. Use this option when you want to change the name of the a dynamic group.
- `--projectmembership=project=inherit|nomembers|per-project-membership`

  specifies the membership for a project that the project administrator is assigned to.

  where:
  - *project*

    specifies the project you want to assign membership to.
  - `inherit`

    specifies to inherit the membership from the parent project to the dynamic group.
  - `nomembers`

    specifies that the project does not have any groups and users as members of the dynamic group.
  - *per-project-membership*

    specifies the group and user membership for the project, where, `per-project-membership` is in the form\u00a0`user-list|group-list|user-list:group-list`.

Note

Specifying users, but no groups removes any existing groups. Similarly, specifying groups, but no users removes any existing users.

See Also

- Commands: im createdynamicgroup, im viewdynamicgroup, im deletedynamicgroup, im dynamicgroups
- Miscellaneous: options

im editfield

edits the properties of a field

Synopsis

`im editfield [--overrideForType=type] [--removeOverride=description,relevanceRule,editabilityRule,ranges,displayPattern,phases,defaultColumns,default,substituteParams,displayAs,incomingTraceProvider] [--[no|confirm]removeDescriptionOverride] [--[no]confirmMultiValued] [--[no]multiValued] [--defaultBrowseQuery=[user:]query] [--name=value] [--description=value] [--position=<number>] [--default=value] [--defaultColumns=value] [--displayAs=[default|checkbox]] [--displayTrueAs=value] [--displayFalseAs=value] [--min=value][--max=value] [--[no]displayAsProgress] [--[no]displayAsLink] [--associatedField=fieldName] [--backedBy=value] [--backingStates=value] [--backingTextField=field] [--backingIBPLTextFormat=value] [--backingType=type] [--[no]sortIBPLDescending] [--sortIBPLField=field] [--backingfilter=<querydefinition>] [--computation=value] [--[no]allowComputationUpdatesOnVersion] [--[no]staticComputation] [--storeToHistoryFrequency=[never|daily|weekly|monthly|delta]] [--addEntry=value] [--editEntry=value] [--deleteEntry=value] [--loggingText=[none|mostRecentFirst|mostRecentLast]] [--picks=text:value:image,...] [--maxLength=value] [--suggestions=text1,text2,text3,...] [--relevanceRule=rule] [--relevanceRuleFile=filename] [--editabilityRule=rule] [--editabilityRuleFile=filename] [--copyRelevanceRule=field] [--copyEditabilityRule=field] [--allowedTypes=type:type,type,...[;...]] [--addLinkFlags=name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];...] [--[no]cycleDetection] [--query=[user:]query] [--correlation=src-field:dest-field,...] [--displayRows=value] [--displayStyle=value] [--enableLinkFlags=value] [--disableLinkFlags=value] [--displayName=value] [--[no]trace] [--displayPattern=value] [--showDateTime] [--relationshipBrowseStyle=browseStyle=[query|finder],queryOption=[allQueries|specificQueries],specificQueriesList=query::query::..., finderQuery=[user:]query,outlineFormat=value,relationshipstoFollow=relationship::relationship::...] [--[no]richContent] [--[no]showTallRows] [--[no]textIndex] [--[no]substituteParams] [--defaultAttachmentField=field] [--[no]computeNow] [--[no]forceRecompute] [--user=name] [--hostname=server][--password=password] [--port=number] [--editLinkFlags=existingName=value|name=name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];...] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]][--ierDefaultColumns=[server,title,name...]][--incomingTraceProvider=value]field`

Description

`im editfield` edits the properties of a field for workflows and documents. For the following standard fields, only the name, position, and description can be modified: ID, Type, Created By, Created Date, Modified By, Modified Date, Signed By, and Signature Comment. For the following standard fields, only the name, position, description, editability, and relevance can be modified: Summary, State, Assigned User (can modify to allow mult-values), Project, Assigned Group (can modify to allow mult-values). For example:

```
im editfield --hostname=abcFinancial --user=jriley\u00a0 --storeToHistoryFrequency=weekly "Days_In_Phase"
```

changes the *Days_In_Phases* field to calculate on a weekly basis.

Note

Important: Inactive values are not accepted in as field values. If you specify an inactive value for a user, group, or project field with a default value, the existing value is cleared. For multi-valued user and group fields, the entire field value is cleared even if you specified one inactive value.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--overrideForType=type`

  sets type-specific overrides for the field. The customized value supersedes the global settings for the field attribute when referenced through the selected type or an issue of the selected type. You can only override the attributes for description, relevance, editability, default, minimum, and maximum. The value is the name of the type you want to apply the override to.

  Note

  If you override one of the following, all three values are overridden: default value, minimum value, maximum value, ranges, phases, display patterns, and default columns. If you don't specify all these options, the following values are used: an existing override value, the global setting, or null.

  - `--[no|confirm]removeDescriptionOverride`

    confirms removing the translations in all the locales for the overridden description.

  - `--removeOverride=description,relevanceRule,editabilityRule,ranges,displayPattern,phases,defaultColumns,default,substituteParams,displayAs,incomingTraceProvider`

    removes the specified type override for a field.

  - `--[no]confirmMultiValued`

    specifies whether to confirm the setting of the `--[no]multiValued` option.

  - `--[no]multiValued`

    specifies whether to allow users to select multiple pick list values at one time or link to multiple issues in a relationship field.

    Note

    Caution: Converting a single-valued pick list to a multi-valued pick list may take considerable time depending on the number of issues and issue history in your database. Upon conversion, the storage format for the field is also permanently changed and cannot be reverted to single-value storage. However, you can still revert the field to a single-value pick list, if needed.

  - `--defaultBrowseQuery=[user:]query`

    specifies the user and name of the default Admin query to use when adding a related item by browsing. This option only applies to relationship fields.

  - `--name=value`

    specifies the name of the field (if modifying the field name). Names can be a maximum of 100 characters and cannot contain square brackets.

  - `--description=value`

    specifies a description of the field.

  - `--position=<number>`

    specifies the position in the list of fields.

  - `--default=value`

    specifies the default value for the field.

    Note

    - To include the time with date fields, specify the `--showDateTime` option and use the *mm/dd/yyyy hh/mm/ss* format, where *hh/mm/ss* is the hour, minutes, and seconds. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format. However, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. For a date field, you can specify the current date and a time of 00:00:00 (midnight) when the issue is submitted by typing *today*. For a date/time field, you can specify the current date and time when the issue is submitted by typing *today*. To specify an empty value for a date field, type *none*.
    - Displayed date fields do not change based on the time zone in which a user is operating. However, displayed date/time fields and time entries vary based on the time zone in which a user is operating.
    - Date/time fields cannot be converted to date fields.
    - Integer fields allow a maximum of nine digits and floating point fields allow a maximum of 15 digits.
    - When editing a floating point field to set the default, minimum, or maximum value, you can enter a negative exponent using the E number notation, for example, -123.1E-3.

  - `--defaultColumns=field1,field2,...,mks:virtualField1,mks:virtualField2...`

    specifies the default ColumnSet for the field. This option is only valid for the following field data types: Relationship and Query Backed Relationship. This option only applies to the field being edited, not the relationship field pair.

  - `--displayAs=[default|checkbox]`

    specifies an alternate display for a logical field. By default, a logical field allows users to select true or false values. However, you can choose to display the field as a checkbox (a checkbox that is checked indicates a true value and a clear checkbox indicates a false value). To display a logical field as a checkbox, specify `--displayAs=checkbox`. To display a logical field with true or false values, specify `--displayAs=default` or do not specify the option. This option is only valid with the `--type=logical` option.

  - `--displayTrueAs=value`

    specifies the custom value that represents how the true value displays in a logical field, for example, `--displayTrueAs=yes`. The maximum number of characters is 100 and this option is only valid with logical fields and the `--displayFalseAs=value` option.

    Note

    Important: Specifying custom values for logical fields impacts existing custom scripts that use the default true and false values. PTC recommends reviewing the scripts and making necessary modifications to reflect new custom values.

- `--displayFalseAs=value`

  specifies the custom value that represents how the false value displays in a logical field, for example, `--displayFalseAs=no`. The maximum number of characters is 100 and this option is only valid with logical fields and the `--displayTrueAs=value` option.
- `--min=value`

  specifies the minimum value for the integer, float, or date type field. See the `--default=value` option for rules.
- `--max=value`

  specifies the maximum value for the integer, float, or date type field. See the `--default=value` option for rules.
- `--backedBy=value`

  specifies the issue backed pick list (IBPL) field attribute, where value uses the format *ibpl-name.field-name*. *ibpl-name* is the IBPL that you want to back the field value attribute (FVA) field. *field-name* is the field in the backing issue type whose value you want to display in the FVA field. This option is used with the `--type=fva` option.

  Note
  - The specified IBPL must be single valued.
  - The field must be visible in the IBPL field's backing issue type.
  - You can set display options such as `--displayPattern=value` and `--[no]displayAsProgress`, if the field data type allows them.

- `--backingStates=value`

  specifies active states to populate the issue backed pick list with. Populating an issue backed pick list with backing issues in specific states essentially allows you to "deactivate" entries in an IBPL. For example, if an Employee issue contains an `Active` and `Inactive` state, and you select `Active` as the active states, only Employee issues in a state of `Active` display pick text entries in the IBPL. This option is used with the `--type=ibpl` option.
- `--backingTextField=field`

  specifies the non-computed short text field in the backing issue type that you want to use as pick text. For example, if you select the `Manager` field and `James Riley`, `Sherry Robertson`, and `Dan Evans` are field values in some of the backing issues, those names display as pick text in the issue backed pick list field. This option is used with the `--type=ibpl` option.
- `--backingIBPLTextFormat=value`

  specifies multiple fields in the backing issue type that you want to link together to use as pick text. The format is similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). For example:

  `--backingIBPLTextFormat="{ID}:{Summary}"`

  Note

  This option overrides any value specified by the `--backingTextField` option.

- `--backingType=type`

  specifies the issue type containing the short text field that you want to reference. This option is used with the `--type=ibpl` option.
- `--[no]sortIBPLDescending`

  specifies the sort order of the field on the backing item type in the IBPL. `--[no]sortIBPLDescending` sorts the field in ascending order. `--sortIBPLDescending` sorts the field in descending order. This option is used with the `--type=ibpl` and `--sortIBPLField=field` options.

- `--sortIBPLField=field`

  specifies a visible field on the backing item type to sort by in the IBPL. For example, if an IBPL field uses a `Build` item as the backing item type, you could display the most recent build from the `Build Number` field at the top of the IBPL field. If this option is not specified, field values in the IBPL are sorted alphanumerically. This option is used with the `--type=ibpl` and `--[no]sortIBPLDescending` options.

  Note
  - The following field types on the backing item type cannot be sorted: type, attachment, IBPL, FVA, QBR, range, relationship, source link, and SI project fields.
  - If a field is currently specified as the sorting field and you choose to configure that field as invisible in the item type, a warning message notifies you that this will affect the sort order, allowing you to save or cancel the change. Saving your changes causes field values in the IBPL to be sorted alphanumerically.

- `--backingFilter=<querydefinition>`

  specifies a string to define the picklist value constraints. This option is used with the `--type=ibpl` option. The *<querydefinition>* must be of the same format as the query definition for a query. For details of the query definition format, see im createquery.

  Note

  The constraints defined for this filter are applied in addition to the filtering specified by the `--backingType` and the `--backingStates` options.

- `--computation=value`

  specifies a computed expression for the computed field. For more information on computed fields, see the im createfield command.
- `--[no]allowComputationUpdatesOnVersion`

  specifies to record the computation value at the time of versioning and prevent further updates. By default, the computation value in versioned items continues to update based on the computed field definition. If you specify `--[no]allowComputationUpdatesOnVersion` on a non-computed field, an error message displays. If you specify `--allowComputationUpdatesOnVersion` on a non-computed field, the option is silently ignored.
- `--[no]staticComputation`

  specifies the computation type. `--staticComputation` performs the computation and stores it in the issue's history based on the value specified in `--storeToHistoryFrequency=value`. PTC recommends a static computation if your expression involves expensive external functions, such as query or aggregate functions. `--[no]staticComputation` performs the computation every time field values used in the expression change. `--[no]staticComputation` is a `dynamic` computation and is selected by default.

  Note

  Because dynamically computed fields are not stored in the database, dynamically computed short text fields cannot be located with an all text field search in the Integrity Client. To search for dynamically computed short text fields, create a query that includes a specific "field contains" comparison.

- `--storeToHistoryFrequency=[never|daily|weekly|monthly|delta]`

  indicates how often the computed field should be calculated and stored in the issue's history. Acceptable values are *daily*, *weekly*, *monthly*, *delta* or *never*. *delta* specifies to store the computed field's value to the item history only when a delta is detected. To specify a custom frequency, specify *never* and create an event trigger that specifies the desired frequency. Selecting a frequency is useful for historical charting. *never* is specified by default.

  Note

  Using the `im analytics --recomputeHistory` command, you can calculate a computed field within a specific time frame, storing the value in the issue history. This command is useful for historical charting and reporting, allowing you to calculate, then compare the stored historical values of the computed field between current and past projects.

- `--addEntry=value`

  adds a phase or range field value to a new or existing field.

  For phases, *value* is specified as *PhaseName=PhaseName:state,state,...:image PhaseFieldName*, where *state* specifies an included state, and *image* specifies the file path and the name of a custom image, or *none*.

  For ranges, *value* is specified as *RangeNane=RangeName:lowerValueupperValue:image*

  Note the following about adding phases:
  - Only a single phase entry can be edited per command execution.
  - Separate multiple states using commas.
  - States not grouped into a phase are referred to as out of phase states. When an issue is in an out of phase state, the phase field displays `Out of Phase`. You cannot edit the `Out of Phase` phase. This phase identifies the states that are not included in any user defined phase.
  - Images for phases must be GIF or JPEG format, and be no larger than 16 by 24 pixels.
  - No two phases can use the same state.

  Note the following about adding range limits:
  - You cannot specify different range category names and icons for types. However, you can specify different range limits for types.

- Range field values are automatically determined based on an associated numeric field. Range fields cannot be edited in an Issue Details view.
- Range value limits can be overridden on a per type basis. However, range category names and icons cannot be overridden.
- If *lowerValue* is not set, `-Infinity` is automatically entered. If *upperValue* is not set, `+Infinity` is automatically entered. An `Infinity` value indicates that all numbers are valid.
- A numeric value must be contained in one defined range; range intersections are invalid. For example, the following ranges are invalid: 0;5 and 4;8, or 0;5 and 5;10. For an integer field, an acceptable range would be 0;5 and 6;10. For a floating point field, an acceptable range would be 0;5 and 5.01;10.
- If a value is entered in the associated numeric field that is beyond the set range values, the range field displays `Out of Range` in the issue. If no value is entered in the associated numeric field, the range field is empty.
- For ranges, *value* is specified as *text:lowerValue;upperValue:image,...*, where *label* specifies the range category name, *lowerValue* specifies the lower range, *upperValue* specifies the upper range. For example, `--addEntry=Golden:-Infinity;0,Acceptable:1;6,Watch:7;19,Trouble:20;+Infinity`. Range category names can be 100 characters long.
- At this time, you can only create ranges using the <= operator.

- `--editEntry=`*value*

edits an existing phase or range field value on an existing field.

For phases, *value* is specified as *--editEntry=existingPhaseName=newPhaseName:state,state,...:image*

For ranges, *value* is specified as *--editEntry=existingRangeName,...* to define one or more values.

- `--deleteEntry=`*value*

deletes an existing phase or range field value from an existing field.

For phases, *value* is specified as *--deleteEntry=existingPhaseName, ...* to define one or more values.

For ranges, *value* is specified as *--deleteEntry=existingRangeName, ...* to define one or more values.

- `--associatedField=`*fieldName*

specifies the numeric field to associate with the range field.

Note

Range fields cannot contain an associated field that includes a computed expression with an external information function.

- `--[no]displayAsProgress`

displays the integer value as progress bar in the GUI.

- `--[no]displayAsLink`

displays the value selected in the item backed pick list as a hyperlink to the backing item. The hyperlink displays in the GUI and the Web UI.

- `--[no]trace`

sets the field as a trace relationship. Trace relationships are defined via field pairs and are presented to the user in domain-specific language, for example, Test and Requirements. To more about trace relationships, see your *PTC Integrity User Guide*.

Note

When used with `im editfield`, this option cannot be used to enable or disable tracing on a sourcelink field. For more information, see the reference page for `im createfield`.

- `--picks=`*text:value:image,...*

specifies the list of valid active values for a pick list field, where image can be `none`, or the file path and the name of a custom image. Unspecified existing pick list values are considered inactive. Pick text must be 100 characters or less in length and empty values are not supported.

Note

- This option requires the complete list of active pick list values to be specified. Unspecified pick list values are inactive. Do not specify duplicate pick field values.
- Inactive pick list values continue to display in fields, history, query filters, relevance and editability rules, constraint filters, charts, and reports.
- Inactive pick list values can be specified in query filters, relevance and editability rules, constraints, charts, and reports.
- Inactive pick list values cannot be specified in pick list fields. However, pick list fields retain inactive pick list values. If a user edits a multi-valued pick list field, inactive pick list values can no longer be specified, even if only one of the values was previously inactive.
- If one or more active pick list values are referenced in a trigger field assignment and you attempt to make one of those values inactive, an error message displays the pick list values and the trigger(s) where the assignment occurs. The references in the event trigger(s) must be removed before making a pick list value inactive.

- `--maxLength=`*value*

specifies the maximum character length value for a short text, long text, or rich content field. For long text fields, the maximum number of characters is 4,000.

- `--suggestions=`*text1,text2,text3,...*

specifies a list of suggested values for a short text field.

Note

This setting requires the complete list of suggested values to be specified.

- `--displayName=`*value*

specifies the name assigned as the display name of the field.

- `--displayPattern=`*value*

assigns a format to floating point or integer field values, known as a display pattern. Display patterns allow you to quantify numeric field values, for example, as currency or percentages. You can define a display pattern by combining a currency symbol, text that represents a measurement, and/or one or more of the following characters:

- `0` - Displays as a zero in the output. For example, a display pattern of `000.00` displays an input value of `12.14` as `012.14` in the numeric field.
- `#` - Displays as a digit in the output. If the digit is a zero and it is leading or trailing the input value, it is left out of the value displayed in the numeric field. For example, a display pattern of `#0.00` displays an input value of `0.126` as `0.13` in the numeric field.
- `.` - Locale specific decimal separator.
- `-` - Minus sign.
- `,` - Locale specific grouping separator.
- `E` - Scientific notation, in the format `aEb`, where `a` is any real number, and `b` is the exponent..
- `;` - Separates positive and negative patterns.
- `%` - Multiplies by 100 and displays as a percentage.
- `'` - Escapes special characters. Use `''` to create a single quote.

For example, if you specified a display pattern of `$#,###` and a user types `12345.123` in the associated numeric field, the numeric field displays `$12,345`. Similarly, if you specified a display pattern of `# minutes` and a user types `123` in the associated numeric field, the numeric field displays `123 minutes`

Note

- If the display pattern is invalid or the field type is not an integer or floating point, an error message displays. If no display pattern is specified, the field displays the value in a localized form.
- Display patterns appear only when viewing one or more issues. Integrity stores the field value as an unformatted numeric value in the database.
- By default, a floating-point field value displays the same number of decimal places as when the value was entered. Previously, floating point values rounded to three decimal places.
- Display patterns are applied to numeric values only when shown in the context of an issue. This means that query filters, rules, and trigger assignments display the unformatted, localized version of the numeric field value.

- `--relevanceRule=`*rule*

specifies the rules that determine when users see the field. For the rule syntax, see Specifying Rules on the options reference page.

Relevance rules are evaluated on the Integrity Client's time zone.

- `--relevanceRuleFile=`*filename*

specifies a file that contains the field relevance rules. See `--relevanceRule` for notes on the option and obtaining rule syntax for the file format.

- `--editabilityRule=`*rule*

specifies the rules for when users are permitted to edit the field. For the rule format, see Specifying Rules on the options reference page.

To prevent the field from being edited, specify `--editabilityRule="(false)"`. This option is useful for fields that are updated by event triggers and are not meant to be edited by

users. For example, you could create a date field where the date is automatically specified when the issue enters a certain state.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date or date/time field, type *today*. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can choose yourself by specifying "me". "me" is a symbolic user which refers to the currently logged in user. For example, you could create an editability rule that specifies the Requirements field can be edited if the currently logged in user is one of the users defined in the multi-valued Stakeholders field.
  - Editability rules are evaluated on the Integrity Client's time zone.
  - Configuration management project and attachment fields cannot be specified.
  - By default, --editabilityRule="(false)" is specified for read-only custom fields (i.e. phase, range, computed) and cannot be unspecified.
  - Editability rules cannot include a computed expression that requires an external information function.

- --editabilityRuleFile=*filename*

  specifies a file that contains the field editability rules. For the file format, see Specifying Rules on the options reference page.
- --copyRelevanceRule=*field*

  copies the relevance rules of an existing field to the one being edited.

Note

A false relevance rule (--relevanceRule="(false)") can be copied in the CLI. However, you cannot do this in the GUI.

- --copyEditabilityRule=*field*

  copies the editability rules of an existing field to the one being edited.

Note

A false editability rule (--editabilityRule="(false)") can be copied in the CLI. However, you cannot do this in the GUI.

- --allowedTypes=*type:type,type,...[;...]*

  specifies the types of issues that can be linked using the relationship field. Specify the issue type that will use the relationship field, then list the issue types that can be linked to using the reverse relationship field. For example, for a one-way relationship showing the relationship between documentation issues and bugs, specify *Docs:Bugs* for the forward field, and *Bugs:Docs* for the reverse field.

  For a two-way relationship, you need to specify allowed types for both sides of the relationship. For example, to allow the field to be used to create relationships between documentation issues and bugs, specify *Docs:Bugs;Bugs:Docs*.
- --addLinkFlags=*name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];..*

  defines the relationship flags that can be added to relationships in the relationship field.
- --editLinkFlags=*existingName=value|name=value,displayChar=char,onImage=path,enabled=[true|false],suspect=[true|false];...*

  edits the values of existing relationship flags for the relationship field.

Note

Important: The |symbol in [true|false] means you can choose any value between them. The | in existingName=value|name=value is a literal, so you have to specify both existingName and name and use the |to connect them to make the command work with this stipulated value.

For example:

--editlinkflags=existingName='example'|name='abc',displayChar=?,onImage=c:/temp/small.jpg,enabled=true,suspect=true

---

- --disableLinkFlags=*value,...*

  a comma separated values list of the relationship flag names to disable for the relationship field.
- --enableLinkFlags=*value,...*

  a comma separated values list of the relationship flag names to enable for the relationship field.
- --[no]cycleDetection

  specifies whether or not the system will prevent relationship loops from occurring in the relationship field. For more information on relationship loops, see the *PTC Integrity User Guide*.
- --query=*[user:]query*

  specifies an administrator query to use as the backing query for a query backed relationship field.
- --correlation=*src-field:dest-field,...*

  specifies a pair of fields to correlate between the type containing the query backed relationship field and the issues returned by the query. For example, if you create a query backed relationship field called Defects for the Feature type and specify Project as the source field and Project as the target field, the Defects field displays all Defect issues that have the same project as the one specified in the Feature type's Project field. This option is not mandatory. However, if it is not specified, the list of relationships returned does not change with different issues.
- --displayRows=*value*

  specifies the number of rows to display for a long text, rich content, sourcelink, or relationship field. There is a maximum of 80 rows permitted for a long text, sourcelink, or relationship field, and 15 for a rich content field.
- --displayStyle=*value*

  specifies whether the attachment, relationship, sourcelink, or query backed relationship field displays in table format or in a comma separated values (CSV) format. Acceptable values are *csv* or *table*.

  For attachment, relationship and query backed relationship fields, if you specify -g or --gui, the table format allows you to sort the issues and manipulate the columns that display in the table. The CSV format only displays the IDs of the issues. For sourcelink fields, if you specify -g or --gui, the table format allows you to manipulate the columns that display in the table. The CSV format only displays the source file name, revision number, server and project.
- --loggingText=[*none|mostRecentFirst|mostRecentLast*]

  specifies logging text field mode, including the order the entries are displayed. Logging text field mode is only a valid option for longtext type fields.
- --showDateTime

  specifies to include the time with the date in a date field.

Note

Important: Once you include the time and save the date field, you cannot change the date field to display the date only.

- --relationshipBrowseStyle=*browseStyle=[query|finder],queryOption=[allQueries|specificQueries],specificQueriesList=query::query::...,finderQuery=[user:]query, outlineFormat=value,relationshipstoFollow=relationship::relationship::...*

  specifies the relationship browse style that is used for editing a related item. browseStyle can be either set to query or finder.

  After you set browseStyle to query or finder, the following options can be edited:

  - queryOption is applicable for browseStyle=query and can be specified as either "allQueries" or "specificQueries". When browseStyle=query is specified, the queryOption=[allQueries|specificQueries] is mandatory.
  - specificQueriesList is applicable for queryOption=specificQueries. This option specifies the list of the names of Admin queries, separated by double colons. For example: specificQueriesList="query1::query2::query3
  - finderQuery is applicable for browseStyle=finder. This option specifies the name of the Admin query to be used for the content of the Find Items to relate window left pane.
  - outlineFormat is applicable for browseStyle=finder. This option specifies the outline in the right pane of the Find Items to relate window.
  - relationshipsToFollow is applicable for browseStyle=finder. This option specifies the list of relationships, separated by double colons (for example, relationshipsToFollow="relationship::relationship2::...). In the Find Items to relate window, the right pane is populated with the list of related items derived from the selected items in the left pane. The relationshipsToFollow defines the related items with which to populate the right pane.

  Note

  - If you do not provide a valid sub-option and its value for any browseStyle, then its value remains the same as specified in the respective relationship field.
  - If browseStyle is changed from query to finder or vice-versa, the values of the invalid sub-options is set to none. For example, when you change the browseStyle from query to finder, the values for finderQuery, reationshipToFollow, and outlineFormat is set to none.
  - If you want to set the browseStyle as the default style, then specify the relationshipBrowseStyle as --relationshipBrowseStyle="".

For example:

- When the `browseStyle` is set to `query`:

  im editfield --defaultBrowseQuery=Query1_default --relationshipBrowseStyle=browseStyle=query,queryOptions=allQueries Relationship_Field_ABCD

  im editfield --defaultBrowseQuery=Query1_default --relationshipBrowseStyle=browseStyle=query,queryoption=specificQueries,specificQueriesList=Q2::Q3::Q4::Q5 Relationship_Field_ABCD

- When `browseStyle` is set to `finder`:

  im editfield --defaultBrowseQuery=Query1_default --relationshipBrowseStyle=browseStyle=finder, finderQuery=Q1,outlineformat="{Section}-{Category} {SharedCategory}",relationshipstofollow=R1::R2 Relationship_Field_ABCD

- `--[no]richContent`

  specifies whether to configure the long text field as a rich content field. `Rich content` enhances the display of text in long text fields by adding formatted text, tables, background colors, images, and hyperlinks. This option is enabled by default and does not support logging text fields.

Note

Caution: You can convert rich content fields back to long text fields. However, any existing rich content is displayed as HTML tags and attributes.

Because rich content is expressed using a limited set of HTML elements and attributes, you can define screen and printer Cascading Style Sheets (CSS) that ensure a consistent look when viewing and printing rich content field data in different Web browsers. For more information, see the *PTC Integrity Server Administration Guide*.

- `--defaultAttachmentField=`*field*

  specifies the default attachment field that images are retrieved from when inserting images into a rich content field via attachment field. The default is the default `Attachment` field

- `--[no]computeNow`

  calculates the field in issues where the values of the underlying fields used in the computed expression have changed since the last computation.

- `--[no]showTallRows`

  specifies whether the relationship field should display variable height rows.

- `--[no]forceRecompute`

  if you changed the computed field's underlying computed expression and/or you want to calculate the field in all issues containing the field, then this option resets the last computation time associated with the computed field and recalculates the computed field in all issues containing the computed field.

- `--[no]textIndex`

  specifies whether queries against the field will be treated as word searches.

- `--[no]substituteParams`

  specifies whether parameter references in this text field are replaced with parameter values when you view the item through a view or report that supports parameter substitution. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.

- `--ierDefaultColumns=[`*server*`,`*title*`,`*name*`...`

  specifies the columns that appear for trace relationships. This option is only valid for a `type` of `ier`.

  Available values are:

  - `server`—Server where the object resides
  - `title`—Summary of the object
  - `name`—Name of the trace
  - `reversename`—Trace type, such as "Satisfied By"
  - `suspect`—Indicates whether the trace is suspect
  - `icon`—Icon for this object type
  - `largePreview`—Large preview of the object
  - `smallPreview`—Small preview of the object
  - `URI`—Uniform resource identifier of the object
  - `localURI`—Uniform resource identifier of the requirement

- `--incomingTraceProvider=`*value*

  The incoming trace provider that is used to retrieve external references. If you are using ThingWorx, this is the name of the ThingWorx thing that returns IER values. This option is only valid for a `type` of `ier`.

- *field*

  specifies the name of the field you want to edit.

See Also

- Commands: im createfield, im viewfield, im fields, im analytics
- Miscellaneous: options

im editgroup

edits a group

Synopsis

```
im editgroup [--name=value] [--description=value] [--queryTimeout=value] [--sessionLimit=value] [--image=[none|default|<path>] [--email=value] [--notificationRule=rule] [--notificationRuleFile=value] [--copyNotification=[user|group]:<name>] [--[no]active] [--name=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] group
```

Description

`im editgroup` edits a group for workflows and documents. You can edit the following group details: group name, e-mail address, image, description, and notification settings. For example:

```
im editgroup --image=/admin/creategroup/services2.jpg "Services"
```

changes the custom image in the *Services* group.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the new name of the group. Names can be a maximum of 100 characters and cannot contain square brackets.

  Note

  PTC recommends you do not use commas, colons, or special characters in group names. If you use commas in the group name, you cannot select the group in a multi-valued user field. If you use colons in the group name, the group cannot edit shared system provided objects (charts, queries, reports, and dashboards).

- `--description=value`

  specifies a description of the group.

  `--image=[none|default|<path>]`

  specifies whether an image appears for the group.

  - `--image=none` does not specify an image for the group.
  - `--image=default` specifies the default "group" image for the group.
  - `--image=[<path>]` specifies the path and name of a custom image for the group, for example, `c:\images\group_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--email=value`

  specifies the email address of the group.

- `--notificationRule=rule`

  specifies the notification rule for this principal. For the rule syntax, see Specifying Rules on the options reference page.

  Note

  - To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date field, type *today*. This option can be specified for a date field or a date field configured to display the date and time. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - E-mail notification is subject to project, type, and field visibility rules. Only users that have visibility for a given project and type receive e-mail notification for issues related to that project and type. In addition, e-mail notifications include only the fields they have permission to view.

- `--notificationRuleFile=value`

  specifies the file that contains the rules to read. For the file content format, see Specifying Rules on the options reference page.

- `--copyNotification=[user|group]:<name>`

  copies the given user/group's notification rules. This command copies the given user/group's notification rule into this group's rule at the time that the command is run. The command does not create a pointer to another notification rule.

- `--[no]active`

  Specifies if the group is actively used in Integrity. This option is useful for removing groups that are no longer needed. Specifying `--noactive` prevents the group from being displayed as an available default group field value; however, the group still appears in existing issue data.

  Note

  If the group is referenced in a trigger assignment or as the default value in a group field, moving the group from active to inactive displays an error message. The reference must be removed before making the group inactive.

- `--queryTimeout=value`

  specifies the total time in seconds allowed for queries performed by members of the specified group.

- `--sessionLimit=value`

  specifies the number of sessions each user in the group can open at one time. This option assists in managing resource allocation for the Integrity Server. For example, if you specify 5, a user can begin a session in the Web interface, and open 5 tabs in a browser to run a chart in each tab. You can specify a maximum of 100 connections. Default is 5; however, a value of 0 specifies the default value.

  Note

  If there are Web interface users with heartbeat and idle timeout, then an idle-timed-out user still continues to exist, and is not subject to this limit.

- *group*

  specifies the name of group you want to edit.

See Also

- Commands: im creategroup, im viewgroup, im deletegroup, im groups
- Miscellaneous: options

im editissue

edits an existing Integrity issue

Synopsis

```
im editissue [--[no]showWorkflow] [--[no]batchEdit] [--query=[user:]query] [--removeAttachment=value] [--removeFieldValues=value] [--removeRelationships=value] [--addAttachment=value] [--addFieldValues=value] [--updateAttachment=value] [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--removeSourceTrace=value] [--removeSourceLink=value] [--customFieldDefinition=value] [--customFieldValue=value] [--removeCustomFieldDefinition=name] [--field=value] [--richContentField=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] issue id...
```

Description

You can edit and update issues, depending upon the permissions assigned to you by your administrator.

You would need to edit an issue if, for example, someone on your project team assigned an issue to you, making you responsible for performing some action; or if you were responsible for reviewing all new submissions and assigning them to other team members.

When you use the `-g` or `--gui` option, Integrity displays an issue selection dialog box. If you browse for issues from the issue selection dialog box, any changes you make to the columns in the Issues View will not be applied to that view when it is accessed through the Integrity Client GUI.

Note the following:

- Your administrator can configure long text fields to support rich content. *Rich content* enhances the display of text in long text fields by adding formatted text, tables, background colors, images, and hyperlinks. From the CLI, rich content is applied using a limited set of HTML elements and attributes. For a complete list of supported elements and attributes, see the *PTC Integrity User Guide*.
- Your administrator defines which issue types and custom fields you are allowed to edit. If your administrator defines a field as a logging text field, you may only enter new text and not edit existing text.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields vary based on the time zone that a user is operating in.
- You cannot edit historical versions of issues.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- The list of users in the Assigned User field is limited to those with permissions to the issue's project. The same applies to the Assigned Group field.
- Depending on your workflow, you may not be able to edit an issue that is in an end state.
- Your administrator may include the time in date fields. You can specify the time when you select a date from the calendar. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. If you do not specify a time, the current time displays in the date field.
- To retrieve metrics from a configuration management project related to the issue you are editing, your administrator may define a field that accepts a configuration management project as a value. Optionally, you can specify a checkpoint revision or development path. If you specify a configuration management project and a checkpoint, then save the issue, one or both of the following may occur when you view the issue in the GUI or Web interface:
- One or more computed expressions in the issue calculate specific metrics about the project, displaying the results as a read-only value in a computed field (the visibility of the computed field depends on the field's relevance rules). For example, once you specify a project for the `Source Code` field, a `Lines of Code` field could calculate and display the number of lines of code in that project. As lines of code are added or removed from the project, the `Lines of Code` field updates to display the new value.
- A `metrics` hyperlink displays in the configuration management project field. Clicking the hyperlink displays various configuration management metrics about the project.

In addition, the server and project information display in the configuration management project field as a hyperlink. Clicking on the hyperlink displays the project in a Project view.

To select a configuration management project, you require the `OpenProject` permission for the specified project. Once a configuration management project has been specified, metrics can be obtained by any user with permissions to view the configuration management project field. For more information on selecting configuration management projects and viewing configuration management metrics, refer to the *PTC Integrity User Guide*. For more information on creating configuration management metrics, refer to the *PTC Integrity Server Administration Guide*.

Important:

Metrics are only maintained against project checkpoints; therefore, to generate metrics, you must specify a checkpoint when you specify the configuration management project.

- You cannot set a date field to null if the date has been previously set.
- Your administrator may include the time in date fields. You can specify the time when you select a date from the calendar. Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying `13:56:45` displays the time as `1:56:45 PM`. If you do not specify a time, the current time displays in the date field.
- You can modify a value from a mandatory field and save it.
- If your administrator has set up electronic signatures, you may need to provide your user name and password when making specific edits to an issue. For example, you could be required to provide an electronic signature when you change an issue's state to `Completed`.
- Integer fields allow a maximum of nine digits and floating point fields allow a maximum of 15 digits. Your administrator can define default, minimum, and maximum values.
- By default, Integrity allows file attachments to a maximum size of 4 MB and a maximum of 255 characters for file names. Your administrator may define a higher or lower limit depending on the requirements of your system. You can also attach more than one file to a single issue.
- If you attempt to save changes to an issue after another user saves changes to the same issue, the following error message may appear: Could not save modified issue: The issue was changed by another user after you began your edit. Typing `Cancel` discards your changes. Typing `OK` displays your unsaved changes to the issue. PTC recommends copying your changes, canceling the issue, then re-editing the issue and adding your changes.
- Setting a user, group or project field to an inactive value results in an error message.
- Inactive pick list values cannot be specified for pick list fields; however, pick list fields retain inactive pick list values. If you edit a multi-valued pick list field, inactive pick list values can no longer be specified, even if only one of the values was previously inactive.
- If an issue contains multi-valued user or group fields with inactive values, editing the inactive values and saving the issue prompts you to leave the fields unchanged or clear the inactive values.

For example,

```
im editissue --query="Deferred Release 2.0 Features" --field=Project:/"Release 3.0"
```

edits all issues returned by the `Deferred Release 2.0 Features` query to change the `Project` field to `Release 3.0`.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--addAttachment=value`

  adds attachments, where *value* is of the form "*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]* " .

Note
The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

For example,

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
```

adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. An issue cannot have more than one attachment with the same name.

Note
Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

- `--addFieldValues=`*value*

  specifies new field values to add, where *value* is of the form "*fieldName=fieldValue*."

Note
This option is only valid for the following multi-valued field types: pick, user, group, item backed picklist (IBPL), and relationship.

- `--updateAttachment=`*value*

  updates information for an existing attachment, where *value* is of the form "[*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]*]".

  Note the following:

- If there are no differences in the updated attachment, the existing attachment is not updated.
- If the specified attachment does not exist, it is added to the item.
- If the updated attachment exists, but there are differences, the existing attachment is removed and the new attachment is added.
- The comparison of attachments is based on a message dialog algorithm (MD5) which relies on a checksum to calculate and compare the size of the attachment files. This comparison method is not guaranteed to work in all cases and it is possible to have collisions between two different attachments that are identical in size. While such collisions are rare, to avoid any possibility of collisions when using the `im editissue`\u00a0command, you can manually update the attachments using the `--addAttachment`\u00a0and `--removeAttachment`\u00a0command options.
- `--addRelationships=`*value*

  adds related issues, where *value* is of the form *[FieldName]*:*IssueID*[*relationshipFlags*][,...].If no field name is specified, the `Forward Relationships` field is used. Adding a related issue is only permitted if your administrator has allowed relationships for the issue type.

Note
This option is deprecated. Instead, use the `--addFieldValues`=value option.

- `--addSourceTrace=`*value*

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

- If you are specifying a variant or build project for *project=scmProjectName* use a keyword string. For build projects, make sure you specify the build keyword (`#b=`) immediately after the top level project keyword (#). For more information on keyword strings, see the description for `--project=`*project* under general options on the options reference page.
- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
- `--addSourceLink=`*value*

  adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
- `--[no]showWorkflow`

  specifies whether to display the workflow for the issue type, if your administrator has enabled it. This option can only be specified with `-g` or `--gui`. Viewing the `Workflow` panel is useful for determining where you can progress in the workflow. The `Workflow` panel displays the complete workflow for the issue type, unvisited states, visited states, the current state, and other state transitions, as indicated by the Legend.
- `--[no]batchEdit`

  specifies whether to edit multiple issues as one operation, or to edit each issue individually.

  `--batchEdit` edits multiple issues as one operation and saves the changes only after all the issues are edited.

  `--nobatchEdit` edits and saves the changes to each issue individually.

Note
You cannot perform the following edits in a batch edit:

- remove attachments
- add or remove source links (or source traces from a source link field with trace enabled)
- `--field=`*value*

  specifies a field and its value for the issue, where *value* is of the form "*fieldName=fieldValue*", for example, `--field=`"*Severity=Critical*". If the field is multi-valued, *value* is of the form "*fieldName=fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add to the issue.

  To specify an Integrity project, project names must be preceded by a (/), for example, `--field=Project=/testProject`.

  To specify a configuration management project, use the following syntax: `--field=`*field-name*`=server=`*server:port*`: :project=`*projectname/project.pj*`::(devpath=`*devpath*`|revision=`*checkpoint-revision*).
- `--richContentField=`*value*

  specifies a rich content field and its value for the new issue, where `value` is of the form "`richcontentfieldname=fieldValue`".

  To specify a link to an item by ID: `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547">Part Requirement</a>`

  To specify a link to an item by ID and label: `<a href="mks:///item?itemid=`*ID*`&label=LABEL">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>`

To specify a link to an item by ID and revision: `<a href="mks:///item?itemid=ID&revision=REVISION">DisplayText</a>`

For example: `<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>`

To specify a link to an item by ID and date: `<a href="mks:///item?itemid=ID&asof=DATETIME">DisplayText</a>` where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

For example: `<a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>`

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "`b>Feature Overview/b>`". In the Windows command line interface (cmd.exe), the ^ escape character must precede the and > characters in HTML tags, for example, `^b^>Feature Overview^/b^>` . You may need to escape nested " characters, for example, `--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"`.

- `--query=`*`[user:]query`*

  specifies the name of a query to populate the issue selection, for example, "`Cosmos Critical Defects`". All issues returned in the query are selected for editing. If any issues are specified, the `--query` option is ignored).

Note

Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the query named `CosmosDefects` created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the query named `jhoyt:CosmosDefects` created by any user.

- `--removeAttachment=`*`value`*

  removes attachments, where *value* is of the form "[*field=fieldName,name=nameOfAttachment*]".If no attachment field is specified the default `Attachments` field is used.
- `--removeFieldValues=`*`value`*

  specifies existing field values to remove, where *value* is of the form "*fieldName*=`fieldValue`."

Note

This option is only valid for the following multi-valued field types: pick, user, group, item backed picklist (IBPL), and relationship.

- `--removeRelationships=`*`value`*

  removes related issues, where `value` is of the form [`fieldName:`]`id`[,...]. If no field name is specified, the `Forward Relationships` field is used.

Note

This option is deprecated. Instead, use the `--removeFieldValues`=value option.

- `--removeSourceTrace=`*`value`*

  removes a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:
- You can specify the configuration path for *project=scmProjectName* using a keyword string. For more information on specifying project paths using a keyword string, see the description for `--project=`*project* under general options on the options reference page.
- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- `--removeSourceLink=`*`value`*
- removes a link to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,revision=memberRevisionID, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:
- You can specify the configuration path for *project=scmProjectName* using a keyword string. For more information on specifying project paths using a keyword string, see the description for `--project=`*project* under general options on the options reference page.
- If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
- *issue id...*specifies the ID of the issue you want to edit. Use spaces to specify more than one issue, for example *34 23*. This option must be used if a query is not used to select issues for editing. This selection overrides the `--query` option.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

See Also

- Commands: im copyissue, im createissue, im extractattachments, im viewissue
- Miscellaneous: options

im editproject

edits the properties of a project

Synopsis

`im editproject [--name=value] [--parent=project] [--description=value] [--permittedAdministrators=u=user1,user2,...;g=group1,group2] [--permittedGroups=group,group,...] [--[no]inheritPermittedGroups] [--openImage=[none|default|<path>] [--closedImage=[none|default|<path>] [--[no]active] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]project`

Description

`im editproject` edits the properties of a project for workflows and documents. For example:

```
im editproject --inheritPermittedGroups --parent=/FinancialToolkit "SavingsManager"
```

changes the *SavingsManager* group to inherit the permitted groups from the parent project *FinancialToolkit*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  Specifies the new name of the project. Names can be a maximum of 100 characters, cannot contain square brackets or a forward slash (/).

- `--parent=project`

  Specifies the new name of the parent project. If specified, then the project becomes subproject of the new parent project. The specified parent project must exist before creating a subproject. When specifying a parent project, you must include a forward slash and the full project name, for example, `/AuroraProject`.

- `--description=value`

  Specifies a description of the project.

- `--permittedAdministrators=u=user1,user2,...;g=group1,group2,...`

  Specifies a comma delimited list of users and/or groups that can administer this project. Project Administrators are allowed to edit and view projects. Project Administrators can also manage all child projects of the project they are approved for, but they cannot edit projects assigned to another Project Administrator. Project Administrators can only create and delete subprojects, they cannot create top level projects unless they have been granted the CreateProject ACL permission. Project Administrators can also view all users, groups, and dynamic groups, but they can only edit dynamic groups membership for the projects they are assigned to. For more information, see the *PTC Integrity Server Administration Guide*.

- `--permittedGroups=group,group,...`

  Specifies a comma delimited list of groups that have visibility for this project.

- `--[no]inheritPermittedGroups`

  Controls whether or not permissions are inherited from the parent project. Project settings cannot be inherited unless there is a parent project. Inherited permissions are mutually exclusive with `--permittedGroups`.

- `--openImage=[none|default|<path>]`

  Specifies whether an image appears for an open project. The default image is an open folder.

  - `--image=none` does not specify an image for the project.
  - `--image=default` specifies the default "open" image for the project.
  - `--image=<path>` specifies the path and name of a custom image for the project, for example, `c:\images\project_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--closedImage=[none|default|<path>]`

  Specifies file path for an image icon file used for a closed project when displayed in the graphical user interface. See `--openImage` for details. The default image is a closed folder.

- `--[no]active`

  Specifies if the project is actively used in Integrity. This option is useful for removing inactive projects in your organization. Specifying `--noactive` prevents the project from being displayed as an available default project field value; however, the project still appears in existing issue data.

  Note

  If the project is referenced in a trigger assignment, moving the project from active to inactive displays an error message. The reference must be removed before making the project inactive.

- *project*

  specifies the project you want to edit. Include a forward slash (/) when specifying a high level project. For subprojects, include the full path to the subprojects, for example, `/AuroraProject/BorealisSubProject`.

See Also

- Commands: im createproject, im viewproject, im deleteproject, im projects
- Miscellaneous: options

im editquery

edits the properties of an existing Integrity query

Synopsis

```
im editquery[--copyColumnsFromQuery[=[user:]query] [--fields=field,field,...] [--[no]sortAscending] [--sortField=field] [--image=[none|default|<path>]][--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--[no]confirmSharedAdmin] [--description=value] [--name=value] [--sharedAdmin] [--hostname=value] [--queryDefinition=query] [--queryDefinitionFile=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]][user:]query
```

Description

You use `im editquery` to edit the properties of an existing Integrity query.

For example, the following edits the existing `Release 2.0 Docs Issues` query to show the estimated and actual hours for the issues:

```
im editquery --fields="State","Estimated Hours","Actual Hours","Assigned User" "Release 2.0 Docs Issues"
```

Note the following:

- You may edit only one query at a time and only queries that you have created.
- The quick query cannot be hidden, deleted, shared with other users and groups, renamed, or given a description.
- Queries with several filters may take longer to run than queries with only one or two filters.
- You cannot edit a query's associated column set if it is currently being used in an Issues view; however, you can edit the default column set if it is currently being used in an Issues view.
- You cannot query on configuration management project fields.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.
- Symbolic dates are evaluated on the Integrity Client's time zone.
- When specifying a date range in a query, all dates are treated as timestamps, converted to the time zone on the Integrity Server, and then truncated to a date-only format. The resulting date-only format is then converted to a Structured Query Language (SQL) statement format on the Integrity Server, and the query is run based on the time zone of the server. If the user defining the date range is not in the same time zone as the Integrity Server, a day can be lost or gained at either end of the defined range. The rollback of dates can cause the query results to vary.

Note

Important: The date range conversion does not cause any problems when the user is in the same time zone as the Integrity Server. For example, if the Integrity Server is in the America/New_York time zone and the following query date range is defined by a user in Germany: Jan 1, 2006 to Jan 31, 2006. The final conversion of the date range is: Dec 31, 2005 06:00:00 AM GMT+01:00 to Jan 31, 2006 06:00:00 AM GMT+01:00. To avoid any date rollbacks when working with query date ranges in Integrity, PTC recommends that users specify the time zone that is used by the Integrity Server to which they are connecting.

- Because dynamically computed fields are not stored in the database, dynamically computed short text fields cannot be located with an all text field search in the Integrity Client. To search for dynamically computed short text fields, create a query that includes a specific \u201cfield contains\u201d comparison. For more information about computed fields, see your administrator.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--copyColumnsFromQuery[=[user:]query]`

  specifies a query to copy the column configuration from to use as the new default for the query you are editing.

Note

--fields=*field, field,...*,

--[no]sortAscending, and

--sortField=*field* are not mandatory, but if specified, they overwrite the current default column configuration.

- *--fields=field,field,...*

  specifies the fields to use as default columns for the query. This overrides the current default columns.
- `--[no]sortAscending`

  specifies the sort direction that issues are displayed in. This overrides the current default sort direction.
- `--sortField=field`

  specifies the field that issues are sorted by. This overrides the current default sort field.
- `--image=[none|default|<path>]`

  specifies whether an image appears for the new query.

  `--image=none` does not specify an image for the query.

  `--image=default` specifies the default funnel image for the query.

  `--image=<path>` specifies the path and name of a custom image for the query, for example, `c:\images\defect_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...`specifies the users and groups that can use and modify the query. Your administrator defines users and groups.
- `--[no]confirmSharedAdmin`

  specifies whether to confirm the conversion of the query to a system provided object.
- `--description=value`

  specifies a short description for the query.
- `--name=value`

  specifies the new name of the query. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--queryDefinition=query`

  specifies a string to define the query constraints. The query must be of the following format:

  <rule> is defined as (*<filtergroup>*)

  <filtergroup> is defined as one of the following:

  (*<filtergroup>* and *<filtergroup>* and ...) (*<filtergroup>* or *<filtergroup>* or ...)((*<filter>*) and (*<filter>*) and ...)((*<filter>*) or (*<filter>*) or ...)

  where

<filter> is defined as disabled (*<filter>*)*<filter>* is defined as not (*<filter>*)*<filter>* is defined as *<fields>*|*<subquery>*|*<attachment>*|*<relationship>*|*<branch>*|*<genericcp>*|*<histval>*|*<histdate>*|*<histuser>*|*<timeentry>*|*<testresult>*|*<item>*

<histuser> is defined as *histuser[Summary*|*State*|*..]* was changed by *<user>*

<histuser> is defined as *histuser.any* field was changed by *<user>*

<histdate> is defined as *histdate[Summary*|*State*|*..*] was changed *<datevalue>*

<histdate> is defined as *histdate.any* field was changed *<datevalue>*

<histval> is defined as *histval[Summary*|*State*|*..*] *<value>*

<genericcp> is defined as *genericcp*:*<cptype>*:*<attrfieldidentifier>*:*[fieldname]*

where *<attrfieldidentifier>* is attribute or *entryattribute* and *[fieldname]* is the real name, not the display name, of the attribute. Use the `im viewcptype` command to find the attribute name.

<genericcp> is defined as *not(genericcp:si:attribute[resolutionlist]is empty)*

<genericcp> is defined as *genericcp:<cptype>.exists*

<relationship> is defined as *relationship[ID*|*Created User*|*..]* using *[Relationship Field]=<value>*

<relationship> is defined as *relationship.exists backward*|*forward* using *[Relationship Field]*. To restrict your query to either backward or forward relationships, *backward* must be specified if you specify the Backward Relationships field, and the *forward* option must be specified if you specify the Forward Relationships field or a custom relationship field. *<relationship>* is defined as *relationshipFlag [Relationship Flag Name] backward*|*forward* using *[Relationship Field]*

<attachment> is defined as *attachment[file size*|*file name*|*mime type]* <value>*<attachment>* is defined as *attachment.exists*

<timeentry> is defined as *timeentry[issue ID*|*user*|*entry date*|*source*|*duration*|*notes*|*created by*|*created date*|*modified by*|*modified date]<timeentry>* is defined as *timeentry.exists*

<testresult> is defined as *testresult.isrelatedto <testresult>* is defined as *testresult.exists <testresult>* is defined as *testresult.hasattachment <testresult>* is defined as *testresult.hasrelateditem <testresult>* is defined as *[ID]comparison* where *ID* can be one of *Verdict*, *Verdict Type*, *Modified By*, *Modified Date*, *Session ID*

<branch> is defined as *branch.isBranch<branch>* is defined as *branch.hasBranch*

<branch> is defined as *branch[X]* in parent contains Y. This filter finds items where field X of the branched from item has a specific value. In other words, an item is a child (is a branch), and the parent item has field X with value Y. Examples follow:

<branch> is defined as branch[X] in child contains Y. This filter finds items where field X of the branched item has a specific value. In other words, an item is a parent (has a branch), and the child item has field X with value Y. Examples follow:

- branch[ALM_Text] in parent contains "Project Description"
- branch[ALM_Actual Budget] in parent = 12
- branch[ALM_Actual Effort] in parent > 20

<item> is defined as *item.node*

<item> is defined as *item.content*

<item> is defined as *item.segment*

<item> is defined as *item.subsegment*

<item> is defined as *item.lockeddocument*

<item> is defined as *item.meaningful*

<item> is defined as *item.nonmeaningful*

<item> is defined as *item.shareditem*

<item> is defined as *item.group*

<item> is defined as *item.teststep*

<item> is defined as *item.testcase*

<item> is defined as *item.testsession*

<item> is defined as *item.testsuite*

<item> is defined as *item.live*

<item> is defined as *item.versioned*

`Note`: If document versioning is enabled, queries return all items (live and versioned) by default. To specify live or versioned items, include the `item.live` or `item.versioned` filter. For more information on document versioning, see the *PTC Integrity User Guide*.

<subquery> is defined as *subquery[Query1*|*Query2*|*...]*

<fields> is defined as *field[ID*|*Created User*|*Created Date*|*..]<value><fields>* is defined as *field.anyTextField*

If document versioning is enabled, you can specify the following conditions to display live and versioned items by ID:

- display a specific live item only. For example, "`(field[ID]=123)`" returns `123`.
- display a live item and all versions of the item. For example, "`(field[ID]=123 include versions)`" returns `123`, `123-1.0`, `123-1.1`, and `123-1.2`.
- display a range of live items. For example, "`(field[ID]>123 and 128)`" returns `124`, `125`, `126`, and `127`.
- display a specific versioned item. For example, "`(field[ID]=123-1.0)`" returns `123-1.0`.

Note

You cannot display a range of versioned items, for example, "`(field[ID]>123-1.0 and 128-1.0)`".

For more information on live and versioned items, see the *PTC Integrity User Guide*.

<value> is defined as *<value>* or *"is empty" <value>* is defined as *"is empty"<value>* is defined as *<leftrangeop> num* and *<rightrangeop> num<value>* is defined as contains *<text><value>* is defined as *<operator> num* and *<operator> num<value>* is defined as *<operator> num<value>* is defined as *hasSourceLinks*

<rightrangeop> is defined as < | <=*<leftrangeop>* is defined as | >=

value> is defined as = *<uservalue>*, *<uservalue>*, ..

<uservalue> is defined as *"me"* | *"unspecified"* or *"is empty"* | *user1* | *user2* | ...

<value> is defined as *<datevalue>*

<datevalue> is defined as between *mm/dd/yyyy* and *mm/dd/yyyy*<datevalue> is defined as between mm/dd/yyyy hh/mm/ss and mm/dd/yyyy hh/mm/ss (Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format; however, Integrity displays the time in 12 hour format. For example, specifying 13:56:45 displays the time as 1:56:45 PM.)*<datevalue>* is defined as in the *last*|*next num days*|*months*|*years<datevalue>* is defined as in the *last*|*next num days*|*months*|*years hours*|*minutes*|*seconds (Time is specified from 00:00:00 to 23:59:59 inclusive in 24 hour format;* however, Integrity displays the time in 12 hour format. For example, specifying *13:56:45* displays the time as *1:56:45 PM.)<datevalue>* is defined as *today*|*yesterday*|*tomorrow*

num is defined as .. | *-1* | *0* | *1* | ..

<operator> is defined as = | > | >= | <= | < | <>

For example: `(((field[Summary]contains"Hello")or(field[Assigned Group]="everyone"))and(attachment.exists))((field[Summary]contains"Requirement")or(field[Assigned Group]="Project Managers"))((field[Category]="Technical Requirement") or (field[Category]="System Requirement"))`

Note
To reference the original query in the new query so that any changes to the original query are reflected in the new query, define *<subquery>* as *subquery[OriginalQuery]*.

- `--queryDefinitionFile=`*value*

  specifies a file that contains the complete definition of the query. See `--queryDefinition` for the file format.
- `--sharedAdmin`

  specifies the query as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important:

  Once a user object is converted to a system provided object, you cannot revert it to a user object again.
- *[user:]query*

  specifies the name of the user who the query belongs to and the query name, for example, `jhoyt:"Cosmos Critical Defects"`. If you are editing a query you own, you do not have to specify the user name, but you must specify the query name.

Note
Integrity initially assumes that text before the colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` query created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the `jhoyt:CosmosDefects` query created by any user.

- branch[Project] in child=/Projects/Release2
- branch[Summary] in child contains "Some issue Summary"
- branch[ALM_Content] in child 10

See Also

- Commands: im copyquery, im createquery, im deletequery, im viewquery, im queries
- Miscellaneous: options

im editreport

edits an existing Integrity report

Synopsis

```
im editreport [--query=[user:]query] [--reportTemplate=value] [--reportTemplateFile=value] [--recipeParam=value] [--recipeParams=value] [--recipeParamsFile=value] [--shareWith=u=user1[:modify],user2[:modify],...;g=group1[:modify],group2[:modify],...] [--[no]confirmSharedAdmin] [--name=value] [--description=value] [--sharedAdmin] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [username:]report
```

Description

`im editreport` edits the properties of an Integrity report. You may only edit one report at a time, and only reports created by you are available for editing. For more information on reports, refer to the *PTC Integrity User Guide*.

For example,

```
im editreport --name="Docs Weekly Status Release 2.0" "Docs Weekly Status"
```

edits the `Docs Weekly Status` report to change its name to `Docs Weekly Status Release 2.0`.

Note the following:

- Reports can do more than just display field information. You can also perform arithmetic calculations between numeric fields, displaying the values in the report. For example, you can add up column totals or count the number of issues in a specific state. To perform these calculations, you create a computed expression. For more information on the syntax, operators, functions, and operations applicable to computed expressions, see your administrator or the *PTC Integrity Server Administration Guide*.
- You cannot create or edit a query while creating a report.
- A report can be edited by the user who created it. Principals (users and groups) that a report is shared to can edit it if they have edit permissions assigned to them by the report creator. A report can only be deleted by the user who created it or by the administrator.
- Because reports are based on queries, reports are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or contact your administrator.
- Symbolic dates in rules and queries are evaluated on the Integrity Client's time zone.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Creating deeply nested reports with a large number of inter-related issues can create extremely large reports and/or cause the Integrity Server to stop responding. When creating a report, take into consideration that the average number of links per issue and the number of levels in the report multiply the size of the report.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.
- Although the electronic signature fields `Signed By` and `Signature Comment` are only visible in an issue's history (if enabled by your administrator), you can report on the historical values by specifying the fields in the report.

Options

This command takes the universal options available `to all im commands`, as well as some general options. See the `options` reference page for descriptions.

- `--query=`*[user:]query*

  specifies the name of the query that defines the selection criteria for the report, and the user who created the query.
- `--reportTemplate=`*value*

  specifies the report template on which the report is based. For information on the report template format, see the *PTC Integrity Server Administration Guide*.

  Note
  This option can only be used with template-based reports, and is not supported for recipe-based reports.

- `--reportTemplateFile=`*value*

  specifies the file name that contains a report template. For information on the report template file format, see the *PTC Integrity Server Administration Guide*.

  Note
  This option can only be used with template-based reports, and is not supported for recipe-based reports.

- `--recipeParam=`*value*

  specifies a single parameter *key*:*value* pair.

  The report parameters (or attributes) determine what is displayed to users when creating and editing reports in the Report Wizard.

  The available list of report recipe parameters and attributes can be viewed for each report type using the `im viewreport` and `im reports` commands. For more detailed information, see the knowledge base in the Support section of the PTC Web site (http://www.ptc.com).
- `--recipeParams=`*value*

  specifies a semicolon separated list of report recipe parameters or attributes in `key:value` pairs.
- `--recipeParamsFile=`*value*

  specifies the file that contains the recipe parameters in the report.
- `--shareWith=u=`*user1[:modify],user2[:modify],...;*`g=`*group1[:modify],group2[:modify],...*

  specifies the users and groups that can use and modify the report. Your administrator defines users and groups.
- `--[no]confirmSharedAdmin`

  specifies whether to confirm the conversion of the report to a system provided object.
- `--name=`*value*
- specifies the new name of the report. Names may be a maximum of 100 characters and cannot contain square brackets.
- `--description=`*value*

  specifies a description for the report.
- `--sharedAdmin`

  specifies the report as a system provided object (objects within the Integrity object model that support solution definition and management, as well as workflow migration). For more information, see your administrator.

  Important: Once a user object is converted to a system provided object, you cannot revert it to a user object again.
- *[username:]report*

  specifies the report to edit and the user who created it. If you are editing a report you created, you do not have to specify the user name, but you must specify the report name. You can only edit reports that you created unless you are an administrator.

  Note
  Integrity initially assumes that text before the colon (:) is a user name and text after it is a report name. If Integrity fails to find a matching user name and report name, it searches for a report name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the

`CosmosDefects` report created by `jhoyt`. If Integrity cannot find the report and/or user, it searches for the `jhoyt:CosmosDefects` report created by any user.

See Also

- Commands: im createreport, im copyreport, im viewreport, im runreport, im deletereport, im reports
-
- Miscellaneous: options

im editstate

edits the properties of a state

Synopsis

```
im editstate [--name=value] [--description=value] [--image=[none|default|<path>] [--position=<number>|first|last|before:<name>|after:<name>] [--capabilities=MKSSI:OpenChangePackages,MKSSI:ChangePackagesUnderReview,MKSIM:TimeTracking,MKSTM:ModifyTestResult] [--overrideForType=type] [--removeOverride=description,image,capabilities] [--[no|confirm]removeDescriptionOverride] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]state
```

Description

`im editstate` edits the properties of a state for workflows and documents. For example:

```
im editstate --description="Initial state" --name="1st_State" Submit
```

changes the name of the *Submit* state to *1st_State*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the new name of the state. Names can be a maximum of 100 characters and cannot contain square brackets.
- `--description=value`

  specifies a description of the state.
- `--image=[none|default|<path>]`

  specifies whether an image appears for the state.
  - `--image=none` does not specify an image for the state.
  - `--image=default` specifies the default image for the state.
  - `--image=<path>` specifies the path and name of a custom image for the state, for example, `c:\images\state_icon.gif`.

Note

Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--position=<number>|first|last|before:<name>|after:<name>`

  specifies the position in the order of states.
- `--capabilities=MKSSI:OpenChangePackages,MKSSI:ChangePackagesUnderReview,MKSIM:TimeTracking,MKSTM:ModifyTestResult`

  sets the given state capabilities to allow open change packages, change packages under review, time entries, or modifications to test results.
- `--overrideForType=type`

  sets a type-specific override for the state through the selected type or an issue of the selected type. The customized value supersedes the global settings for the state attribute when referenced through the selected type or an issue of the selected type. You can only override the attributes for description, image, and capabilities. The value is the name of the type you want to apply the override to.

  To set an empty override, use the following syntax, leaving a blank space after the option for `description`, `image`, or `capabilities`:

  ```
  im editstate --overrideForType=<type name> --capabilities=<state name>
  ```
- `--removeOverride=description,image,capabilities`

  removes the specified type override for the state and reverts to the global state attribute defined for that type. Use this option as a comma-separated list to name each of the attributes you wish to remove. Acceptable values are `capabilities`, `description`, and `image`.

  For example, the following command removes an override for state capability for the `Approved` state in the `Defect` type:

  ```
  im editstate --overrideForType=Defect --removeOverride=capabilitiesApproved
  ```
- `--[no|confirm]removeDescriptionOverride`

  confirms removing the translations in all the locales for the overridden description.
- *state*

  specifies the name of state you want to edit.

See Also

- Commands: im createstate, im viewstate, im deletestate, im states
- Miscellaneous: options

im edittrigger

edits an event trigger

Synopsis

```
im edittrigger [--name=name] [--type=[scheduled|rule|timeentry|copytree|branch|label|testresult|lock|unlock]] [--runAs=user] [--
query=[user:]query] [--position=<number>|first|last|before:<name>|after:<name>] [--description=value] [--
frequency=[manual|hourly|daily|monthly]] [--script=filename] [--scriptParams=[arg=value[;arg2=value2...]]] [--
scriptTiming=pre|post|pre,post|none] [--assign=[field=value[;field2=value2...]]] [--rule=value] [--copyRule=trigger] [--
ruleFile=filename] [--hostname=server] [--user=value] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--
selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--
settingsUI=[gui|default]] [--status=[none|gui|default]]trigger
```

Description

`im edittrigger` edits an event trigger for workflows and documents. For example:

```
im edittrigger --query=RequestsForChange sendmail
```

changes the query for the *sendmail* trigger to *RequestsForChange*.

Caution

Inactive values are not accepted in trigger assignments. If you specify inactive values for a trigger assignment containing existing user, group, or project field values, the existing values are cleared. For multi-valued user and group fields, the entire trigger assignment definition is cleared even if you specified one inactive value.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=`*name*

  specifies the new name of the trigger. May be a maximum of 100 characters and cannot contain square brackets.
- `--type=[`*scheduled*`|`*rule*`|`*timeentry*`|`*copytree*`|`*branch*`|`*label*`|`*testresult*`|`*lock*`|`*unlock*`]`

  specifies the type of trigger. If you do not specify a type, `rule` is specified by default. If you specify `timeentry`, the values for `--rule`, `--ruleFile`, `--runAs`, `--frequency`, `--assign`, and `--query` are ignored.

  Note

  Scheduled event triggers are evaluated on the Integrity Server's time zone.

- `--runAs=`*user*

  specifies the user for a scheduled trigger. All modifications appear as though they were made by the specified user.
- `--query=`*[user:]query*

  specifies the name of the query to use for a scheduled trigger, and the username of the person who created the query.
- `--position=`*<number>*`|`*first*`|`*last*`|`*before:<name>*`|`*after:<name>*`

  specifies the position of the trigger in the run order.
- `--description=`*value*

  specifies description of what the trigger does.
- `--frequency=`*[manual|hourly|daily|monthly]*

  specifies the frequency of a scheduled trigger. May be the following:

  ```
  [manual]
  [hourly [start=[00:]mm]          [hours=00,01,...,23]]
  [daily [start=hh:mm]             [days=mon,tue,...]]
  [monthly [start=hh:mm]           [day=1-31] [months=jan,feb,...]]
  ```
- `--script=`*filename*

  specifies the filename of the script, for example, `scriptfile.js`. The script must be located in the following directory: `<installdir>/data/triggers/scripts`.
- `--scriptParams=`*[arg=value[;arg2=value2...]]*

  specifies the list of script arguments.
- `--scriptTiming=[pre|post|pre,post|none]`

  specifies if the script should be run pre, post, both pre and post on issue commit to the database, or no timing is associated with the script. If you do not specify an option, `none` is specified by default.
- `--assign=`*[field=value[;field2=value2...]]*

  specifies the list of field assignments.
- `--rule=`*<rule>*

  specifies the rule to associate with the trigger. For the rule syntax, see Specifying Rules on the options reference page.

  Note

  - To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *today*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can choose yourself by specifying "`me`". "`me`" is a symbolic user which refers to the currently logged in user. For example, you could create an event trigger that specifies Project issues can only be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
  - Configuration management project fields are invalid in event trigger rules.

- `--copyRule=`*trigger*

  copies the given trigger's rule as it exists at the time of the copy. This does not create a a pointer to another rule.
- `--ruleFile=`*filename*

  specifies a file containing the rule text. See `--rule` for notes on the command use and where to obtain file format information.
- *trigger*

  specifies the name of the trigger you want to edit.

See Also

- Commands: im createtrigger, im copytrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers, im echo
- Miscellaneous: options

im edittype

edits the properties of an issue type

Synopsis

```
im edittype [--addWordTemplate=displayName,templatePath=pathToFile[,description=description][,defaultEdit]] [--
removeWordTemplate=value] [--editPresentation=value] [--printPresentation=value] [--printReport=report] [--viewPresentation=value] [-
-name=value] [--image=[none|<path>] [--description=value] [--[no]enableRevision] [--majorRevisionRule=rulename] [--
minorRevisionRule=rulename] [--copyMajorRevisionRule=type] [--copyMinorRevisionRule=type] [--majorRevisionRuleFile=filename] [--
minorRevisionRuleFile=filename] [--documentClass=[none|segment|node|shareditem]] [--[no]duplicateDetectionMandatory] [--
duplicateDetectionSearchField=field] [--associatedType=type] [--defaultReferenceMode=[Reuse|Share]] [--
significantFields=field,field,...] [--position=<number>] [--[no]allowChangePackages] [--createCPPolicy=value] [--
versionEditFields=field,field,...] [--visibleFields=field:group,group,...[;...]] [--notificationFields=field,field,...] [--
stateTransitions=state:state:group|dynamic group,group|dynamic group,...[;...]] [--addLabelRule=rule] [--moveLabelRule=rule] [--
copyMoveLabelRule=type] [--moveLabelRuleFile=filename] [--properties=name:value:description[;...]] [--addLabelRuleFile=value] [--
branchRule=rule] [--branchRuleFile=value] [--branchFields=field,field,...] [--copyAddLabelRule=type] [--copyBranchRule=type] [--
copyCopyTreeRule=type] [--copyDeleteLabelRule=type] [--copyTreeRule=rule] [--copyTreeRuleFile=value] [--deleteLabelRule=type] [--
deleteLabelRuleFile=value] [--[no]enableDeleteItem] [--deleteItemRule=value] [--deleteItemRuleFile=value] [--copyDeleteItemRule=type]
[--[no]enableBranch] [--[no]groupDocument] [--[no]enableCopyTree] [--[no]enableLabel] [--
testRole=[none|testSession|testCase|testStep|testSuite]] [--[no]enableTestSteps] [--[no]enableTestResultRelationship] [--
testCaseResultFields=field,field,...] [--modifyTestResultPolicy=value] [--editabilityRule=rule] [--editabilityRuleFile=value] [--
copyEditabilityRule=type] [--copyFields=field,field,...] [--mandatoryFields=state:field,field,...[;...]] [--
addFieldRelationship=constraint] [--fieldRelationships=constraint[;constraint]] [--removefieldRelationship=constraint] [--
fieldRelationshipsFile=value] [--[no]showHistory] [--[no]showWorkflow] [--phaseField=field] [--[no]enableProjectBacking] [--
[no]enableTimeTracking] [--permittedAdministrators=u=user1,user2,...;g=group1,group2] [--permittedGroups=group,group,...] [--
[no]allowDocumentLocks] [--documentLockPolicy=value] [--documentUnlockGroup=group] [--[no]allowAdditionalLockFields] [--
additionalLockFieldsRule=rule] [--additionalLockFieldsRuleFile=filename] [--copyAdditionalLockFieldsRule=type] [--
additionalSegmentLockFields=field,field,...] [--additionalContentLockFields=field,field,...] [--[no]lockingRequired] [--
lockingRequiredRule=rule] [--lockingRequiredRuleFile=filename] [--copyLockingRequiredRule=type] [--user=name] [--hostname=server] [--
password=password] [--port=number] [(-?|--usage)] [(-F file|-selectionFile=file)][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--
hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]] [--status=[none|gui|default] type
```

Description

`im edittype` edits the properties of an issue type. For example:

```
im edittype --allowAttachments Defect
```

changes the *Defect* type to allow attachments.

Caution

A type can contain a maximum of 1000 fields. Exceeding the maximum number of fields results in exception errors when creating, editing, or viewing the issue type in the GUI.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--editPresentation=`*value*

  specifies the presentation template to use when editing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--printPresentation=`*value*

  specifies the presentation template to use when printing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--printReport=`*report*

  specifies the administrator report that will provide the structure and format when a user prints a document using **Document ▸ Print** from the GUI. This must be specified with `--documentClass=`*segment*.
- `--viewPresentation=`*value*

  specifies the presentation template to use when viewing this type. Presentation templates allow you to customize issue presentation. For more information on presentation templates, see the *PTC Integrity Server Administration Guide*.
- `--name=`*value*

  specifies the name of the type. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
  - `--image=[`*none*`|<`*path*`>]` specifies whether an image appears for the type.
  - `--image=`*none* does not specify an image for the type.
  - `--image=<`*path*`>` specifies the path and name of a custom image for the type, for example, `c:\images\type_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=`*value*

  specifies a description of the type.
- `--[no]enableRevision`

  specifies whether to enable item revisioning for the specified item type or enable document versioning for the specified document model type. For more information on item revisioning or document versioning, see the *Integrity Server Administration Guide*.
- `--majorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the major revision. For example, if the major rule is `Type=Requirement`, then only items of type Requirement may have major revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--minorRevisionRule=`*rulename*

  specifies a rule that must be true for the item type (used with item revisioning) or document model type (used with document versioning) in order to increment the minor revision. For example, if the minor rule is `State=Open`, then only items in an open state may have minor revisions incremented. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyMajorRevisionRule=`*type*

  copies the major revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one being created.
- `--copyMinorRevisionRule=`*type*

  copies the minor revision rule of an existing item type (used with item revisioning) or document model type (used with document versioning) to the one being created.
- `--majorRevisionRuleFile=`*filename*

  specifies a file that contains the major revision rule for the specified item type (used with item revisioning) or document model type (used with document versioning). For the file format, see Specifying Rules on the options reference page.
- `--minorRevisionRuleFile=`*filename*

specifies a file that contains the minor revision rule for the specified item type (used with item revisioning) or document model type (used with document versioning). For the file format, see Specifying Rules on the options reference page.

- `--documentClass=[none|segment|node|shareditem]`

  allows you to specify a document class on a type. Each document domain requires a segment, a node type associated with the segment, and a shared item type associated with the node type. Each document instance requires the segment root and a set of nodes. The options are as follows:

  - `none` specifies that this item type is not used in documents. The default for all non-document types.
  - `segment` specifies that this type is used in the segment root. For example, you need to specify segment if you are creating a document domain.
  - `node` specifies that this type is used in nodes (content). Node correlates to the content (i.e. Requirement, Input, Test, Specification) item type. To create content, you need a node and a corresponding shared item. Therefore, you would create two types: one with `documentClass=node`, and the other with `documentClass=shared item`. Each node in a document is a reference to a shared item or a subsegment, and all three types expose arbitrary fields. Nodes also expose FVA fields from the shared item.
  - `shared item` specifies that this type is a shared item. Shared items should be associated with a corresponding node type (content). Each node class on a type requires an associated shared item.

    Example:

    ```
    im createtype --name=Document --description="Requirement Document"--documentClass="segment" --associatedType="Requirement(node name)"
    ```

    creates a segment with an associate node type called `Requirement`.

- `--[no]duplicateDetectionMandatory`

  makes it mandatory for users to view potential duplicates before they create a new item.

- `--duplicateDetectionSearchField=field`

  specifies the name of the short text field to search when using duplicate detection. Setting an empty field means that duplicate detection will not be available to users. The short text field cannot be a computational field or Field Value Attribute to a short text field. Before enabling duplicate detection, the selected short text field must be visible on the type. In addition, to use duplicate detection searches on a short text field, the created field must have text search indexing enabled. When searching for potential duplicates, the default number of results is 20 items.

- `--testRole=[none|testSession|testCase|testStep|testSuite]`

  allows you to specify the test management role for the type. The options are as follows:

  - `none` specifies that this item type does not a have a specific test management role. However, it can still be related to test results using the `--[no]enableTestResultRelationship` option.
  - `testSession` specifies that this type is a test session. A test session is a concrete run or execution of a group of test cases. If a type is a test session, also specify `modifyTestResultPolicy`.
  - `testCase` specifies that this type is a test case. A test case describes the test conditions and provides a container for adding test results from the test session. A test case must have the `--documentClass` option set to node. If a type is a test case, also specify `--[no]enableTestSteps` and `--testCaseResultFields`.
  - `testStep` specifies that this type is a test step. A test step is a specific testing operation performed as part of executing a test case.
  - `testSuite` specifies that this type is a test suite. A test suite is a grouping of test cases. A test case must have the `--documentClass` option set to segment.

- `--associatedType=type`

  specifies an associated type for segments and nodes. If the document class is `segment`, the associated type is of type node. If it is `node`, the type is of `shared item`.

- `--defaultReferenceMode=[Reuse|Share]`

  specifies the reference mode to apply to nodes after they are branched. `Reuse` points to the same shared item until the original node changes and a new shared item is created.`Share` points to the same shared item as the original branched node. Editability will not be allowed on shared fields; this mode only allows observation of the changes made on the other item.

- `--significantFields=field,field,...`

  allows you to specify additional fields by type that, when edited, results in an update to the content revision. This is reflected in a change to the revision date in the item history. Each type has default associated significant edit fields.

- `--position=<number>]`

  specifies the position in the list of types.

- `--addLabelRule=rule`

  specifies the rules for when users are permitted to add a label rule. For the rule syntax, see Specifying Rules on the options reference page.

- `--addLabelRuleFile=filename`

  specifies a file that contains the add label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page.

- `--moveLabelRule=rule`

  specifies the rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--copyMoveLabelRule=type`

  specifies the type from which to copy rules for when users are permitted to move a label. For the rule syntax, see Specifying Rules on the options reference page.

- `--moveLabelRuleFile=filename`

  specifies a file that contains the move label rule set for the specified issue type. For the file format, see Specifying Rules on the options reference page.

- `--branchRule=rule`

  specifies the rules for branching by users. For the rule syntax, see Specifying Rules on the options reference page.

Note

Users or groups do not require editability of the issue or any given field to be able to branch an issue; however, they must have project or type visibility to branch an issue. If a user does not have visibility to a given field during a branch operation, the field will be copied unchanged.

- `--branchRuleFile=filename`

  specifies a file that contains the branch rule set for the specified issue type. For the file content format, see Specifying Rules on the options reference page.

- `--branchFields=field,field,...`

  specifies the list of fields to make visible on the Branches tab for viewing item branches.

Note

The branched fields that can be configured are only visible fields on the item type. This means that field visibility permission is checked when adding a branched field. The `--branchFields` option can be used with `--visibleFields=field:group,group,...[;...]` for each command.

- `--copyAddLabelRule=type`

  specifies the rules for when users or groups are permitted to copy an add label rule.

- `--copyBranchRule=type`

  specifies the rules for when users are permitted to copy a branch rule.

- `--copyCopyTreeRule=type`

specifies the rules for when users or groups have the ability to copy the copy tree rule for the specified issue type.

- `--copyDeleteLabelRule=`*type*

  copies the delete label rule of an existing type to the one being edited.

- `--[no]enableBranch`

  specifies whether issues of this type can be branched. For more information on branching, see the *PTC Integrity User Guide*.

- `--[no]enableCopyTree`

  specifies whether users or groups can copy a tree of issues belonging to the hierarchy for this type.

- `--[no]enableLabel`

  specifies whether users or groups can add labels to issues of this type.

- `--copyTreeRule=`*rule*

  copies the tree rule of an existing type to the one being edited.

- `--copyTreeRuleFile=`*value*

  specifies a file that contains the copy tree rule set for the specified issue type.

- `--deleteLabelRule=`*type*

  specifies the rule for when users or groups can delete label rules.

- `--deleteLabelRuleFile=`*value*

  specifies a file that contains the delete label rule set for the specified issue type.

- `--[no]enableDeleteItem`

  specifies whether items of the specified type can be deleted. If items of the specified type can be deleted, a rule set must specified with `--deleteItemRule=`*value* or `--deleteItemRuleFile=`*value*.

- `--deleteItemRule=`*value*

  allows specific users or groups the ability to delete items of the specified item type. To change the rule on who can delete items, users or groups require the `ModifyDeleteItemRule` permission.

Caution

The `ModifyDeleteItemRule` permission differs from the `DeleteItem` permission, which allows users or groups to delete items of any type without a defined rule. To define strict item deletion rules in your organization, PTC recommends defining a delete item rule and assigning the `ModifyDeleteItemRule` permission to the appropriate users and groups.

- `--deleteItemRuleFile=`*value*

  specifies a file that contains the rule set for deleting items of the specified item type.

- `--copyDeleteItemRule=`*type*

  specifies a type to copy an existing delete rule set from.

- `--[no]allowChangePackages`

  specifies whether configuration management change packages are permitted for the type.

- `--[no]groupDocument`

  specifies whether items of the specified type will be used to group content. For group documents, the `--documentClass` option must be `segment`.

- `--[no]enableTestSteps`

  specifies whether the test case type can use test steps. A test step is a specific test for a test case. A test case can contain an ordered list of steps to follow in order to complete the test. Test steps can be shared across test cases.

- `--[no]enableTestResultRelationship`

  specifies whether the type can be related to test results. For example, defects for failed test results.

- `--testCaseResultFields=`*field,field,...*

  specifies fields from the test case type that will display in the Test Result Details view.

- `--modifyTestResultPolicy=`*value*

  specifies which users and/or groups are allowed to modify test results for the test session item type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to the test session are permitted to create test results. Valid values are `userField=<`*field*`>`, `groupField=<`*field*`>`, *anyone*, `groups=`*group1,group2,...*.

- `--createCPPolicy=`*value*

  specifies which users and/or groups are allowed to create change packages against issues of this type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to issues within that type are permitted to create change packages. Valid values are `userField=<`*field*`>`, `groupField=<`*field*`>`, *anyone*, `groups=`*group1,group2,...*.

- `--versionEditFields=`*field,field,...*

  specifies the fields that users will be able to edit on document and content versions. For document and content versions, you can allow editing only on pick fields and relationship fields. Fields that are included in the `--significantFields` list may not also be included in `--versionEditFields` list. Conversely, fields that are included in the `--versionEditFields` list may not also be included in `--significantFields` list. For more information on configuring editable fields for document versions, see the *PTC Integrity Server Administration Guide*.

- `--visibleFields=`*field:group,group,...[;...]*

  specifies which groups have visibility for which fields. By default, the everyone group is specified for each field you specify. All previous values are replaced with specified values.

Note

Special fields are always visible, even if they are not specified.

- `--notificationFields=`*field,field,...*

  specifies notification fields for including in every email notification related to this type.

Note

SI project and attachment fields cannot be specified.

- `--copyFields=`*field,field,...*

  specifies the list of fields to be copied by default for items of this type. This setting overrides `--copyCommonFields` if set. Users can override these default fields by adding and removing any fields that they can view and edit.

- `--stateTransitions=`*state:state:group|dynamic group,group|dynamic group,...*`[;...]`

  specifies a state transition from one state to another, and the groups/dynamic groups permitted to make that state transition. At least one group/dynamic group must be specified for each transition.

- `--properties=`*name:value:description*`[;...]`

  defines type properties. Type properties provide custom code (such as triggers and API) with attributes to read and act on based on their values. The following can be specified for each property:

  - *name* specifies the name of the property.

- *value* specifies the value for the property.
- *description* specifies an optional description for the property.

- `--editabilityRule=`*rule*

specifies the rules for when users are permitted to edit the type. For the rule syntax, see Specifying Rules on the options reference page.

Note

- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date and a time of 00:00:00 (midnight) for a date field, type *today*. This option can be specified only if the date field is configured to display the date. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
- When specifying a user, you can choose yourself by specifying "me". "me" is a symbolic user which refers to the currently logged in user. For example, you could create an editability rule that specifies a Project type issue can be edited if the currently logged in user is one of the users defined in the multi-valued `Stakeholders` field.
- SI project and attachment fields cannot be specified.
- If a group name contains blank spaces, a second pair of quotes around ""group name"" is required. For example:

```
im editfield --hostname=server --port=7001 --editabilityRule="((user is a member of "Project Management") or (user is a member of ""Project Management""))\" fieldname
```

- `--editabilityRuleFile=`*filename*

specifies a file that contains the issue editability rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.

- `--copyEditabilityRule=`*type*

copies the issue editability rule of an existing type to the new one being created.

- `--mandatoryFields=`*state:field,field,...*`[;...]`

specifies mandatory fields for a state in the type's workflow. All previously specified mandatory fields are replaced with the new values. Mandatory fields can also be set on type constraints. For more information on constraints, see the *PTC Integrity Server Administration Guide*.

- `--addFieldRelationship=`*constraint*

specifies a single constraint (field relationship) to be added to a type.

- `--removeFieldRelationship=`*constraint*

specifies a single constraint (field relationship) to be removed from a type.

- `--fieldRelationships=`*constraint[;constraint]*

specifies constraints between fields. All previously existing constraints on this type are replaced with the new constraints specified here. The syntax specified below also applies for the `--addFieldRelationship` and `--removeFieldRelationship` options.

There are four constraint methods: Basic, Field Relationship, Rule, and Item Backed Pick List (IBPL). For more information on constraints and constraint methods, see the *PTC Integrity Server Administration Guide*.

Caution

When creating or copying a type, you cannot create a basic constraint.

where *constraint* is one of:

a Basic or Field Relationship constraint:

sourceField=sourceValue1[,sourceValue2,...]:targetValue`[:[all][,mandatory][,errInvalidated=`*invalidMessage*`]`
`[,errMandatory=`*mandatoryMessage*`][,description=`*descriptionText*`]]`

a Rule constraint:

`constraintrule=(`*rule*`):`*targetValue*`[:[all][,mandatory][,errInvalidated=`*invalidMessage*`][,errMandatory=`*mandatoryMessage*`]`
`[,description=`*descriptionText*`]]`

an IBPL constraint:

`rule=(`*ibplRule*`):ibplField[:[mandatory][,errInvalidated=`*invalidMessage*`][,errMandatory=`*mandatoryMessage*`]`
`[,description=`*descriptionText*`]]`

and where *sourceField* is any field with predefined values (Pick, User, Group, IBPL, Boolean, Project, State, Phase, Type). For the basic constraint method, *sourceField* must be the type you are editing, and the source value must be the specified type.

and where *targetValue* is:

targetField=[value1][,value2,...] where *targetField* is a field of type other than User/Group.

or

targetField=[value1][,value2,...]`[,hasProjectPermission]` where *targetField* is a field of type Group.

or

targetField=[value1][,value2,...]`[,hasProjectPermission][memberOf(`*dynamicGroup1[,dynamicGroup2,...]*`)][valueOf(`*group*`)]` where *targetField* is a field of type User.

---

targetField can be any field that is visible on the type and that is not the *sourceField*. Only the following fields can have values specified: User, Group, Pick, State, Project, IBPL, Logical. All other fields (such as Attachment, Integer, Short Text, etc...) can only be made mandatory.

Specifying values (*value1, value2,* etc...) is optional; however, if not specified, the `all` and `mandatory` options must be used to indicate that the field is mandatory and all values are acceptable.

Note

For all *targetField* types, if a mandatory option is specified, then at least one value (*value1,value2, ...*) must be specified, or the `all` option must be present.

`hasProjectPermission` constrains the values of the field to only those users or groups that have permissions for the item's project. This option cannot be specified with any values (`value1, value2...`), or with `memberOf` or `valueOf` options.

`memberOf` constrains the values of the field only to those users that are a members of the listed dynamic groups (*dynamicGroup1, dynamicGroup2, ...*). Note that the `everyone` group can also be used as a value for the dynamic groups. This option cannot be specified with any values (*value1, value2...*), or with `hasProjectPermission` or `valueOf` options.

`valueOf` constrains the values of the field to only the specified group. This option cannot be specified with any values (*value1, value2...*), or with `hasProjectPermission` or `memberOf` options.

and where *rule* is a specified rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

---

and where *ibplRule* is a specified IPBL rule. For the rule syntax, see "Specifying Rules" on the options reference page.

Note

The IBPL rule can be made up of:

- sub-rules on the field values of the items backing the IBPL target. In the CLI, add an apostrophe (') at the end of the field to indicate the Target Field option in the GUI.
  - sub-rules on the item being edited. If no apostrophe is added, then the 'Editing Item Value' GUI option is selected.

In addition:

- Attachment fields, text fields, relationships fields, and dynamic computed fields cannot be specified.
- If one field is a date type field, then the other field must also be a date type field.

and where *ibplField* is an IBPL field that is visible on a type.

and where *invalidMessage* is any string up to 2000 characters.

and where *mandatoryMessage* is any string up to 2000 characters.

and where *descriptionText* is any string up to 200 characters.

Examples:

Basic Constraint Method:

```
Type=AdminRequest:"Assigned User"=memberOf(manager,administrator):mandatory,errMandatory="The {ConstrainedField} cannot be empty."
```

Specifies that all items of the `AdminRequest` type must be assigned to a user who is either in the manager or an administrator group, and that the "Assigned User" field is mandatory.

Field Relationship Constraint Method:

```
Importance=High,Critical:Escalated=True:mandatory,description="Ensure that the Incident is escalated if Importance is either High or Critical".
```

Specifies that an incident is escalated if `Importance` is either `High` or `Critical`.

Rule Constraint Method:

```
constraintrule=(field[importance]>"8"):"Assigned User"=valueOf(manager):mandatory
```

Specifies that if the item is very important (importance > 8), then assign the item to a manager.

IBPL Constraint Method:

```
rule=(field["State"] = "Prioritized"):Priority
```

Specifies that if the state is `Prioritized`, then the `Priority` IBPL is visible.

```
rule=(field'["Graduated"] = true):Students
```

Filters the list of students such that the only values displayed are those where the student's backing item has `Graduated` set to true.

Note

You can specify the options for `--fieldRelationships`, `--addFieldRelationship`, and `--removeFieldRelationship` at the same time. If all three options are specified at the same time, the constraints on the type are modified in the following order:

1. `--fieldRelationships` is executed first and all constraints are replaced by the ones specified here.
2. `--removeFieldRelationship` is executed next and the specified constraints are removed from the ones specified in the `--fieldRelationships` option.
3. `--addFieldRelationship` is executed last and the specified constraints are added to the type.

The options for `--removeFieldRelationship` and `--addFieldRelationship` can be specified multiple times on the command.

Note

If you need to specify a large number of constraints (field relationships), use the `--fieldRelationshipsFile` option.

- `--fieldRelationshipsFile=`*value*

specifies the name of the file containing the constraint. The file should use the same format as the `--fieldRelationships` option, with all constraints on the same line and separated by semi-colons. For information on constraints, see the *PTC Integrity Server Administration Guide*.

Note

If you specify both `--fieldRelationships` and `--fieldRelationshipsFile`, only `--fieldRelationships` is used.

- `--[no]showHistory`

specifies whether to display the item history for all items of the selected type. When you set `--noshowHistory` for an existing type, the item history is always hidden from users when they are viewing or editing items of that type. The setting applies in all Integrity Client interfaces. For example, in the Integrity Client GUI and Web interface, when you set the `--noshowHistory` option, the **History** tab is no longer displayed for items of the selected type. By default, the item history is displayed for newly created types.

- `--[no]showWorkflow`

specifies whether to display the **Workflow** tab to the user in the **Create Issue** view, **Edit Issue** view, and **Issue Detail** view. The **Workflow** tab contains a read-only display of the issue type workflow to the user. Users in the Web interfaces do not have a user preference to override the display of the **Workflow** tab. Users launching the GUI view from the CLI can override the display of the **Workflow** tab on a per command basis, if workflow for that type is enabled.

- `--phaseField=`*field*

specifies the phase field to display in the **Workflow** tab of the **Create Issue** view, **Edit Issue** view, and **Issue Detail** view. Only a phase field that is specified in the `--visibleFields` option may be specified. This option must be specified with the `--showWorkflow` option.

- `--[no]enableProjectBacking`

Enables the type to back a project, allowing you to create a link between an issue of this type and a project in the `Project` field. By creating a Project issue type and specifying this option, then creating a Project issue and linking it to a specific project (using the `im createissue` command), the power and workflow of projects as issues becomes available. Important metadata and metrics can be recorded in the Project issue, for example, the assigned Project Manager, estimated and actual budgets (using computed fields), and important milestone dates. Linking a Project issue to a project also reduces possible confusion about the details of projects in your database.

Note

- A link between a Project issue and a project is optional.
  - If you enable this option, creating issues of this type and linking them to projects is useful only to certain users. You may want to prevent most users from being able to create issues of this type, for example, you can allow only users in the **ProjectManagers** group to create Project issues. To restrict type visibility, see the `--permittedGroups` option.
  - Projects and Project issues can be used independent of one another; you do not have to create a Project type to use the **Project** field.
  - Multiple types can back projects; however, only one issue may back a given project.

- To enable this option, you must be an administrator for the specified type. This option can be disabled at any time.
- This option cannot be specified unless the **Project** field is specified as a visible field for this type.
- To create the issue that backs a project, you must be an administrator for the specified project and belong to a group that has permission to create issues of that type.
- When you create an issue to back a project, Integrity warns you if there is more than one type that can back a project, displaying a list of types that have this option enabled and that you have "view" and "modify" permissions for. Specify the type that you want to back the project.
- For Admin Staging, you cannot stage issues. Issues that back projects created on a Staging server will not be staged; you must re-create the issues that back projects on the production server. Rules, queries, or computed fields that explicitly mention the issue number that backs a project will not work after they have been transferred from the staging server to the production server.

- `--[no]enableTimeTracking`

  Allows one or more users to allocate time spent working on issues of this type. When enabled, users can specify time entries when they edit an issue. In addition, a **Time Entries** tab is available when you view an issue of this type. You can use time entries to develop metrics (in the form of queries, charts, and reports) that measure the amount of effort spent on projects.

Note

- To create, edit, and delete time entries on behalf of other users, the `TimeTrackingAdmin` ACL permission is required.
- The ability to create, edit, and delete time entries is governed by state-based capabilities. For more information, see the `im createstate` and `im editstate` commands.

- `--permittedAdministrators=u=user1,user2,...;g=group1,group2,...`

  specifies a comma delimited list of users and/or groups that can administer this type. Integrity Type Administrators are allowed to edit and view types. Type Administrators are not allowed to assign themselves as administrators to any other types, or to edit types that they don't administer. A Type Administrator can create fields, but can only edit fields that are referenced by a type they administer. For more information, see the *PTC Integrity Server Administration Guide*.
- `--permittedGroups=group,group,...`

  specifies a comma delimited list of groups that have visibility for this type.
- `--addWordTemplate=name=templateName,path=templatePath[,description=description][,defaultEdit]`

  specifies a template to add to the type for users to use when editing items in Microsoft® Word. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
  - *name*

    specifies the name of the template. The name is visible to users if more than one template is available for selection.
  - *path*

    specifies the path of the template to add to the type. The file must be DOTX format.
  - *description*

    specifies an optional description for the template.
  - *defaultEdit*

    specifies to use the template as the default template for users editing an item or document in Word. Specifying this option pre-selects the template for the user. However, the user may still select a different template if needed. This option can be used to streamline the edit process for users. Only one template per type can have this option enabled. Enabling this option on one template clears it from any other that has it specified.
- `--removeWordTemplate=value`

  specifies a Microsoft® Word template to remove from the type. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.
- `--[no]allowDocumentLocks`

  specifies whether document locking is supported for the specified type.
- `--documentLockPolicy=value`

  specifies which users and/or groups are allowed to lock documents of the specified type. The default value for the policy is `userField=assignedUser`, that is, only users who are assigned to documents of this type are allowed to lock them. Valid values are `userField=<field>`, `groupField=<field>`, *anyone*, `groups=group1,group2,...`.
- `--documentUnlockGroup=group`

  specifies the lock administration group for the type. Members of the specified group can unlock any locked document of the specified type.
- `--[no]allowAdditionalLockFields`

  specifies whether additional fields (beyond significant fields) can be locked for the specified type. By default, only significant fields are affected by document locking.
- `--additionalLockFieldsRule=rule`

  specifies a rule that determines when additional fields (if allowed) are affected by locking. For the rule syntax, see Specifying Rules on the options reference page.
- `--additionalLockFieldsRuleFile=filename`

  specifies a file that contains the additional locks field rule set for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyAdditionalLockFieldsRule=type`

  copies the additional field locks rule of an existing type to the new one being created.
- `--additionalSegmentLockFields=field,field,...`

  specifies the fields on a segment root of the specified type that, in addition to the significant fields, are affected by document locking.
- `--additionalContentLockFields=field,field,...`

  specifies the fields on content nodes of the specified type that, in addition to the significant fields, are affected by document locking.
- `--[no]lockingRequired`

  specifies whether a document of the specified type must be locked before the significant fields (plus any defined additional fields) can be edited.
- `--lockingRequiredRule=rule`

  specifies a rule that defines when documents of the specified type must be locked before the appropriate fields can be edited. For the rule syntax, see Specifying Rules on the options reference page.
- `--lockingRequiredRuleFile=rule`

  specifies a file that contains the rule that defines when locking is required for the specified type. For the rule syntax, see Specifying Rules on the options reference page.
- `--copyLockingRequiredRule=rule`

  copies the rule that defines when locking is required for an existing type to the new type being created.
- `--hiddenMenus=[None|CreateItem|CreateRelatedItem|CreateDocument|InsertContent]`

  specifies the menus and related dialog boxes where creation or insertion of the item type is to be hidden. When an item type is hidden, it can still be created or inserted from the command line. The options are as follows:
  - *none* specifies that the item type is not to be hidden from any menu. This is the default. No other option can be specified along with this option. When item types are to be hidden from menus, multiple options can be specified.
  - *CreateItem* specifies that this item type is to be hidden from menus and dialog boxes for creating items.

- *CreateRelatedItem* specifies that this item type is to be hidden from menus and dialog boxes for creating related items.
- *CreateDocument* specifies that this item type is to be hidden from menus and dialog boxes for creating documents.
- *InsertContent* specifies that this item type is to be hidden from menus and dialog boxes for inserting content in documents.

- *type*

  specifies the name of the type you want to edit.

See Also

- Commands: im createtype, im viewtype, im deletetype, im types, im copytype
- Miscellaneous: options

im edituser

edits a user

Synopsis

`im edituser [--name=value] [--description=value] [--image=[none|default|<path>]] [--fullName=value] [--email=value] [--notificationRule=rule] [--notificationRuleFile=value] [--copyNotification=[user|group]:<name>] [--[no]active] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] user`

Description

`im edituser` edits a user for workflows and documents. You can edit the following user details: name, full user name, email address, status, image, description, and notifications. For example:

```
im edituser --description=MarketingManager jriley
```

changes the description for *jriley*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the new name of the user. Names can be a maximum of 100 characters and cannot contain square brackets.

  Note

  PTC recommends you do not use commas, colons, or special characters in user names. If you use commas in the user name, you cannot select the user in a multi-valued user field. If you use colons in the user name, the user cannot edit shared system provided objects (charts, queries, reports, and dashboards).

- `--description=value`

  specifies a description of the user.

- `--image=[none|default|<path>]`

  specifies whether an image appears for the user.

  - `--image=none` does not specify an image for the user.
  - `--image=default` specifies the default "person" image for the user.
  - `--image=<path>` specifies the path and name of a custom image for the user, for example, `c:\images\defect_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--fullName=value`

  specifies the full name of the user.

- `--email=value`

  specifies the email address of the user.

- `--notificationRule=rule`

  specifies the notification rule for this principal. For the rule syntax, see Specifying Rules on the options reference page.

  Note

  - To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date field, type *today*. This option can be specified for a date field or a date field configured to display the date and time. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
  - When specifying a user, you can specify the user's name or "`me`". "`me`" is a symbolic user which refers to the user that the notification rule is created for. This is useful if you want to create a common notification rule and share it with other users.
  - E-mail notification is subject to project, type, and field visibility rules. Only users that have visibility for a given project and type receive e-mail notification for issues related to that project and type. In addition, e-mail notifications include only the fields they have permission to view.

- `--notificationRuleFile=value`

  specifies the file that contains the rules to read. For the file content format, see Specifying Rules on the options reference page.

- `--copyNotification=[user|group]:<name>`

  copies the given user/group's notification rules. This command copies the given user/group's notification rule into this user's at the time that the command is run.

- `--[no]active`

  Specifies if the user is actively used in Integrity. This option is useful for removing users that no longer exist in your organization. Specifying `--noactive` prevents the user from being displayed as an available default user field value; however, the user still appears in existing issue data.

  Note

  If the user is referenced in a trigger assignment or as the default value in a user field, moving the user from active to inactive displays an error message. The reference must be removed before making the user inactive.

- *user*

  specifies the name of user you want to edit.

See Also

- Commands:

  im createuser, im viewuser, im deleteuser, im users

- Miscellaneous:

  options

im exit

exits the current Integrity Client session

Synopsis

```
im exit [--[no]abort] [--[no|confirm]shutdown] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

DESCRIPTION

`im exit` exits the current Integrity Client session. When you run any Integrity command from the CLI, or when you open the Integrity Client GUI or Web interface, you start a client session. Only one client session is running at a time, regardless of how many GUI windows you have open or how many CLIs you are using.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]abort`

  controls whether to shut down any other Integrity commands that may be running. Some commands allow you to specify a `--persist` option that keeps those commands active during a client session. Using `--abort` with `im exit` is recommended for stopping all persistent views that have been specified with another command's `--persist` option.
- `--[no|confirm]shutdown`

  controls the shutting down of the Integrity Client without getting a prompt.

Note

Note:

Specifying `--noshutdown` with `im exit` is essentially a non-operation: it does nothing.

SEE ALSO

- Commands: im about, im connect
- Miscellaneous: options

im exportissues

exports query results to Microsoft Excel

Synopsis

```
im exportissues[--[no]exportHeadings] [--[no]openOutputFile] [--outputFile=value] [--[no|confirm]overwriteOutputFile] [--
[no|confirm]trimExcessColumns] [--fieldFilter=value] [--[no]sortAscending] [--sortField=field[:asc|desc]] [--[no]substituteParams] [-
-asOf=[<date> |label:<label >] [--fields=field[:width[:rich|plain],field[:width[:rich|plain],...] [--height=value] [--
[no]showTallRows] [--structureFieldDisplayFormat=value] [--[no]showDecorators] [--width=value] [-x value] [-y value]
[query=[user:]query...] [--queryDefinition=query] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)]
[(-F value|--selectionFile=value )] [(-N|--no)][(-Y|--yes)][--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--
quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] item id...
```

Description

`im exportissues` exports the visible item fields and columns returned by a query to Microsoft Excel for further analysis. You can also export items as of a historical date or label.

When you export items, Integrity saves the output as a temporary Excel file (`.xls`) that you can rename and save.

Once the temporary file open in Excel, you can then manipulate the data to create charts and reports for further analysis.

Important:

Once you export data to Excel, you cannot import the data into Integrity. Exporting query results to Excel is intended for data analysis only. To export and import data between Integrity and Excel, use the Microsoft Excel integration with Integrity.

Key Considerations:

- Integrity supports exporting data to Microsoft Excel 2003 and 2007.
- In Excel, the column names display as the first row. The data displays in the following rows.
- Sort order and filters applied to items are maintained in the exported data.
- If you exit Integrity without saving the Excel file, the temporary file is deleted.
- Before you export query results, note that Excel contains the following data limitations:
  - maximum 32,767 characters per cell
  - maximum 65,536 rows per sheet
  - maximum 256 columns per sheet

  When the row limit is exceeded, data is split across sheets.

  Long text fields that exceed the character limitation per cell are truncated, with "[Truncated]" appended to the end of the cell.

  For a complete list of Excel specifications and limitations, browse to:

  http://office.microsoft.com/en-us/excel/HP051992911033.aspx
- Integrity's date/time fields format is not supported in Excel. Instead, Excel uses `MM/dd/yyyy` for date fields and `MM/dd/yyyy HH:mm:ss` for date/time fields. For example, the date/time field `Jul 22, 2009 5:15:14 AM` in Integrity displays as `07/22/2009 05:15:14` in Excel.
- The following Integrity display patterns are supported in Excel:
- "#,###"
- "#,###.##"
- *currency symbol* + "#,###"
- *currency symbol* + "#,###.00"
- "#,###;(#,###)"
- "#,###.##;(#,###.##)"
- "0.###E0" (displays as "0.###E+0" in Excel)
- "#%"

  Integrity display patterns that use exclusive text are not supported in Excel. Any unsupported display pattern is disregarded; however, a standard format of "0" for integer and "0.00" for floating-point fields is applied.
- In the GUI, column widths from the view specify column widths in Excel. In the CLI, default column widths specify column widths in Excel.
- If wrapping is enabled for table content in Integrity, cell content is wrapped in Excel.
- In the GUI, Excel starts automatically. From the CLI, you can specify whether to start Excel.
- If no items are exported, Excel does not start, for example, if you attempt to export an item that you do not have permission to view.
- For relationships, the Relationship Flags and Order fields are not exported. For documents, the Section field is not exported.
- The following describes the format of each Integrity field after exporting to Excel:
- ID fields display as a number (0 decimal places, no other formatting).
- Floating-point and integer fields display as numbers with display patterns.
- State, type, project, phase, range, SI Project, short text, long text, user, logical, and FVA fields display as text.

Note
For long text fields, rich content formatting does not display.

- Attachment fields display as a CSV list of file names.
- Relationship fields display as a CSV list of ID/relationship flag rules.
- IBPL and QBR fields display as a CSV list of IDs.
- Group and pick fields display as a CSV list of text.
- Date fields display as a date.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]exportHeadings`

  specifies whether to export column names. By default, column names are exported.
- `--[no]openOutputFile`

  specifies whether to open the output file in Excel. By default, the file opens in Excel. If you specify `--noopenOutputFile` or are using a UNIX client, a message indicates the path and name of the temporary Excel file.
- `--outputFile=`*value*

  specifies the path and name of the Excel (`.xls`) output file.
- `--[no|confirm]overwriteOutputFile`

  specifies whether to confirm overwriting the existing Excel output file.
- `--[no|confirm]trimExcessColumns`

  specifies whether to confirm skipping specified columns that exceed Excel limitations. If you do not agree to skip columns that exceed Excel limitations and columns are detected, the command terminates, and no data is exported.
- `--fieldFilter=`*field*`=[`*value*`,`*value*`...]`

  specifies any field filters to apply to the query results. The first component of the value is the field name. The second component specifies the project(s)

that you want to filter the issues by. For example, `--fieldFilter="Project=/Project1"` filters for issues that have a value of `Project1` in the `Project` field. If you do not specify a value, Integrity filters for issues with a value of `Unspecified` in the `Project` field.

Note
Any project filtering applied to the query results in the GUI or Web interface does not apply in the CLI.

- `--[no]sortAscending`

  specifies whether to sort the specified field in ascending or descending order.
- `--sortField=`*`field`*`[:asc|desc]`

  specifies the primary field that issues sort by and the sort direction. This overrides the current default sort field and sort direction. You can sort fields in ascending (`asc`) or descending (`desc`) order, for example, `--sortField=State:asc`. To specify additional fields to sort items by, specify `--sortField=field:[asc|desc]` for each field. The sort order of fields is indicated by the order that the fields display in the command. For example, `--sortField=Type:asc --sortField=State:desc` sorts items by Type in ascending order first, and then by State in descending order. Specifying the sort direction is optional; however, if specified it overrides the sort direction specified in the `--[no]sortAscending` option. If the sort direction is not specified, the `--[no]sortAscending` option is used. If the `--[no]sortAscending` option is not specified, the default sort direction is used.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `--asOf=`*<date* |label:*<label*

  allows you to export items as of a historical date or label. Use in conjunction with the `--fields` option. For example, to export items containing the Description field as of a specific date, type:

  im exportissues --fields=Description --asOf="January 8, 2009 10:00:00 AM EST" 123 124 125

  Note the following:
- If you do not specify the `--asOf` option, the current field values for the specified items are exported.
- If you specify a `WEST` or `IST` time zone, the time does not display correctly. Instead, use the time zone GMT+/-*hours*:*minutes*.
- This option can only be specified when items are explicitly provided as a selection; it does not work when the `--query`, `--queryDefinition`, or `--queryDefinitionFile` option is used.
- This option is intended only for use with the CLI or API; it does not work when used with the `-g` option.
- `--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...`*

  specifies the issue fields, and their respective widths, to display. Your administrator defines the fields in an issue type. Fields can include `ID`, `Type`, `Assigned User`, `Assigned Group`, `Summary`, and others. Use commas to specify more than one field. The default fields that display in the CLI are independent of the default fields (columns) specified in the GUI and Web interface.
- `--[no]showTallRows`

  specifies whether to display variable height rows.
- `--structureFieldDisplayFormat=`*`value`*

  specifies the fields and style to display for tree nodes. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control. Uses the same values as `--fields`, but similar to a Java MessageFormat string (that is, it requires `{ }` to enclose each field). For example:

  ```
  im exportissues --query=jhoyt:Defects --structureFieldDisplayFormat="{ID},{Summary}"
  ```
- `--[no]showDecorators`

  specifies whether to display the `!` decorator to indicate a computed field in a versioned item contains an ambiguous computation. By default, the `--noshowDecorators` option is specified.
- `--query=[user:]query...`

  specifies the name of the user who the query belongs to and the query name, for example, `jhoyt:Cosmos Critical Defects`. For queries that have the same name, but different users, you must specify `[user:]query`. By default, the fields specified in the query are exported. To export specific fields in the query, specify the `--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...` option. To export specific items, specify the *item id...* option.

Note
Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` query created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the `jhoyt:CosmosDefects` query created by any user.

- `--queryDefinition=`*`query`*

  specifies a string to define the query constraints. For details of the query format, see the `im createquery` reference page.

Note
To reference the original query in the new query so that any changes to the original query are reflected in the new query, define *<subquery>*as *subquery[OriginalQuery]*.

- *item id...*

  specifies the ID of the item you want to export. Use a space-separated list to specify more than one item ID, for example, `240 241 242`. If you specify this option, you do not need to specify the `--query=[user:]query...` option, which exports all items returned by the specified query.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example,184-1.2.

See Also

- Commands: im createquery, im editquery, im issues, im queries
- Miscellaneous: options

im exporttranslations

exports all the available translations for the administrative objects such as Fields, Test Result Fields, Types, and States for the display name and description attributes.

Synopsis

```
im exporttranslations [--user=name] [--password=password] [--hostname=server] [--port=number ] [--adminType=value] [--
outputFile==value] [--sourceLocale=value] [--targetLocale=value] [--[no|confirm]overwriteOutputFile]
```

Description

`im exporttranslations` exports all the available translations for the administrative objects such as Fields, Test Result Fields, Types, and States for the display name and description attributes.

The following is an example of the command that exports the administrative object Field, for German as the target locale and for which the source locale is English. Specifies whether to confirm overwriting the file with the output file "`export.xml`", when the output file already exists at the same location.

```
im exporttranslations --adminType=field --sourceLocale=en --targetLocale=de --hostname=localhost --port=7001 --
outputFile="export.xml" --confirmoverwriteOutputFile
```

The following is an example of how the command with selection works. This command exports the translations for the administrative objects Field (Field1, Field2), State (State1), and Type (Type1).

```
im exporttranslations --sourceLocale=en --targetLocale=de --hostname=localhost --port=7001 --outputFile="export.xml" --
confirmoverwriteOutputFile Field:Field1 Field:Field2 State:State1 Type:Type1
```

Options

This command takes the universal options available to all the `im` commands, as well as some general options. See the options reference page for descriptions.

- `--adminType`

  specifies the field value for the name of the administrative object such as Field, Type, and State to be exported.
- `--sourceLocale`

  specifies the locale value in which translatable strings are available.
- `--targetLocale`

  specifies the locale value for which translations are required.
- `--outputFile`

  specifies the path and name of the XML file to export the translations to.
- `--[no|confirm]overwriteOutputFile`

  uses the following settings:
- `--[no|confirm]`

  specifies if to overwrite the existing file. The default is to confirm.
- `selection` is a set of multiple administrative object and its relevant instance, separated by a colon, such as Field:field1. When we have multiple sets, they are separated by space.

See Also

- Commands: im importtranslations
- Miscellaneous: options

im extractattachments

saves one or more attachments from an Integrity issue

Synopsis

```
im extractattachments [--issue=value] [--field=field] [--asOf=[<date>|label:<label>] [--outputFile=value] [--
[no|confirm]overwriteExisting] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F
value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] file...
```

Description

`im extractattachments`

saves one or more attachments from an Integrity issue for later viewing or printing.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--asOf=[<date>|label:<label>]`

  allows you to extract attachments as of a historical date or label. For example, to extract an attachment on an issue as of a specific date, type

  ```
  im extractattachments --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If a value is not provided the attachment displays as of the server's current time. This field is optional.
- `--issue=value`

  specifies the ID of the issue containing the attachment you want to save. This option is mandatory.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

Note
If you are extracting an attachment from an historical item, the attachment is extracted based on the historical date and time.

- `--outputFile=value`

  specifies the name of the file that the attachment is extracted to. If not specified, the name of the attachment is used as the output file.
- `--field=field`

  specify the attachment field you want to extract from, for example, *attachmenturl field=Attachments abc.txt*. If not specified, the default attachment field is used.
- `--[no|confirm]overwriteExisting=value`

  specifies whether to overwrite an existing file.
- *file...*

  specifies the name of the attachment to save, for example, `test_spec.htm`. Use spaces to specify more than one file. If this option is not specified, all attachments are extracted.

Note
If `--cwd` is not specified, the attachment is saved to the current working directory.

See Also

- Commands: im copyissue, im createissue, im editissue, im viewissue
- Miscellaneous: options

im extractwordtemplates

displays

Synopsis

`im extractwordtemplates [--outputFile=`*value*`] [--[no|confirm]overwriteExisting [--type=`*type*`] [--user=`*name*`] [--hostname=`*server*`] [--password=`*password*`] [--port=`*number*`] [(-?|--usage)] [(-F `*file*`|--selectionFile=`*file*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [-g|--gui] [--settingsUI=`*[gui|default]*`] [--quiet] [--status=`*[none|gui|default]*`] `*word template...*

Description

`im extractwordtemplates` extracts the Microsoft® Word template files from the Integrity type and outputs them to a file. For information on using Microsoft® Word templates, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--outputFile=`*value*

  Specifies to output a single specified template file to the current directory. If this option is not specified, all templates are outputted to the current directory.
- `--[no|confirm]overwriteExisting`

  Specifies if to overwrite existing template files in the current directory.
- `--type=`*type*

  Specifies the name of the type that contains the template files the command is to extract.
- *word template...*

  specifies the name of word template you want to extract.

See Also

- Commands: im createtype, im edittype, im viewtype, im deletetype
- Miscellaneous: options

im fields

displays a list of fields used to define custom fields

Synopsis

```
im fields [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value]
[--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)]
[(-Y|--yes)] [--[no]batch] [--[no]asAdmin] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] field .....
```

Description

`im fields` displays a list of defined fields. By default, all fields are selected to be displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field *n* can be any of the following:

  - `allowedTypes`

    displays the issue types that can be linked to the relationship field.
  - `cycleDetection`

    displays whether or not the system will prevent relationship loops from occurring in the relationship field.
  - `name`

    displays the name of the field. By default, only the name is shown.
  - `displayName`

    displays the name assigned as the display name of the field.
  - `pairedField`

    displays the field name of the paired relationship field. For forward relationship fields, the corresponding backward relationship field displays; for backward relationship fields the corresponding forward relationship field displays.
  - `rules`

    displays the trigger rules associated with the field.
  - `paramSubstitution`

    displays whether or not parameter references in the text field are replaced with parameter values when you view the item through a view or report that supports parameter substitution. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
  - `isMultiValued`

    displays whether or not the field can contain multiple values.
  - `description`

    displays a description of the field.
  - `type`

    displays the field data type.
  - `default`

    displays default value for the field (CLI only).
  - `defaultAttachmentField`

    displays default attachment field for the rich content field.
  - `defaultBrowseQuery`

    displays the default Admin query to use when adding a related item by browsing to the relationship field.
  - `displayAsLink`

    displays whether or not the selected value in the item backed pick list field displays as a hyperlink to the backing item in the GUI and the Web UI.
  - `min`

    displays the minimum value for the field for integer, float, and date fields (CLI only).
  - `max`

    displays the maximum value for the field for integer, float, and date fields (CLI only).
  - `displayAsProgress`

    displays whether or not the value of the integer field is displayed as progress bar in the GUI.
  - `displayLocation`

    displays the name of the issue tab where the relationship field is displayed (field or relationship).
  - `picks`

    displays the list of valid values for a pick list field, of the form *text:value:image* (CLI only).
  - `suggestions`

    displays the list of suggested values for a short text field (CLI only).
  - `textindex`

    displays whether or not queries against the field will be treated as word searches.
  - `trace`

    is valid for both relationship fields and sourcelink fields.

    For relationship fields, displays whether or not the field is a trace relationship. Trace relationships are defined via field pairs and are presented to the user in domain-specific language, for example, Test and Requirements. To learn more about trace relationships, see the *PTC Integrity User Guide*.

    For sourcelink fields, displays `true` if it is a trace field, and `false` if it is a non-trace (links-only) field.
  - `relevanceRule`

    displays the relevance rule for the field (CLI only).
  - `editabilityRule`

    displays the editability rule for the field (CLI only).
  - `id`

    displays the database ID of the field. This is for PTC - Integrity Support only.
  - `isForward`

    displays whether or not the field is a forward relationship field.
  - `linkFlags`

    displays relationship flags for the relationship field.
  - `maxLength`

    displays the maximum length of a long or short text field (CLI only).

- `displayRows`

  displays the number of rows used for a long text field or a relationship field.
- `displayStyle`

  displays the format used for the field: table or csv (comma separated values).
- `loggingText`

  displays the logging type of field. Valid for long text fields only (CLI only).
- `position`

  displays the position in the list of fields.
- `computation`

  displays the expression for the computed field.
- `staticComputation`

  displays whether the computed field is a static or dynamic computation.
- `storeToHistoryFrequency`

  displays the frequency at which the computed field is calculated and stored to issue history.
- `lastcompute`

  displays the date and time that the computed field was last calculated.
- `references`

  displays a list of object references.
- `associatedField`

  displays the field associated with a field value attribute.
- `backedBy`

  displays the issue backed picklist that backs the field value attribute.
- `backingStates`

  displays the active states that back the issue backed picklist.
- `backingTextField`

  displays the short text field containing text for an issue backed picklist.
- `backingTextFormat`

  displays the fields containing values that are linked together to form the picklist values for an issue backed picklist.
- `backingType`

  displays the type that backs an issue backed picklist.
- `backingFilter`

  displays the filter applied to values in an issue backed picklist.
- `correlation`

  displays the field contained in the type containing the query backed relationship field and a field in the issues returned by the query.
- `defaultColumns`

  displays the default columns for the relationship field.
- `displayPattern`

  displays the display pattern for a numeric field.
- `isSystemManagedFields`

  displays whether the field is system-managed. Possible values are `true` and `false`.
- `phases`

  displays the phases for a phase field.
- `query`

  displays the query for the query backed relationship field.
- `ranges`

  displays the ranges for the range field.
- `richContent`

  displays whether the field supports rich content and the specified screen and printer CSS files.
- `showTallRows`

  displays if the field displays variable height rows. Possible values are `true` and `false`.
- `sortIBPLDescending`

  displays the sort order of the field on the backing item type in the IBPL. This is only displayed when a sort field and a direction have been set on an IBPL field.
- `sortIBPLField`

  displays the field on the backing item to sort by. This is only displayed when a sort field and a direction have been set on an IBPL field.
- `relationshipBrowseStyle`

  displays the relationship browse style for the relationship specified. The relationship browse style can be either set to `Queries` or `Finder` through the `im createfield` command.
- `incomingTraceProvider`

  displays the name of the incoming trace provider that is used to retrieve external references. If you are using ThingWorx, this is the name of the ThingWorx thing that returns IER values. This option is only valid for a field with a `type` of `ier`.

- `--[no]asAdmin`

  allows non-admin access to fields. This option is used specifically for third party integrations using the Integrity API. The default setting is `--asAdmin` which requires ViewAdmin permissions. Specifying `--noasAdmin` prevents visibility to fields that give information about other users. This option overrides the fields specified using the `--fields` option. Accessible and non-accessible fields are defined by the system for this command.

  To learn more about the API, see the *PTC Integrity Integrations Builder Guide*.
- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between fields displayed in the CLI.
- *`field...`*

  specifies the name of the fields to display.

See Also

- Commands: im createfield, im editfield, im viewfield
- Miscellaneous: options

im getdbfile

retrieves a workflow and document configuration file from the database

Synopsis

```
im getdbfile [--encoding=value] [--output=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...
```

Description

`im getdbfile` retrieves a workflow and document configuration file from the database, such as a ViewSet or item presentation template (IPT). Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing. Once you are finished editing these files, you can store them in the database using the `im putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=`*`value`*

  specifies the code set to save the file in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--output=`*`value`*

  specifies the name of the file to store the output to on the local file system.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `im diag --diag=listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\`*`templatename`*`.xml`, for example, `im getdbfile --output=c:/defect.xml data/im/issue/templates/defect.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

im groups

displays the list of groups

Synopsis

`im groups [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] group...`

Description

`im groups` displays a list of groups for workflows and documents. By default, all groups are selected to be displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the group. By default, only the name is shown.
  - `id`

    displays the database ID of the group. This is for PTC - Integrity Support only.
  - `isActive`

    displays whether or not the group is active.
  - `queryTimeout`

    displays the query timeout value for the group.
  - `description`

    displays a description of the group.
  - `image`

    displays whether or not there is an image, and if the image is custom or default.
  - `email`

    displays the email address of the group.
  - `notificationRule`

    displays the email notification rule associated with the group (CLI only).
  - `isInRealm`

    displays whether or not the group exists in the authentication realm (CLI only).
  - `references`

    displays a list of object references.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields
- *group...*

  specifies the names of the groups to view.

See Also

- Commands: im creategroup, im editgroup, im viewgroup, im deletegroup
- Miscellaneous: options

im gui

starts the Integrity Client graphical user interface

Synopsis

`im gui [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`im gui` starts the Integrity Client graphical user interface (GUI).

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--height=`*value*

  specifies the height of the GUI window, in pixels; `value` must be a whole number.
- `--width=`*value*

  specifies the width of the GUI window, in pixels; `value` must be a whole number.
- `-x` *value*

  specifies the location of the GUI window on the x axis, in pixels; `value` must be a whole number.
- `-y`*value*specifies the location of the GUI window on the y axis, in pixels; `value` must be a whole number.

See Also

- Miscellaneous: options, preferences

im importcontent

creates content from imported information

Synopsis

```
im importcontent[--parentID=value] [--insertLocation[=number|first|last|before:name|after:name] [--segmentType=type] [--field=value]
[--richContentField=value] [--subInsertMode=[reference|include]] [--addRelationships=value] [--addSourceTrace=value] [--
addSourceLink=value] [--addAttachment=value] [--type=type] [--quiet] [--user=name] [--hostname=server] [--password=password] [--
port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] item id...
```

Description

`im importcontent` creates content from imported information. You can import content into at a specified location in the parent's segment structure. For example:

```
im importcontent --parentID=123 --field='Category=System Requirement' --field='Text=New requirement' -----insertLocation=after:34 15
```

creates content, inserts it after ID 34 in the relationship list for parent segment 123, and adds existing item 15 as a child for the new content.

Note

This command is normally used with an API.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--field=`*value*

  specifies a field and its value, where *value* is of the form "*fieldName=fieldValue*", for example, `--field=Severity=Critical`. If the field is multi-valued, *value* is of the form "*fieldName=fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add.

  To specify a workflow and document project, project names must be preceded by a (/), for example, `--field=Project=/testProject`.

  To specify an configuration management project, use the following syntax: `--field=`*fieldname*`,server=`*server*`,project=`*projectname*, [`devpath=`*devpath*`|revision=`*checkpointrevision*].

- `--addRelationships=`*value*

  adds related items, where *value* is of the form [`FieldName`]:*ID*[*relationshipFlags*][,...]. If no field name is specified, the *Forward Relationships* field is used.

  Note

  Adding a related item is only permitted if your administrator has allowed relationships for the item type.

- `--addSourceTrace=`*value*

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName] [,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addSourceLink=`*value*

  adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber,[file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName] [,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--segmentType=`*type*

  specifies the segment type under which you want to import content if the `--parentID` is not specified. Required if you create content unrelated to a `--parentID`. Segment types are defined by your administrator.

- `--type=`*type*

  specifies the type of the segment. Your administrator defines segment types. This option is required if more than one segment type is allowed and the type field is not specified.

- `--richContentField=`*value*

  specifies a rich content field and its value, where *value* is of the form "*richcontentfieldname=fieldValue*".

- To specify a link to an item by ID: `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

  For example:

  ```
  <a href="mks:///item?itemid=4547">Part Requirement</a>
  ```

  To specify a link to an item by ID and label: `<a href="mks:///item?itemid=`*ID*`&label=LABEL">`*DisplayText*`</a>`

  For example:

  ```
  <a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>
  ```

  To specify a link to an item by ID and revision: `<a href="mks:///item?itemid=`*ID*`&revision=REVISION">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>`

  To specify a link to an item by ID and date: `<a href="mks:///item?itemid=`*ID*`&asof=DATETIME">`*DisplayText*`</a>` where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

  For example:

  ```
  <a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>
  ```

  Note

  In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "`b>Feature Overview/b>`". In the Windows command line interface (`cmd.exe`), the ^ escape character must precede the and > characters in HTML tags, for example, `^b^>Feature Overview^/b^>`. You may need to escape nested " characters, for example,

  ```
  --richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"
  ```

.

- `--addAttachment=`*value*

  adds attachments, where *value* is of the form "*fieldName*`,path=`*pathToFile*`[,name=`*nameOfAttachment*`][,summary=`*shortDescription*`]" .

> **Note**
> The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

For example,

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for item 123""
```

adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. A segment or node cannot have more than one attachment with the same name.

> **Note**
> Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

- `--subInsertMode=`*[reference|include]*

  specifies how subdocuments are inserted when content is imported into a document. The available options are `Reference` or `Include`. `Reference` inserts a subdocument but document contents are not visible in the tree structure; a subdocument must be opened in order to manage its contents. `Include` exposes all contents as if they were a sequential part of the parent document. You can only reference or include all content at the same time. The default is `Reference`. The options are as follows:

  reference imports content as reference only.

  include imports content as included subsections.

  For example:

  ```
  im importcontent --type=Node --field='Category=Heading' --subInsertMode=reference 57
  ```

  creates a new node 60 and inserts a reference to document 57 beneath 60.
- `--parentID=`*value*

  the ID of the parent segment or node that will contain the reference to the node being imported. This option is required.
- `--insertLocation[=`*number*`|first|last|before:`*name*`|after:`*name*`]`

  determines where the content should go in the parent segment's structural relationship list. This option is required if you specify a `--parentID`. You cannot use this option without specifying a `--parentID`. The options are as follows:

  `first` inserts the content at the beginning of the list

  `last` inserts the content at the end of the list

  `before:`*name* inserts the content before the specified ID

  `after:`*name* inserts the content after the specified ID

  [0,...] inserts the content at the specified location. If a negative is specified, the content is inserted at the beginning of the list. If the number specified is too large, the content is inserted at the end of the list.
- *item id...*

  the ID(s) of content you want to import into an existing segment or node. Use a space separated list to specify more than one item ID, for example, `240 241 242`.

See Also

- Commands: im createcontent, im importsegment
- Miscellaneous: diagnostics, options, preferences

im importgroup

imports groups into Integrity

Synopsis

```
im importgroup [--[no]allowNonRealm] [--quiet] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--n)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...
```

Description

`im importgroup` imports one or more groups into Integrity. For example:

```
im importgroup --allowNonRealm Services Development QA
```

imports the *Services*, *Development*, and *QA* groups that are not part of the security realm.

Note

Integrity automatically imports the *everyone* group from your security realm.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]allowNonRealm`

  specifies whether to import groups that are not part of the security realm. `--noallowNonRealm` is specified by default.

  Note

  Importing non-realm groups creates new group principals.

- *string...*

  specifies the names of one or more groups to import.

See Also

- Commands: im importuser, im importissue, im creategroup,im createuser
- Miscellaneous: options

im importissue

imports an issue into Integrity

Synopsis

```
im importissue [--createdDate=value] [--createdUser=value] [--type=type] [--addSourceTrace=value] [--addSourceLink=value] [--addAttachment=value] [--addRelationships=value] --customFieldDefinition=value --customFieldValue=value [--field=value] [--richContentField=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [-g|--gui] [(-N|--no)] [--quiet] [(-Y|--yes)] [--[no]batch] [--settingsUI=[gui|default]] [--cwd=directory] [--status=[none|gui|default]] [--forceConfirm=[yes|no]]
```

Description

`im importissue` imports an issue into the Integrity database, creating a new issue to represent it. For example:

```
im importissue --createdDate="12/13/2008 7:00:00 AM" --type=Defect
```

imports the *Defect* item created on *12/13/2008* at *7:00:00 AM*, and creates a new item to represent it.

Note

The `Admin` permission is required to use the `im importissue` command.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--createdDate=`*value*

  specifies the date that the issue was originally created on. This option supports the following timestamp formats: *MM/dd/yy h:mm:ss a z*, *MM/dd/yy h:mm:ss a*, and *MM/dd/yy hh:mm a* (where *a* is AM or PM, and *z* is a timezone name, for example, for example, CEST, CET, EST, PST, or GMT*+/-hours:minutes*). Dates that occur in the future cannot be specified. If this option is not specified, the date and time that the command is run is recorded as the created date.
- `--createdUser=`*value*

  specifies the user name of the user who originally created the issue If this option is not specified, the user running the command is recorded as the created user.
- `--type=`*type*

  specifies the type of issue to create in the database.
- `--addSourceTrace=`*value*

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName] [,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*"

  Note

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addSourceLink=`*value*

  adds a link to a source file, where `value` is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision] [,start=startLineNumber][,end=endLineNumber]*"

  Note

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.

- `--addAttachment=`*value*

  adds attachments, where *value* is of the form

  "*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]* ".

  Note

  Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

- `--addRelationships=`*value*

  specifies the related issues for the issue, where *value* is of the form `[fieldname]:id[linkflags][,...]`.
- `--field=`*value*

  specifies a value for the field, of the form *fieldName=fieldValue*.

  Note

  It is possible to set the state value to any state in the workflow, including an the end state if there is one.

- `--richContentField=`*value*

  specifies a rich content field and its value, where *value* is of the form "*richcontentfieldname=fieldValue*".

  To specify a link to an item by ID:

  `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547">Part Requirement</a>`

  To specify a link to an item by ID and label:

  `<a href="mks:///item?itemid=`*ID*`&label=LABEL">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>`

  To specify a link to an item by ID and revision:

  `<a href="mks:///item?itemid=`*ID*`&revision=REVISION">`*DisplayText*`</a>`

  For example: `<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>`

  To specify a link to an item by ID and date:

`<a href="mks:///item?itemid=ID&asof=DATETIME">DisplayText</a>`
,where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

For example: `<a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>`

-

Note

In the korn shell command line interface, HTML tags must be surrounded by quotes, for example,"`<b>Feature Overview</b>`". In the Windows command line interface (cmd.exe), the ^ escape character must precede the < and > characters in HTML tags, for example, `^<b^>Feature Overview^</b^>` . You may need to escape nested " characters, for example, `--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"`.

See Also

- Commands: im deleteissue
- Miscellaneous: options

im importsegment

creates one or more new segments from imported requirement information

Synopsis

```
im importsegment[--type=type] [--parentID=value] [--addAttachment=value] [--addRelationships=value] [--addSourceTrace=value] [--addSourceLink=value] [--customFieldDefinition=value] [--customFieldValue=value] [--field=value] [--richContentField=value] [--insertMode=[reference|include]] [--subInsertMode=[reference|include]] [--insertLocation[=number|first|last|before:name|after:name] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [--parentID=value] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [segment id...]
```

Description

`im importsegment` creates one or more new segments from imported information. A segment can be a segment root or a subsegment. If importing as a subsegment, you can optionally add the segment at a specified location in the parent segment's structural relationship list. For example:

```
im importsegment --parentID=23 --insertLocation=after:34 --field='Project=Project2' --type='Subsegment' --field='Category=SubFolder'
```

creates a segment and inserts it after ID 34 in the structural relationship list for parent segment 23

```
im importsegment --field='Project=/Project2' --field='Category=Document' 241 245 243
```

creates a segment using the default type and adds children with IDs 241, 245 and 243 beneath it.

Note

This command is normally used with an API.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--subInsertMode=[reference:include]`

  specifies how subdocuments are inserted when segments are imported into a document.

  `Reference`inserts a subdocument but the document contents are not visible in the tree structure; the subdocument must be opened in order to manage its contents.

  `Include`exposes the entire contents of the subdocument as if they were a sequential part of the parent document. You can only reference or include all subdocuments at the same time. The default is `Reference`.

  For example:

  ```
  im importsegment --type=Segment --field='Project=/Document' --field='Shared Category=Document' --subInsertMode=include --parentId=55 56
  ```

  results in the newly created document 57 being imported as a reference underneath 551, and 562 being inserted as an inclusion beneath the newly created document.
- `--field=value`

  specifies a field and its value, where *value* is of the form "*fieldName=fieldValue*", for example, `--field=Severity=Critical`. If the field is multi-valued, *value* is of the form "*fieldName=fieldValue*,...".

  To specify more than one field, specify this option for each field you want to add.

  To specify a workflow and document project, project names must be preceded by a (/), for example,`--field=Project=/testProject`. An Integrity project is required.

  To specify a configuration management project, use the following syntax: `--field=`*fieldname*`,server=`*server*`,project=`*projectname*, [`devpath=`*devpath*`|revision=`*checkpointrevision*]
-
- `--addRelationships=value`

  adds related items to the segment, where *value* is of the form [*FieldName*]:*ID*[*relationshipFlags*][,...]. If no field name is specified, the `Forward Relationships` field is used.
-

  Note

  Adding a related item is only permitted if your administrator has allowed relationships for the segment's item type.

-
- `--addSourceTrace=value`

  adds a trace to a source file, where *value* is of the form "*field=scmSourceFieldName,server=scmServerName,port=scmPortNumber,file=fileName,project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source trace more than once. In order to be considered a new source trace, the trace must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
-
- `--addSourceLink=value`

  adds a link to a source file, where *value* is of the form "*field=scmSourceFieldName,revision=memberRevisionID, server=scmServerName,port=scmPortNumber, [file=fileName|subproject=subprojectName|isProject],project=scmProjectName[,devpath=devpathName][,projectRevision=revision][,start=startLineNumber][,end=endLineNumber]*".

  Note the following:

  - If you specify a configuration path for *project=scmProjectName* then *devpath=devpathName* and *projectRevision=revision* cannot be specified.
  - You cannot add the same source link more than once. In order to be considered a new source link, the link must have a unique combination of the following attributes: member name, server name, server port, project, project revision, development path, revision, start line number, and end line number.
- `--richContentField=value`

  specifies a rich content field and its value for the segment, where *value* is of the form "*richcontentfieldname=fieldValue*".

  To specify a link to an item by ID: `<a href="mks:///item?itemid=`*ID*`">`*DisplayText*`</a>`

  For example:

  ```
  <a href="mks:///item?itemid=4547">Part Requirement</a>
  ```

  To specify a link to an item by ID and label: `<a href="mks:///item?itemid=`*ID*`&label=LABEL">`*DisplayText*`</a>`

  For example:

  ```
  <a href="mks:///item?itemid=4547&label=EDF-343">Part Requirement, Label EDF-343</a>
  ```

To specify a link to an item by ID and revision: `<a href="mks:///item?itemid=`*ID*`&revision=REVISION">`*DisplayText*`</a>`

For example:

```
<a href="mks:///item?itemid=4547&revision=1.2">Part Requirement, Revision 1.2</a>
```

To specify a link to an item by ID and date: `<a href="mks:///item?itemid=`*ID*`&asof=DATETIME">`*DisplayText*`</a>` where date is in the format for your locale; such as US date formats "mm/dd/yy", "mm/dd/yyyy", or "mmm d, yyyy" with the time "h:mm:ss a".

For example:

```
<a href="mks:///item?itemid=4547&asof=03/25/2012 12:04:00 AM">Part Requirement, March 25, 2012 12:04am</a>
```

-

Note
In the korn shell command line interface, HTML tags must be surrounded by quotes, for example, "`b>Feature Overview/b>`". In the Windows command line interface (cmd.exe), the ^ escape character must precede the and > characters in HTML tags, for example, ^`b^>Feature Overview^/b^>`. You may need to escape nested " characters, for example,

```
--richContentField=Description="<a href=\"mks:///item?itemid=4547\">Part Requirement</a>"
```

.

-
- `--addAttachment=`*value*
- adds attachments to the segment, where *value* is of the form "*fieldName*`,path=`*pathToFile*`[,name=`*nameOfAttachment*`][,summary=`*shortDescription*`]` " .

Note
The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

For example,

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for item 123""
```

adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

Note
Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

`--type=`*type*

specifies the item type of the segment. Your administrator defines item types. This option is required if more than one segment type is allowed in the solution and the type field is not specified.

`--parentID=`*value*

the ID of the parent segment that will contain the reference to the segment being imported. The parent item must be a segment or a node. This option is required.

`--insertLocation[=`*number*`|first|last|before:`*name*`|after:`*name*`]`

determines where the segment should go in the parent segment's structural relationship list. You must specify a `--parentID`. The options are as follows:

first

inserts the segment at the beginning of the list

last

inserts the segment at the end of the list

before

:*name* inserts the segment before the specified ID

after

:*name* inserts the segment after the specified ID

[0,...] inserts the segment at the specified location. If a negative is specified, the segment is inserted at the beginning of the list. If the number specified is too large, the segment is inserted at the end of the list.

`--insertMode=`*[reference|include]*

indicates the way content related to the segment should be be imported. This option is required if you specify a `--parentID`. The options are as follows:

reference references related items but does not include them.

include includes related items as children of the parent item.

segment id...

the ID(s) of previously created segment(s) that you want to import the segment(s) into. Use a space separated list to specify more than one segment, for example, `240 241 242`.

See Also

- Commands: im importcontent, im insertsegment im createsegment
- Miscellaneous: diagnostics, options, preferences

im importtranslations

imports all the available translations for the administrative objects such as Fields, Test Result Fields, Types, and States for the display name and description attributes.

Synopsis

`im importtranslations [--user=name] [--password=password] [--hostname=server] [--port=number ] [--inputFile==value]`

Description

`im importtranslations` imports all the available translations for administrative objects such as Fields, Test Result Fields, Types, and States for the display name and description attributes.

The following is an example of the command that imports the translations for which the input file is located at `D:\Import\Importtranslations.xml`

```
im importtranslations --hostname=localhost --port=7001 --inputFile=D:\Import\Importtranslations.xml
```

For more information on the tags and subtags for importtranslations, see the *Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--inputFile`

  specifies the name of the XML file which contains the translations to be imported.

See Also

- Commands: im exporttranslations
- Miscellaneous: options

im importuser

imports one or more users into Integrity

Synopsis

`im importuser [--[no]allowNonRealm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--n)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`string...`*

Description

`im importuser` imports one or more users into the Integrity database. This command is helpful for adding users who have not yet auto-registered by logging in, or for adding a number of users at one time. Additionally, you can import non-realm users by specifying a large user list text file. For example:

```
im importuser jriley
```

imports the user *jriley*.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]allowNonRealm`

  specifies whether to import users that are not part of the security realm. `--noallowNonRealm` is specified by default.

  Note

  Importing non-realm users creates new user principals.

- *`string...`*

  specifies the names of one or more users to import.

See Also

- Commands: im importgroup, im importissue, im creategroup, im createuser
- Miscellaneous: options

im incrementrevision

increments the revision of an item

Synopsis

`im incrementrevision [--force] [--major] [--minor] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`issue id...`*

Description

`im incrementrevision` increments the revision number of an item.

Item revisions mark an unambiguous point in the history of an item. When viewing an item in Integrity, the revision provides an identifier into its history. Furthermore, if you export the same item to a partner using a report or through ReqIF, and they call you to discuss it, you are assured that any discussion of Item 123, revision 1.2, for example, is done with the exact same item definition and understanding.

This is also true for everyone with direct access to Integrity. When listing items, you have clear visibility to the item revision. As with the partner scenario, any internal communication through conversations, reports, or e-mail, can include a specific revision to ensure all parties are using the same exact definition.

The revision identifier is numbered as a two-part decimal in the format *major.minor*, where the number to the left of the decimal point is the *major revision*, and the number to the right of the decimal point is the *minor revision*. Your Integrity administrator defines the revision pattern for revision numbers by specifying values in workflow and document properties. For example, the default revision number sequence is 0.1, 0.2, 0.3, ... 1.0, 1.1, 1.2. For information on your revision pattern for revision numbers, contact your administrator.

For example, consider the following item revisions:

- Item 100 is at revision 1.5. There are no changes to the item since the last revision.
- Item 200 is at revision 2.8. There are changes to the item since revision 2.8.

Specifying:

```
im incrementrevision --minor 100
```

does not increment the minor revision of the item. By default, the revision is incremented only if there have been changes since the last revision.

Specifying:

```
im incrementrevision --minor --force 100
```

increments the minor revision of item 100 to revision 1.6. The `--force` option overrides the default behavior and increments the revision, even though there have been no changes since the last revision.

Specifying:

```
im incrementrevision --major 200
```

increments the major revision of item 200 to revision 3.1.

Note the following:

- If item labels are enabled for the item type, incrementing the revision number automatically adds a revision label (using the revision number) to the item. For example, if you increment the revision to 1.2 and item labels are enabled, Integrity automatically creates a 1.2 revision label in the `Labels` tab. Revision labels display in ascending order by default. The label :head is a system provided label and always appears first in the list. The date and time of :head are updated by revision operations; thus, the date and time are always set to that of the latest item revision.
- To control and automate item revisioning in your environment, your administrator may implement event triggers that automatically increment the revision of an item with workflow changes, such as the "Ready for Review" state.
- When copying a revisioned item, the revision on the existing item is not copied to the new item.
- There is no way to set an arbitrary revision. When an item is revisioned, the next available number defined by the revision schema sets the next revision.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--force`

  forces the increment to occur even when the item is unchanged. By default, the revision is not incremented when the item is unchanged.

Note

When incrementing a revision, an item with no previous revision number is considered to contain changes. Therefore, if you do not specify the `--force` option, an initial revision number is created.

- `--major`

  increments the major revision number. For example, if the current revision is 1.5, incrementing the major revision updates the revision to 2.1.
- `--minor`

  increments the minor revision number. For example, if the current revision is 1.5, incrementing the minor revision updates the revision to 1.6. By default, the minor revision number is incremented.
- *issue id...*

  the ID of the item you are incrementing the revision for. Use spaces to specify more than one item.

See Also

- Miscellaneous: ACL, diagnostics, options, preferences

im insertsegment

inserts an existing segment into a parent segment

Synopsis

`im insertsegment[--insertMode=`*`[reference:include]`*`] [--insertLocation[=`*`number`*`|first|last|before:`*`name`*`|after:`*`name`*`] [--parentID=`*`value`*`] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [--quiet] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] segment id...`

Description

`im insertsegment` inserts an existing segment into a parent segment. You can insert the segment into a specified location in the parent segment's structural relationship list. For example:

```
im insertsegment --parentID=23 --insertLocation=after:34 15
```

inserts segment 15 into parent segment 23, placing it after ID 34 in the the structural relationship list.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--parentID=`*`value`*

  the ID of the parent segment or node that will contain the reference to the segment being inserted. This option is required.

- `--insertLocation[=`*`number`*`|first|last|before:`*`name`*`|after:`*`name`*`]`

  determines where the segment should go in the parent segment's structural relationship list. You must specify a `--parentID`. The options are as follows:

  `first` inserts the segment at the beginning of the list

  `last inserts the segment at the end of the list`

  `before`:*name* inserts the segment before the specified ID

  `after`:*name* inserts the segment after the specified ID

  [0,...] inserts the segment at the specified location. If a negative is specified, the segment is inserted at the beginning of the list. If the number specified is too large, the segment is inserted at the end of the list.

- `--insertMode=`*`[reference:include]`*

  indicates how related items are to be treated for the segment when it is inserted. This option is required if you specify a `--parentID`. You cannot use this option without specifying a `--parentID`. The options are as follows:

  reference references related items but does not include them

  include includes related items as children of the parent item

- *segment id...*

  the ID(s) of the segment(s) being inserted. Use a space separated list to specify more than one segment, for example, `240 241 242`.

See Also

- Commands: im createsegment, im viewsegment, im branchsegment
- Miscellaneous: diagnostics, options, preferences

im installsolution

installs a solution template

Synopsis

```
im installsolution [--[no]confirmPartial] [--conflictPrefix=value] [--[no]createData] [--groupMap=value] [--prefix=value] [--
siProjectDir=value] [--solutionFile=value] [--userMap=value] [--user=name] [--hostname=server] [--password=password] [--port=number]
[(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--quiet] [--cwd=directory] [--
forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im installsolution` runs the installer that sets up a solution. For example:

```
im installsolution --createAdmins --prefix=RM_ --solutionFile=C:/MKS/rm2007.imt --
groupMap=Administrators=everyone,Analysts=everyone,Integrations=everyone,Manager=everyone,Testers=everyone --
userMap=administrator=guest,analyst=guest,developer=guest,integration=guest,manager=guest,tester=guest --siProjectDir=C:/RM_repo7
```

⚠ Caution

To successfully install a solution, you must specify the hidden option `--createAdmins`. If you do not specify this option, the installation fails and does not display an error message.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirmPartial`

  specifies whether to be prompted to confirm the partial installation of the solution template when no prefix is specified.
- `--conflictPrefix=`*value*

  specifies a conflict prefix to prepend to projects, states, fields, types, triggers, and other object names that may conflict with existing object names. This is most likely to occur if you are installing the solution without a prefix. Specifying a conflict prefix is optional.
- `--[no]createData`

  specifies whether to create sample data when installing the solution. Sample data populates certain pre-defined configuration management projects. The sample data can be helpful if you intend to use the installation for evaluation and training; however, if you plan to use this installation for production work, PTC recommends that you clear the option and install without the sample data. Sample data is not easily deleted after installation.
- `--groupMap=`*value*

  specifies the mapping of the solution to local groups on the system in the format `solutionGroup=`*`localGroup[,...]`*.
- `--prefix=`*value*

  specifies the prefix to apply to projects, states, fields, types, triggers, and other objects set up by the installer. To distinguish the new objects from your existing ones, a prefix is applied to the newly created objects. For example, if you accept the default prefix of `RM_`, then states such as `Reject` and `Defer` are named instead `RM_Reject` and `RM_Defer`. If your current database is not already configured, then you may also choose to set a blank prefix; however, if you set a blank prefix, subsequent templates will require a prefix upon installation.
- `--siProjectDir=`*value*

  specifies the name of the target directory where the installer places sample data on the configuration management system.
- `--solutionFile=`*value*

  specifies the name of the .imt file containing the solution. For example, for MKS Requirements the file name is `rm_2007.imt`.
- `--userMap=`*value*

  specifies the mapping of the solution to the local user on the system in the form `solutionUser=`*`localUser[,...]`*.

See Also

- Miscellaneous: options

im issues

presents issues found in the query results for the specified query

Synopsis

```
im issues[--[no]applyDisplayPattern] [--asOf=[<date |label:<label ] [--
fields=field[:width[:rich|plain|idlist]],field[:width[:rich|plain|idlist]],...] [--fieldsDelim=value] [--[no]sortAscending] [--
sortField=field] [--fieldFilter=field=[value,value...]] [--[no]inlineEditMode] [--[no]ApplyDisplayPattern] [--[no]showTallRows] [--
[no]showDecorators] [--structureFieldIconDisplayField=field] [--structureFieldDisplayFormat=value] [--focusIssueID=value] [--
expandLevel=value] [--[no]substituteParams] [--[no]showXHTML] [--queryDefinition=query] [--hostname=value] [--port=value] [--
password=value] [--user=value] [--query=[user:]query] [(-?|--usage)][(-g|--gui)] [--height=value] [--width=value] [-x value] [-y
value] [(-F value|--selectionFile=value)] [--quiet][--settingsUI=[gui|default]][--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)]
[--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--hostname=value] issue id [--[no]showEditPanel] [--[no]confirmSave] [--
[no]showIssueView]
```

Description

`im issues` presents, in list form, all the issues for a specified query. You can also choose the columns to use in displaying the issues view information.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]applyDisplayPattern`

  specifies whether to apply a display pattern to numeric fields. Display patterns are configured by your administrator and allow you to quantify integer and floating point field values, for example, as currency or percentages. `--applyDisplayPattern` is enabled by default.

  Note
  If you use scripts, PTC recommends using the `--noapplyDisplayPattern` option to avoid being impacted by administrative changes to display patterns.

- `--asOf=<date |label:<label>`

  allows you to view rich content for issues as of a historical date or label. Use in conjunction with the `--fields=::rich` option. For example, to view the rich content of a Description field as of a specific date, type

  ```
  im issues --fields=Description::rich --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If a value is not provided the rich content information displays as of the server's current time.

  Note the following:

  - If you specify a `WEST` or `IST` time zone, the time is not displayed correctly. Instead, use the time zone GMT+/-*hours*:*minutes*.
  - This option can only be specified when items are explicitly provided as a selection; it will not work when the `--query` option is used.
  - This option is intended only for use with the CLI or API and does not work when used with the `-g` option.
- `--fields=field[:width[:rich|plain|idlist]],field[:width[:rich|plain|idlist]],...`

  specifies the issue fields, and their respective widths, to be displayed. Your administrator defines the fields in an issue type. Fields can include `ID`, `Type`, `Assigned User`, `Assigned Group`, `Summary`, and others. Use commas to specify more than one field. The default fields displayed in the CLI are independent of the default fields (columns) specified in the GUI and Web interface.

  For rich content fields, you can specify display patterns to display field values as rich content (`:rich`) or plain text (`:plain`). The rich content display pattern displays the underlying HTML elements and attributes in the rich content field. For example, `--fields=Description::rich` displays the Description field value with HTML elements and attributes. By default, rich content fields display plain text.

  For relationship fields, you can specify the display pattern as a list of IDs (`:idlist`). When you use the Integrity-API the default presentation for a relationship field is a list of partial-representation items which requires the API to retrieve the related items from the Integrity Server. Use the display pattern with the Integrity-API to present the relationship field as a list of generic Integrity IDs to avoid the need for the additional item retrieval. The display pattern has no visible effect in the CLI.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between the fields in the display.
- `--focusIssueID=value`

  specifies the item ID to focus on in the table of returned issues.
- `--[no]sortAscending`

  specifies whether to sort the specified field in ascending or descending order.
- `--sortField=field`

  specifies the field to sort issues by, for example, `ID`.

  Note the following:

  - By default, issues are sorted by ID. Additional default fields that display are: type, state, and summary.
  - If a sorting field is specified with `--sortField`, that field is used.
  - If a list of fields are specified with `--fields`, the list of fields are used.
- `--fieldFilter=field=[value,value...]`

  specifies any field filters to be applied to the query results. The first component of the value is the field name. Currently, only project field filters are supported. The second component specifies the project(s) that you want to filter the issues by. For example, `--fieldFilter="Project=/Project1"` filters for issues that have a value of `Project1` in the `Project` field. If you do not specify a value, Integrity filters for issues with a value of `Unspecified` in the `Project` field.

  Note
  Any project filtering applied to the query results in the GUI or Web interface does not apply in the CLI.

- `--[no]inlineEditMode`

  specifies whether to allow inline editing. This option must be specified with the `-g`/`--gui` option.
- `--[no]showTallRows`

  specifies whether to display variable height rows.
- `--[no]showDecorators`

  specifies whether to display the `!` decorator to indicate a computed field in a versioned item contains an ambiguous computation. By default, the `--noshowDecorators` option is specified.
- `--[no]showXHTML`

  specifies if to display rich text field data in XHMTL format. Use with the `--fields` option and the `:rich` display pattern. The `--[no]showXHTML` option is intended for use in triggers, scripts, or custom integrations. The output converts all URLs to fully qualified server URLs. Special characters are included in the CLI output, but not the API output (API output is entity protected, for example < is converted to <).
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `--structureFieldIconDisplayField=field`

specifies the field from which the icon is taken. This is the icon that will be displayed for each node in the Outline pane.

- `--structureFieldDisplayFormat=`*value*

  specifies the fields and style that should be displayed for the tree nodes.

  defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  Uses the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). For example:

  ```
  im viewsegment --structureFieldDisplayFormat="{ID},{Summary}"
  ```

- `--query=`*[user:]query*

  specifies the query to use to populate the view. If not specified, Integrity uses the most recently run query.

- `--focusIssueID=`*value*

  specifies the item ID to highlight in the Structure column.

- `--expandLevel=`*value*

  specifies to expand the nodes in the Structure column to a specified level, for example, 1, 2, 3. The default is one.

- `--queryDefinition=`*query*

  specifies a string to define the query constraints. For details of the query format, see the `im createquery` reference page.

Note

To reference the original query in the new query so that any changes to the original query are reflected in the new query, define *<subquery>* as *subquery[OriginalQuery]*.

- *issue id*

  specifies the issue identification number of the issue you want to display in the view. Overrides the `--query` option.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

- `--[no]showEditPanel`

  specifies whether to show the embedded item edit pannel. This option can be used in conjunction with the `--[no]showIssueView` option. If conflicts arise, the`--[no]showEditPanel` option always wins.

- `--[no]showIssueView`

  specifies whether to show the embedded item view. This option can be used in conjunction with the `--[no]showEditPanel` option. If conflicts arise, the `--[no]showEditPanel` option always wins.

- `--[no]confirmSave`

  specifies whether to prompt you to confirm changes are to be saved. This turns on and off whether you a confirmation window oepns when you save in this view.

See Also

- Commands: im viewissue
- Miscellaneous: options

im loadrc

loads the Integrity Client preferences file

Synopsis

```
im loadrc [--[no]merge] [--rc=value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
[(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im loadrc` loads the user's `IntegrityClient.rc` file, which contains your personal preferences for configuring the Integrity Client. If for some reason your personal `IntegrityClient.rc` file has changed, this command will reload your preferences. Your preferences file should not change, unless you happen to copy someone else's file or if you happen to restore a backup that you made.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]merge`

  controls whether settings from the loaded file should be merged into existing preferences.
- `--rc=`*`value`*

  identifies the file containing settings for running the Integrity Client. The default is the `IntegrityClient.rc` file in your home directory.

See Also

- Miscellaneous: options

im lock

locks a document for editing

Synopsis

```
im lock [--asgroup=group_name] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue id...
```

Description

`im lock` locks a document item for editing. Locking a document for editing prevents others from editing particular fields of that document.

You can lock a document as an individual or for a group that you belong to. When no `--asgroup` option is specified, `im lock` assumes that you are attempting to lock the document for the current user. To successfully lock a document, the administrator must have defined the document type as one which uses locking and you must meet the criteria set by the administrator as to who is eligible to lock documents of that type.

The administrator also defines which fields in a given document type (and its content items) are protected by locking. These are called lockable fields.The list of lockable fields for any given document type always includes, at a minimum, the significant fields for that type.

Locking is not recursive. When you lock a document, only that document is locked. Subdocuments of that document are not locked.

You can lock a document as either an individual user or as a group. When a document is locked by an individual user, only that user can edit the lockable fields of that document. When a document is locked by a group, only members of that group can edit those fields.

Normally, when a document is not locked, any user can edit the lockable fields of that document. However, the administrator can configure Integrity to require users to lock a document before they edit the lockable fields of that document. When such a requirement is in place, no one can edit the lockable fields of an unlocked document.

Note

The requirement to lock a document before editing it is rule-based. As a result, it may not be in effect at all times or all documents of the given document type.

You can view the lock information about any document or content item with the `--showLock` option of the im viewissue command.

Working with Locked Documents

The following issues apply to working with locked documents:

- Users with a lock on a document (either individually or as part of a group) can perform all document model operations that they have permission for. Users without a lock cannot perform any document model operations on the locked document, including versioning.

  Note

  The structure field is a significant field. As a result, when a document is locked, users or groups without a lock cannot make structural changes to that document.

- The reference mode of a content node is independent from locking. It does not matter whether the mode is Share, Reuse, or Author because the document and content items are locked not the shared items.
- When you add new content to a locked document, the new content is locked immediately with the same lock attributes as the rest of the document.
- When you include or insert another document in a locked document, the referenced document is not locked but the content item referencing the including or inserted item is locked.
- You cannot lock a versioned document; however, you can version a locked document.
- Any user can copy content from a locked document. If locked content is pasted into another unlocked document, the paste content is now unlocked.
- When a content item is removed from a locked item, that content item becomes unlocked.
- Locking is not a historical operation. Viewing items historically does not show any lock information. Similarly, locking and unlocking operations do not appear in the history information of an item.

  Field relationships (constraints) as well as editability and relevance rules still apply to locked documents. When a field relationship is such that the target field is a lockable field in a locked document, a user without a lock cannot modify the source field to invalidate the target field.

Examples

For example, the current user is a member of the group Reviewers. Consider the following documents:

- Document 100 is an Integrity document of a type that uses locking and can be locked by anyone. Locking is not required to edit this document.
- Document 200 is an Integrity document of a type that uses locking and can be locked by members of the Authors, Reviewers, and Publishers groups. Locking is required to edit this document.
- Document 300 is an Integrity document of a type that uses locking and can be locked by members of the Publishers group. Locking is not required to edit this document.

Specifying:

```
im lock 100
```

attempts to lock Document 100 for the current user. This succeeds because the document type can be locked by anyone. Once this document is unlocked, any user can edit the lockable fields of this document.

Specifying:

```
im lock --asgroup=Reviewers 200
```

attempts to lock Document 200 for the Reviewers group. This succeeds because the current user belongs to the specified group. When this document is unlocked, no user can edit the lockable fields of this document.

Specifying:

```
im lock 200
```

attempts to lock Document 200 for the current user. Once again, this succeeds because the current user belongs to one of the groups who can lock this document type. When unlocked, no user can edit the lockable fields of this document.

Specifying:

```
im lock 300
```

attempts to lock Document 300 for the current user. This fails because the current user is not a member of the Publishers group, which is the only group that can lock documents of this type.

Specifying:

```
im lock --gui 100
```

presents a dialog that the prompts the user to select the desired locking options and locks Document 100 using the specified options.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--asgroup=`*group_name*

  locks the document for editing by any member of the group specified by `group_name`.

  When you do not specify this option, `im lock` attempts to lock the document for the current user.

- *issue id...*

  the ID of the document you are locking. To specify multiple documents, use a space separated list. All specified documents must be of the same type.

See Also

- Commands: im unlock, im viewissue
- Miscellaneous: options

im logging

modifies Integrity logging levels

Synopsis

`im logging [--category=value] [--debug] [--level=value] [--off] [--on] [--target=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`im logging` modifies Integrity logging levels.

In addition to modifying the `logger.properties` file, you can use the `im logging` command to increase or decrease logging levels for several different categories (as outlined below). You can run this command from a client or server machine, and use it to set client or server side logging. Any category included in the `logger.properties` file can be used. For example:

```
im logging --target=client --category=DEBUG --level=5
```

changes *debug* logging on the client to level *5*.

Note

- The client and server do not need to be restarted after making changes.
- Logging changes last only until the Integrity Server or Integrity Client is restarted.
- You need the `DebugServer` permission in the `mks:im` ACL to run this command.
- `server.log` logs information about this command when it runs.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--category=value`

  where `value` specifies the category name. Possible values are:

  - `DEBUG`

    Logs debug messages. Note: This command overrides the settings in , but only until the Integrity Server is restarted. Additionally, this option does not explicitly log debug exceptions. To log exceptions, open `logger.properties`, uncomment `mksis.logger.exception.includeCategory.DEBUG`, and set the value to 10.

  - `SQL`

    Logs all SQL commands to `server.log`. For example, if you view an issue, the SELECT statement is logged. When editing an issue, the INSERT, UPDATE, and SELECT statements are logged. `--level=5` logs all SQL commands. `--level=10` adds additional information such as Rollback time.

    If SQL logging is enabled on the Integrity Server, you can set SQL logging levels for specific users. Setting the logging level can assist in isolating specific user commands and improving server performance. For example, if logging levels are high and server performance is impacted, reducing logging levels may improve performance.

    To set SQL logging levels for a user, type:

    `im/si --diag=sqllogging username logginglevel`

    where

    username is the user ID of the user whose commands you want to log.

    logginglevel is one of the following number or text values: `high(0), medium (5), low (10), or off (-1)`

    For example, typing:

    `im diag --diag=sqllogging jriley high`

    logs all workflow and document SQL commands for the user `jriley` to the category `SQL-jriley`.

  - `CACHE`

    Logs cache operation information. Levels 0–3 are not verbose. Levels 4–15 are verbose.

  - `SMTP`

    Logs communication between the Integrity Server and SMTP server.

  - `IM-NOTIFICATION`

    Logs e-mail notification.

  - `ACL`

    Logs permission checking to `server.log`. For example, the add label command logs the following:

    ```
    2007-08-16 13:45:17,140 INFO [mksis.IntegrityServer] ACL(5): Check user administrator for permission CreateProject against acl mks:im. Resolved ACL: mks:im Decision: GRANTED
    ```

  - `TRANSACTION`

    Logs all transactions performed by a specific user, for example, `im logging --category=TRANSACTION-jdoe --on` logs all transactions performed by jdoe. To log all cache transactions, type TRANSACTION-system

  - `SILIB`

    Similar to TRANSACTION, this option logs more information about a single command performed by a user, for example, `im logging --category=SILIB-jdoe --on`.

  - `SICIAGENT`

    Provides communications between functionality for workflows and documents, and configuration management, and communications between the Integrity Client and Integrity Server.

  - `REALM`

    Logs NT realm information.

  - `API`

    Logs Integrity API information.

  - `HLL`

    Logs Integrity HLL server information.

  - `LDAP`

    Logs LDAP security scheme information.

- `--debug`

  is equivalent to `--category=DEBUG`.

- `--level=value`

  where `value` specifies the log level (-1 disables logging, 10 logs all messages)

- `--off`

  is equivalent to --level=-1 (no reporting in this category).

- `--on`

  is equivalent to --level=10 (logs all messages in this category).
- `--target=`*`value`*

  specifies the target the debugging is enabled for. Valid values for `--target` are `client`, `server`, and `proxy`. The default value for `--target` is `server`.

See Also

- Commands: im purgeauditlog, im viewauditlog, si viewauditlog
- Miscellaneous: options

im movecontent

moves content to a new location

Synopsis

`im movecontent [--parentID=value] [--insertLocation[=number|first|last|before:name|after:name] [--[no]recurse] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue id...`

Description

`im movecontent` moves the selected issue(s) from their current parent and adds them to a new parent. You can specify where you want to move the content to in the new parent's structural relationship list. For example:

```
im movecontent --parentID=23 --insertLocation=first 15 242 590
```

moves issues 15, 242 and 590 and inserts them in that order at the beginning of the structural relationship list for segment 23.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]recurse=value`

  indicates whether to move all the issues in the moved issue's structural relationship list, or just move the selected issues. If only the selected issues are moved, the child issues are moved up to the removed issue's parent.
- `--insertLocation[=number|first|last|before:name|after:name]`
- determines where the moved issue should go in the new parent issue's structural relationship list. The options are as follows

  `first` inserts the content at the beginning of the list

  `last` inserts the content at the end of the list

  `before`:*name* inserts the content before the specified ID

  `after`:*name* inserts the content after the specified ID

  [0,...] inserts the content at the specified location. If a negative is specified, the content is inserted at the beginning of the list. If the number specified is too large, the content is inserted at the end of the list.
- `--parentID=value`

  the ID of the segment that the content will be moved to. This option is required.
- *issue id...*

  specifies the ID of the issue(s) being copied. Use a space separated list to specify more than one issue ID, for example, `240 241 242`.

See Also

- Commands:im copycontent, im removecontent
- Miscellaneous: diagnostics, options, preferences

im obtainadminlock

set an administrative lock on the Integrity Server

Synopsis

```
im obtainadminlock [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-Ffile|--selectionFile=file)] [-g|--gui][--user=name]
[--hostname=server] [(-N|--no)] [--password=password] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]
```

Description

`im obtainadminlock` allows you to set an administrative lock on the target Integrity Server. Locking is a mechanism for preventing changes on the target server while you are testing a workflow and document configuration on a Staging Server. The Admin Migration Wizard allows you to migrate your final workflow and document configuration from the test environment on the Staging Server to the work environment on your Production Server.

An administrative lock means that the administrative configuration of the server is no longer editable directly through the standard command set. Locking ensures that no other administrative changes can occur to the Production Server while the migration wizard is performing its update.

The administrator can manually lock either the Staging or Production Servers, or both, to prevent administrative changes from taking place. Locking both servers means that no one can make any administrative changes to the workflow and document system. For more information, see the *PTC Integrity Server Administration Guide*.

For example:

```
im obtainadminlock --server=abcFinancial --port=7001
```

sets an administrative lock on the *abcFinancial* server.

Options

This command takes the universal options available to all `im` commands. See the options reference page for descriptions.

See Also

- Commands: im releaseadminlock, im viewadminlock
- Miscellaneous: options

im printissue

prints an Integrity issue

Synopsis

```
im printissue [--outputFile=value] [--asOf=[<date>:label:<label>]] [--[no]showAttachments] [--[no]showChangePackages] [--
[no]showFields] [--[no]showLabels] [--[no]showBranches] [--[no]showHistory] [--[no]showHistoryAscending] [--
[no]showHistoryWithIndirectEdits] [--[no]showRelationships] [--[no]showTimeEntries] [--[no]showHistoryWithComputedField] [--
[no]showTestResults] [--query=[user:]query] [--queryDefinition=query] [--hostname=value] [--port=value] [--password=value] [--
user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=
[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] issue id...
```

Description

`im printissue` prints HTML output of the selected Integrity issue to your console.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--asOf=[<date>:label:<label>]`

  allows you to print an issue as of a historical date or label. For example, to print an issue as of a specific date, type

  ```
  im printissue --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If a value is not provided the issue is printed as of the server's current time. This field is optional.
- `--outputFile=value`

  specifies the path to the file you want the issue data to be printed to. If the specified file already exists, you are asked to confirm an overwrite.
- `--[no]showAttachments`

  specifies whether to display attachments when printing the selected issue. The default is to display attachments.

  Note
  This option controls the visibility of the built in attachment field only. Custom attachment fields are always visible.

- `--[no]showBranches`

  specifies whether to show branches of issues when printing the selected issue. Branches will be printed if the issue has at least one branch.
- `--[no]showChangePackages`

  specifies whether to display change packages when printing the selected issue. The default is to display change packages.
- `--[no]showFields`

  specifies whether to display fields when printing the selected issue. The default is to display fields.
- `--[no]showLabels`

  specifies whether to display labels when printing the selected issue. The default is to print labels if there is more than one.
- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the issue. The default is to not show the history. If you do display the issue history, the information displays in reverse chronological order (the most recent changes appear at the top); however, you can configure the chronological ordering of history information using the `--[no]showHistoryAscending` option.
- `--[no]showHistoryAscending`

  specifies whether to display the issue history in ascending or descending chronological order. This option is used with the `--[no]showHistory` option.
- `--[no]showHistoryWithIndirectEdits`specifies whether to include indirect edits in the item's history. An indirect edit is an edit made to the backing field of a field value attribute (FVA) field. When `--showHistoryWithIndirectEdits` is specified, the results of such edits are shown in the item's history. When this option is not specified or `--noshowHistoryWithIndirectEdits` is specified, the results of such edits are not included.
- `--[no]showRelationships`

  specifies whether to display relationships when printing the selected issue. The default is to display relationships.

  Note
  This option controls the visibility of the built in relationship fields only. Custom relationship fields are always visible.

- `--[no]showTimeEntries`

  specifies whether to display time entries for the issue type, if enabled by your administrator. Time entries indicate time spent working on an issue. Time entries display the entry date, user, source, duration, and notes. Time entries are sorted in descending order of entry date.
- `--[no]showHistoryWithComputedField`

  specifies whether to display the changes to computed fields in the issue history. By default, computed fields are not displayed in the issue history.
- `--[no]showTestResults`

  specifies whether to display test results for the issue type, if the issue type has been given a test management role by your administrator. Test results record the outcome of a test session, test case, or test step.
- `--query=[user:]query`

  specifies the query used to populate the issue selection. Includes the name of the user who the query belongs to and the query name, for example, `mchang:ActiveDefects`. You do not have to specify the user name, but you must specify the query name. This option is useful when multiple users have the same name for a query.

  Note
  Integrity initially assumes that text before the first colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `mchang:ActiveDefects`, Integrity searches for the `Active Defects` query created by `mchang`. If Integrity cannot find the query and/or user, it searches for the `mchang:ActiveDefects` query created by any user.

- `--queryDefinition=query`

  specifies a string to define the query constraints for the query used to populate the issue selection. For the format of the query definition, see the `im createquery` reference page.
- *issue id*

  specifies the ID of the issue you want to print. To specify multiple issues, use a space separated list, for example, `im printissue 51 98 102`. By default, attachments, fields, relationships, and change packages are also printed unless you specify otherwise. Overrides the `--query` option.

See Also

- Commands: im viewissue
- Miscellaneous: options

im projects

displays the list of projects

Synopsis

`im projects [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--[no]asAdmin] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] project...`

Description

`im projects` displays a list of projects for workflows and documents. By default, all projects are selected to be displayed.

Note

If you are a project administrator, only the projects that you are assigned to display, allowing you to specify, edit, and view only those projects. If you have additional permissions that allow you to view projects you are not assigned to, such as `Admin` or `CreateProject`, those projects also display.

Options

This command takes the universal options available to all`im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the project. By default, only the name is shown.
  - `id`

    displays the database ID of the project. This is for PTC - Integrity Support only.
  - `description`

    displays a description of the project.
  - `parent`

    displays if there is a parent that exists for the project.
  - `permittedGroups`

    displays the groups that have visibility or inheritance for the project.
  - `isActive`

    displays the active status of the project.
  - `openImage`

    displays if there is an image icon file used for an open project when it is displayed in the graphical user interface, and if the image is custom or default.
  - `permittedAdministrators`

    displays the users and groups who have been granted access to administer the project.
  - `backingIssueID`

    displays the backing issue ID for the project, if specified.
  - `closedImage`

    displays if there is an image icon file used for a closed project when it is displayed in the graphical user interface and if the image is custom or default.
  - `references`

    displays a list of object references.
- `--[no]asAdmin`

  allows non-admin access to projects. This option is used for third party integrations using the Integrity API. The default setting is `--asAdmin` which requires `ViewAdmin` permissions. Specifying `--noasAdmin` allows users visability to permitted projects only. If a user selects a project without the applicable permissions they will receive an error indicating that the project was not found.

  To learn more about the API, see your *PTC Integrity Integrations Builder Guide*.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields displayed in the CLI.
- *project...*

  specifies the names of the projects to view.

See Also

- Commands: im createproject, im editproject, im viewproject, im deleteproject
- Miscellaneous: options

im propagatetraces

allows you to copy selected trace relationships to a branched document

Synopsis

```
im propagatetraces ][--[no]subSegment] [--[no]batch] [--copyFromEnd=value] [--copyFromStart=value] [--copyToEnd=value] [--
traces=field,field,...] [--hostname=value] [--port=value] [--password=value] [--height=value] [--width=value] [-x value] [-y value]
[--user=value] [(-g|--gui)] [(-?|--usage)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=
[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--cwd=value] [--forceConfirm=[yes|no]] segment id...
```

Description

The `im propagatetraces` command enables you to copy selected trace relationships to a branched document.

For example,

```
im propagatetraces --copyFromStart=801456 --copyFromEnd=801789 --copyToEnd=801112 --fields=Tests 801123
```

copies the `Tests` trace relationships that exists between document `801456` and document `801789` to create new `Tests` trace relationships between document `801123` and `801112`.

When you use the `-g` or `--gui` option, Integrity displays the Finder dialog box.If you browse for issues from the issue selection dialog box, any changes you make to the columns in the Issues View will not be applied to that view when it is accessed through the Integrity Client GUI.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]subSegments`

  specifies whether to follow document node references to subsegments if applicable.
- `--traces=`*`field`*`,`*`field`*`,...`

  trace relationship fields to copy to the branched document. If no fields are specified, all trace relationships found in the source documents are copied.
- `--copyFromStart=`*`value`*

  specifies the ID of the document you want to copy trace relationships from
- `--copyFromEnd=`*`value`*

  specifies the ID of the document that has trace relationships with the document identified in `--copyFromStart`.
- `--copyToEnd=`*`value`*

  specifies the document that you want to create the trace relationships for.
- *segment id...*

  specifies the ID of the document that you want to copy trace relationships to.

See Also

- Commands: im branchsegment, im viewsegment, im createsegment
- Miscellaneous: options

im purgeauditlog

deletes and archives specified audit log records on the Integrity Server

Synopsis

`im purgeauditlog [--before=`*`value`*`] [--maxFileSize=`*`value`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [--[no]backup] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui][--user=`*`name`*`] [--hostname=`*`server`*`] [(-N|--no)] [--password=`*`password`*`] [--port=`*`number`*`] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)] [(-Y|--yes)]`

Description

`im purgeauditlog` allows you to delete and archive specified records from the Integrity Server audit log. You can choose to delete or archive records older than a specified date, thereby controlling the size of the audit log table in the database. Each purging or archiving operation is also recorded in the audit log if `mksis.auditor.is=true`.

By default, the purged records are automatically archived in a backup directory on the Integrity Server. The audit archive is stored in a comma-separated value format (CSV). The default location of the audit log backup is controlled by the `mksis.auditlogbackupdir=` property in the `is.properties` file. For more information, see the *PTC Integrity Server Administration Guide*.

For example, the following command purges all audit log records prior to 10:00AM on January 20, 2007 and creates a CSV file archive of the records in the default directory:

```
im purgeauditlog --before="01/20/2007 10:00:00 AM"
```

The following command purges all audit records prior to 10:00AM on January 20, 2007 without creating a CSV file archive:

```
im purgeauditlog --before="01/20/2007 10:00:00 AM" --nobbackup
```

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--before=`*`value`*
- where *`value`* specifies a single date in the format `MM/dd/yyyy h:mm:ss [AM|PM]`. All audit records created prior to the specified date are permanently deleted or archived from the database, according to the options you specify. The date value cannot be used to filter for records newer than a given date. By default, purged records are archived to the `<installdir>\data\auditbackup` directory (where `<installdir>` is the path to the directory where you installed the Integrity Server). The file is assigned the name `auditlogbackup_*.csv` (where `_*` is `_MMDDYYYYhhmmss_0` and `_0` is incremented if a file of the same name already exists).
- `--maxFileSize=`*`value`*

  specifies the maximum file size (in megabytes) allowed for the created archive audit log. By default, the maximum file size is set at 50 MB. You can specify file sizes up to the maximum available on your file system.
- `--[no]backup`

  allows you to delete audit log records without first archiving those records. If you specify `--nobackup`, you are asked to confirm the deletion of the target records before the operation is completed.

See Also

- Commands: im viewauditlog, si purgeauditlog, si viewauditlog
- Miscellaneous: options

im putdbfile

puts a workflow and document configuration file into the database

Synopsis

```
im putdbfile [--encoding=value] [--input=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...
```

Description

Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing using the `im getdbfile` command. Once you are finished editing these files, you can store them in the database again using the `im putdbfile` command.

Tip

When updating an item presentation template (IPT) in the database from the CLI, the Integrity Client GUI may not display the IPT change, even after restarting the client. This issue may also occur when promoting an IPT change from a staging server to a production server. This occurs when the IPT cache on the Integrity Server is not updated correctly.

To clear and refresh the IPT cache on the server, use the following command:
`im diag --diag=flushIPTCache`

After using this command, PTC recommends restarting the Integrity Client GUI to ensure updated IPTs display correctly.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=`*`value`*

  specifies the code set the file is saved in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--input=`*`value`*

  specifies the name of the file on the local file system containing the input.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `im diag --diag=listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\`*`templatename`*`.xml`, for example, `im putdbfile --input=c:/defect.xml data/im/issue/templates/defect.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

im queries

displays the list of queries

Synopsis

`im queries[--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value][-y value] [--user=value] [--hostname=value] [--password=value] [--port=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] query...`

Description

`im queries` displays the list of Integrity queries. By default, the command displays all queries that are currently shared to you.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  specifies the query fields to display and the width of each field in characters. If the output is directed to the GUI, the width is specified in pixels.

  For example,

  ```
  im queries --fields=name,description,lastModified --fieldsDelim=, "Release 2 Features"
  ```

  displays the name, description, and date of the last modification for the Release 2 Features query, with each field separated by a comma.

  The query fields you can specify are:

  - `createdBy`

    displays the name of the user who created the query.
  - `description`

    displays a description of the query.
  - `fields`

    displays the fields visible in the associated column set.
  - `id`

    displays the database ID of the query. This is for PTC - Integrity Support only.
  - `image`

    displays whether there is an image, and if so, whether it uses the default funnel image or a custom image.
  - `lastModified`

    displays the date the query was last modified.
  - `name`

    displays the name of the query.
  - `shareWith`

    displays the users and groups that the query is shared with.
  - `sortDirection`

    displays the order that issues display in the associated column set.
  - `sortfield`

    displays the field that issues are sorted by in the associated column set.
  - `queryDefinition`

    displays the query definition that specifies the query constraints.
  - `references`

    displays all system provided and user objects that reference the query.
  - `isAdmin`

    displays whether the query is a system provided object.
  - `--fieldsDelim=value`

    specifies the string to be used as a delimiter between fields.
  - *query...*

    identifies the names of the queries to view.

SEE ALSO

- Commands: im copyquery, im createquery, im deletequery, im editquery, im viewquery
- Miscellaneous: options

im refmode

sets the sharing behaviour for content

Synopsis

`im refmode [--refmode=[reuse|share] [shareID=value] [[no]useDefault] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] node id...`

Description

`im refmode` allows you to set or change the reference mode on individual or multiple nodes in a segment.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--refmode=[reuse|share]`

  allows you to specify the reference mode for content. Selecting `--share` creates a share relationship while `--reuse` removes one if it exists on a node. The available reference modes explained:

  `Share` :When the author changes their content you will automatically see the changes reflected in your document.

  `Reuse`: If the author changes content, the content will be branched and a copy of the changes is created. The reuser becomes the author of the new shared item. When this option is specified, neither `--shareID` nor `--useDefault` can be specified.
- `--shareID=value`

  When this option is specified, the reference mode does not need to be defined. `Share` is assumed.
- `--[no]useDefault`

  specifies to use the ID of the originating node from where the you want to change was branched. When `--useDefault` is specified, `--shareID` is ignored. When `--[no]useDefault` is specified, `--shareID` is required. Default is `--nouseDefault`.
- *node id...*

  the node ID(s) of the content you want to change. Use a space separated list to specify more than one issue ID, for example, `240 241 242`.

See Also

- Commands: im createsegment, im viewsegment, im insertsegment
- Miscellaneous: diagnostics, options, preferences

im relationships

displays all the relationships for one or more Integrity issues

Synopsis

```
im relationships [--asOf=[<date>label:<label>] [--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...] [--
expandRelationshipDirection=[forward|backward|both]] [--expandRelationshipFields=field,field,...] [--[no]ApplyDisplayPattern] [--
structureFieldIconDisplayField=field] [--structureFieldDisplayFormat=value] [--focusIssueID=value] [--expandLevel=value] [--
[no]batchEdit][--nodeDisplayFields=value] [--[no]showFieldNodes] [--[no]showXHTML] [--query=[user:]query] [--queryDefinition=query]
[--height=value] [--width=value] [-x value] [-y value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-g|--
gui)] [(-?|--usage)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--
no)] [(-Y|--yes)] [--cwd=value] [--forceConfirm=[yes|no]]issue id... [--[no]showEditPanel] [--[no]confirmSave] [--[no]showIssueView]
```

Description

The `im relationships` command displays a relationship tree. You can expand the tree to show the relationship fields for the root issue(s), and expand each relationship field to see the related issues in that field. You can control which relationship fields display in the view, and you can perform functions on the issues in the view.

For example,

```
im relationship --field=ID,type,state,summary 80713
```

displays the `ID`, `type`, `state`, and `summary` for all issues that issue 80713 has a relationship with.

Note

This command is only supported with the `-g` or `--gui` option. When you use the `-g` or `--gui` option, Integrity displays an issue selection dialog box. If you browse for issues from the issue selection dialog box, any changes you make to the columns in the Issues View will not be applied to that view when it is accessed through the Integrity Client GUI.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...`*

  allows you to select fields to be printed, specified in the format *field[:width],field[:width],...*. Specifying the column *[:width]* (in pixels) for each field is optional.

  For rich content fields, you can specify display patterns to display field values as rich content (`:rich`) or plain text (`:plain`). The rich content display pattern displays the underlying HTML elements and attributes in the rich content field. For example,

  ```
  --fields=Description::rich
  ```

  displays the Description field value with HTML elements and attributes. By default, rich content fields display plain text.
- `--[no]applyDisplayPattern`

  specifies whether to apply a display pattern to numeric fields. Display patterns are configured by your administrator and allow you to quantify integer and floating point field values, for example, as currency or percentages. `--applyDisplayPattern` is enabled by default.

  Note

  If you use scripts, PTC recommends using the`--noapplyDisplayPattern`option to avoid being impacted by administrative changes to display patterns.

- `--focusIssueID=`*`value`*

  specifies the item ID to highlight in the tree pane when the segment displays. This colour is defined in the `View > Options` dialog in the context of the Document view.
- `--expandRelationshipDirection=[forward|backward|both]`

  specifies the type of relationship fields to display and expand: forward, backward, or both.
- `--expandRelationshipFields=`*`field,field,...`*

  specifies the relationship fields to display and expand.

  Note the following:

  - Relationship fields include standard forward and backward relationship fields, as well as query backed relationship (QBR) and item backed picklist (IBPL) fields (considered as forward relationship fields).
  - The `--expandRelationshipDirection` and `--expandRelationshipFields` options are mutually exclusive. If both options are specified, the `im relationships` command fails and displays an error message with the reason.
- `--asOf=[`*`<date>|label:<label>`*`]`

  allows you to view relationships as of a historical date or label. For example, to view relationships on an issue as of a specific date, type

  ```
  im relationships --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If a value is not provided the relationships display as of the server's current time. This field is optional.
- `--structureFieldIconDisplayField=`*`field`*

  specifies the field from which the icon is taken. This is the icon that will be displayed for each node in the Outline pane.
- `--expandLevel=`*`value`*

  specifies to expand the nodes to a specified level, for example, 1, 2, 3. The default is one.
- `--[no]showXHTML`

  specifies to enable rich text fields to be displayed as XHTML.
- `--structureFieldDisplayFormat=`*`value`*

  specifies the fields and style that should be displayed for the tree nodes.

  defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  `--structureFieldDisplayFormat` options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires { } to surround each field). For example:

  ```
  im relationships --structureFieldDisplayFormat="{ID},{Summary}"
  ```
- `--[no]showFieldNodes`

  specifies whether or not to display field nodes.
- `--query=`*`[user:]query`*

  specifies the query to use to populate the issue selection. If not specified, Integrity uses the most recently run query.
- `--queryDefinition=`*`query`*

  specifies a string to define the query constraints for the query used to populate the issue selection. For the format of the query definition, see the `im createquery` reference page.
- *issue id...*specifies the ID of the issue(s) you want to view relationships for. Use spaces to specify more than one issue, for example `34 23`.

If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, 184-1.2.

`--[no]showEditPanel`

specifies whether to show the embedded item edit pannel. This option can be used in conjunction with the `--[no]showIssueView` option. If conflicts arise, the`--[no]showEditPanel` option always wins.

`--[no]showIssueView`

specifies whether to show the embedded item view. This option can be used in conjunction with the `--[no]showEditPanel` option. If conflicts arise, the `--[no]showEditPanel` option always wins.

`--[no]confirmSave`

specifies whether to prompt you to confirm changes are to be saved. This turns on and off whether you a confirmation window oepns when you save in this view.

See Also

- Commands: im copyissue, im createissue, im viewissue, im editissue
- Miscellaneous: options

im releaseadminlock

releases an administrative lock on the Integrity Server

Synopsis

```
im releaseadminlock [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-F file|--selectionFile=file)] [-g|--gui][--
user=name] [--hostname=server] [(-N|--no)] [--password=password] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]
```

Description

`im releaseadminlock` releases the administrative lock on the target Integrity Server. Locking is a mechanism for preventing changes on the target server while you are testing a workflow and document configuration on a Staging Server. The Admin Migration Wizard allows you to migrate your final workflow and document configuration from the test environment on the Staging Server to the work environment on your Production Server.

An administrative lock means that the administrative configuration of the server is no longer editable directly through the standard command set. Locking ensures that no other administrative changes can occur to the Production Server while the migration wizard is performing its update.

The administrator can manually lock either the Staging or Production Servers, or both, to prevent administrative changes from taking place. Locking both servers means that no one can make any administrative changes to the workflow and document system.

For example:

```
im releaseadminlock --server=abcFinancial --port=7001
```

releases the administrative lock on the *abcFinancial* server.

Caution

Releasing the lock on the Production Server means that the lock binding between the Staging Server and Production Server is lost and the two servers are no longer in sync. Further staging work will therefore require a new copy of the database from the Production Server. For more information, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `im` commands. See the options reference page for descriptions.

See Also

- Commands: im obtainadminlock, im viewadminlock
- Miscellaneous: options

im removebaseline

removes baseline labels from documents

Synopsis

`im removebaseline [(-L label|--label=label)] [--[no]subSegment][--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] segment id...`

Description

`im removebaseline` removes a baseline from a document. A baseline is a meaningful date and time in a set of issues, marked with a label.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `-L label`
- `--label=label`

  identifies the baseline label to delete. Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.

  Note

  Labels that include spaces must be enclosed by quotes.

- `--[no]subSegment`

  specifying `--subSegment` specifies to remove the baseline on the referenced subsegment instead of the owning segment.

- *segment id...*

  the IDs of the documents you want to remove the baseline label from. Use a space separated list to specify more than one item ID, for example, `240 241 242`.

See Also

- Commands: im baseline
- Miscellaneous: diagnostics, options, preferences

im removecontent

removes content from a document

Synopsis

`im removecontent [--[no]recurse] [--[no]confirm] [--quiet] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] issue id...`

Description

`im removecontent` removes the selected content from their parent segments. If the content being removed is is shared with other segments, then the content is removed from its current parent only. For example:

```
im removecontent 15, 242 590
```

removes content IDs 15, 242 and 590 from their respective parents.

Note
You cannot remove shared content since other documents depend on it.

`--[no]confirm`

specifies whether to be prompted to confirm the deletion of the content. By default, you are prompted to confirm the deletion.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]recurse=`*`value`*

  indicates whether to remove all child content if the content being removed is a parent.
- *issue id...*

  specifies the issue ID of the content being removed. Use a space separated list to specify more than one issue ID, for example, `240 241 242`.

See Also

- Commands: im createcontent, im copycontent, im movecontent
- Miscellaneous: ACL, diagnostics, options, preferences

im reports

displays a list of all Integrity reports visible to the user

Synopsis

`im reports [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--name=value] [--description=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [--height=value] [--width=value] [-x value] [-y value] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]`

Description

`im reports` displays the list of Integrity reports that are visible to you.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`specifies the report fields to display and the width of each field in characters. If the output is directed to the GUI, the width is specified in pixels.

  For example,

  ```
  im reports --fields=name,description,lastModified --fieldsDelim=,
  ```

  displays the name, description, and date of the last modification for all Integrity reports that are visible to you.

  The report fields you can specify are:

  - `createdBy`

    displays the name of the user who created the report.
  - `recipeParams`

    displays the report recipe parameters.
  - `description`

    displays a description of the report.
  - `lastModified`

    displays the date the report was last modified.
  - `name`

    displays the name of the report.
  - `shareWith`

    displays the users and groups that the report is shared with.
  - `query`

    displays the name of the query the report uses to run on.
  - `id`

    displays the database ID of the report. This is for PTC - Integrity Support only.
  - `references`

    displays all system provided and user objects that reference the report.
  - `isAdmin`

    displays whether the report is a system provided object.
  - `--fieldsDelim=value`

    specifies the string to be used as a delimiter between fields.

See Also

- Commands: im createreport, im editreport, im copyreport, im viewreport, im runreport, im deletereport
- Miscellaneous: options

im runchart

runs an Integrity chart

Synopsis

```
im runchart [--fieldFilter=field=[value,value,...]] [--graphStyle=
[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]] [--chartFootnote=value] [--
chartTitle=value] [--description=value] [--[no]displayDescription=value] [--imageHeight=value] [--imageWidth=value] [--
outputFile=value] [--[no|confirm]overwriteOutputFile] [--query=[user:]query] [--queryDefinition=query] [--hostname=value] [--
port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-g|--gui)] [--quiet] [--settingsUI=
[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
[user:]chart
```

Description

`im runchart` runs an Integrity chart, generating chart data. For more information on charts, refer to the *PTC Integrity User Guide*.

For example,

```
im runchart --gui jriley:"Release 2 Features By Priority"
```

displays the Release 2 Features By Priority chart created by `jriley`.

Note the following:

- All charts are subject to visibility rules set by your administrator. Visibility rules restrict access to specific information based on project and/or issue type. For more information, see the *PTC Integrity Server Administration Guide*, or see your administrator.
- For the best performance, avoid generating trend or issue fields trend charts using short intervals over long time spans.
- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fieldFilter=field=[value,value,...]`

  specifies project filter values to apply to the chart.

  Note
  If you are running a distribution chart that is based on a query with a project filter, the project filter values you specify here are combined with the query filter. In some cases, this could result in no data being returned. For example, if a chart is based on a query with an embedded project filter value of `Cosmo1`, and you specify a project filter value of `Cosmo2`, no data is returned.

- `--graphStyle=[VerticalBar|VerticalStackedBar|HorizontalBar|HorizontalStackedBar|Pie|Line|Table|XY|Bubble]`

  overrides the graph style defined for the chart.
- `--chartFootnote=value`

  Overrides the chart footnote with the specified *value*.
- `--chartTitle=value`

  Overrides the chart title with the specified *value*.
- `--description=value`

  Overrides the chart description with the specified *value*.
- `--[no]displayDescription=value`

  Overrides the chart display description with the specified *value*. When you specify `--nodisplayDescription`, the chart description is not displayed.
- `--imageHeight=value`

  specifies the height of the image, in pixels; *value* must be a whole number. This option does not apply to table style charts.
- `--imageWidth=value`

  specifies the width of the image, in pixels; `value` must be a whole number. This option does not apply to table style charts.
- `--outputFile=value`

  specifies the name of the file you want the chart data to be saved in.

  Note
  Table style charts are saved in comma separated values (csv) format. All other styles of charts are saved in PNG format.

- `--[no|confirm]overwriteOutputFile`

  specifies whether to overwrite if the file specified in the `--outputFile` option already exists.
- `--query=[user:]query`

  specifies a query to override the chart query.
- `--queryDefinition=query`

  specifies a string to define the query constraints to override the chart query. For details of the query format, see the `im createquery` reference page.

  Note
  To reference the original query in the new query so that any changes to the original query are reflected in the new query, define *<subquery>* as *subquery[OriginalQuery]*

.

- *[user:]chart*

  specifies the name of the chart to run, and the user who created it. If you are running a chart you created, you do not have to specify the user name, but you must specify the chart name.

  Note
  Integrity initially assumes that text before the colon (:) is a user name and text after it is a chart name. If Integrity fails to find a matching user name and chart name, it searches for a chart name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` chart created by `jhoyt`. If Integrity cannot find the chart and/or user, it searches for the `jhoyt:CosmosDefects` chart created by any user.

See Also

- Commands: im createchart, im editchart, im copychart, im viewchart, im deletechart, im charts imrunchart
- Miscellaneous: options

im rundashboard

runs an Integrity dashboard

Synopsis

`im rundashboard [--fieldFilter=`*`field=[value,value,...]`*`] [--hostname=`*`value`*`] [--port=`*`value`*`] [--password=`*`value`*`] [--user=`*`value`*`] [(-?|--usage)] [(-F `*`value`*`|--selectionFile=`*`value`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`value`*`] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--height=`*`value`*`] [--width=`*`value`*`] [-x `*`value`*`] [-y `*`value`*`] `*`[user:]dashboard`*

Description

`im rundashboard` runs an Integrity dashboard, generating dashboard data. This command only runs when used with a `-g` or `--gui` option. For more information on dashboards, refer to the *PTC Integrity User Guide*.

Note

Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fieldFilter=`*`field=[value,value,...]`*

  specifies the field filter to be applied to the dashboard. Currently, only project field filters are supported.

  If the dashboard has a defined project filter type of `Restricted`, you can only specify filter values that are a subset of the filter values defined for the dashboard.

  If the dashboard has a defined project filter type of `Fixed`, this option is invalid.

  If the dashboard has a defined project filter type of `None`, this option is invalid.

  Note

  Depending on how you design your dashboard layout, the dashboard filter may not be applied to chart, report or query dashboard components.

- `[username:]dashboard`

  specifies the name of the dashboard to run, and the user who created it. If you are running a dashboard you created, you do not have to specify the user name, but you must specify the dashboard name.

  Note

  Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the dashboard named `ProjectOverview` created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the dashboard named `jhoyt:ProjectOverview` created by any user.

See Also

- Commands: im createdashboard, im editdashboard, im copydashboard, im viewdashboard, imdeletedashboard, imdashboards
- Miscellaneous: options

im runreport

runs an Integrity report

Synopsis

```
im runreport [--fieldFilter=field=[value,value,...],field=[value,value,...],...] [--asOf=<date>:label:<label>] [--basetime=<date>:label:<label>] [--issues=value] [--issues2=value] [--outputFile=value] [--[no|confirm]overwriteOutputFile] [--query=[user:]query] [--queryDefinition=query] [--param=value] [--[no]substituteParams] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [username:]report
```

Description

`im runreport` runs an Integrity report, generating report data. For more information on reports, refer to the *PTC Integrity User Guide*.

For example,

```
im runreport --issues=10001,10002,10003 jriley:"Docs Activity"
```

displays the Docs Activity report created by `jriley` showing data for items 10001, 10002, and 10003.

Note

Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields and time entries vary based on the time zone that a user is operating in.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--fieldFilter=field=[value,value,...],field=[value,value,...],...`

  specifies field filters and values to apply to the report. You can specify multiple field filters. You can filter based on values in the Project, Type, or State fields, or on values in any user, group, picklist or item backed picklist field.
- `--asOf=<date>:label:<label>`

  specifies the historical date or label to use for the issues included in the report. To specify a date and time, type *MM/dd/yyyy h:mm:ss [AM|PM]*. Each issue included in the report will be as of the date and time specified. This option is used with the `--issues` option.
- `--basetime=<date>:label:<label>`

  specifies a date or label to use for comparing two reports. To specify a date and time, type `MM/dd/yyyy h:mm:ss [AM|PM]`. This option is used with the `--asOf` option. Two reports are run: one based on the `--asOf` label or date/time, and one based on the `--basetime` label or date/time. The reports are compared and the results automatically display in the Visual Difference tool. For more information on this tool, see the *PTC Integrity User Guide*.
- `--issues=value`

  specifies the issues to report on, where `value` is a comma-delimited list of issue IDs. This overrides the query that the report is based on.
- `--issues2=value`

  specifies the issues to report on, applied to the base time. This overrides the query that the report is based on.
- `--outputFile=value`

  specifies the name of the file you want the report data to be saved in.
- `--[no|confirm]overwriteOutputFile`

  specifies whether to overwrite if the file specified in the `--outputFile` option already exists.
- `--query=[user:]query`

  specifies a query to override the report query.
- `--queryDefinition=query`

  specifies a string to define the query constraints to override the report query. This option cannot be used with historical reports. For details of the query format, see the `im createquery` reference page.

  Note

  To reference the original query in the new query so that any changes to the original query are reflected in the new query, define `<subquery>` as `subquery[OriginalQuery]`.

- `--param=value`

  specifies a keyword contained in the `%arg%>` tag of the report definition and the parameter value to replace the keyword. To specify a keyword and parameter value, type `--param=keyword=parameter value`, for example, `im runreport --param=segmentname=SegmentA BugReport`. Keywords and parameter values allow you to configure report content at runtime. Specify `--param` for each additional keyword and parameter value. For more information on keywords in report definitions, see the *PTC Integrity Server Administration Guide* or contact your administrator.

  Note

  The report must contain `%arg%>` tags; otherwise, this option is ignored when you run the report. A parameter expansion not specified in `--param` substitutes an empty string.

- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `[username:]report`specifies the name of the report to run, and the user who created it. If you are running a report you created, you do not have to specify the user name, but you must specify the report name.

  Note

  Integrity initially assumes that text before the colon (:) is a user name and text after it is a report name. If Integrity fails to find a matching user name and report name, it searches for a report name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the report named `CosmosDefects` created by `jhoyt`. If Integrity cannot find the report and/or user, it searches for the report named `jhoyt:CosmosDefects` created by any user.

See Also

- Commands: im createreport, im editreport, im copyreport, im viewreport, im deletereport, im reports
- Miscellaneous: options

im runtrigger

runs an event trigger

Synopsis

`im runtrigger [--invocationArgs=key1=value1)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] trigger...`

Description

`im runtrigger` runs an event trigger for workflows and documents. For example:

```
im runtrigger sendmail
```

runs the *sendmail* trigger.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--invocationArgs=key1=value1`

  specifies invocation arguments to manually scheduled triggers in the format *key=value*. These dynamic arguments are passed to the triggers in addition to any static configuration parameters that may have been set up when the trigger was created. Argument names and values must match the same number of arguments and argument names that the script expects. For example, if the script receives invocation arguments and reads one int parameter called *myint*, you would type `--invocationArgs=myint=1`.

Note

You cannot delimit the invocation arguments with commas. You must specify the `--invocationArgs` option additional times as necessary.

- *trigger...*

  specifies the name of the scheduled trigger. Use spaces to specify more than one scheduled trigger.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im deletetrigger, im triggers, im echo
- Miscellaneous: options

im serveralerts

displays Integrity Server alert messages for all currently connected servers

Synopsis

`im serveralerts [--height=value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`im serveralerts` displays Integrity Server alert messages for all servers that you are currently connected to. The alert message displays who sent the message, the server it came from, when it was sent, and the message. If you are not connected to any servers, a message informs you that there are no alert messages. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

Note the following:

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

See Also

- Commands: im viewserveralert, im admingui, im servers
- Miscellaneous: options

im servers

displays the current connections to an Integrity Server

Synopsis

`im servers [--[no]showVersion] [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`im servers` displays active server connections in the format *user@host_name:port*.

The default server connection is indicated by *user@host_name:port(default)*.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]showVersion`

  controls whether to show build version information for the connected server. The presentation of this information is in the format `[Build: 2345]`.

- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

See Also

- Commands: im connect, im disconnect
- Miscellaneous: options

im setnotification

sets e-mail notification for a user or group

Synopsis

```
im setnotification [--email=value] [--notificationRule=value] [--notificationRuleFile=value] [(-Ffile|--selectionFile=file)] [--
quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch]
[--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`im setnotification` sets e-mail notification for a user or group. For users, you can also use the `im createuser` and `im edituser` commands to set e-mail notification. For groups, you can also use the `im creategroup` and `im editgroup` commands to set e-mail notification.

Note

- You must have the following permissions assigned to you: `allow ViewMyNotification` to view e-mail notification settings, and `allow ModifyMyNotification` to create and edit notification rules.
- To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date field, type *today*. This option can be specified for a date field or a date field configured to display the date and time. To specify the current date and time for a date field, type *now*. This option can be specified only if the date field is configured to display the date and time. To specify an empty value for the date field, type *none*.
- When specifying a user, you can specify the user's name or "`me`". "`me`" is a symbolic user which refers to the user that the notification rule is created for. This is useful if you want to create a common notification rule and share it with other users.
- Configuration management project fields cannot be used in e-mail notification rules.
- E-mail notification is subject to project, type, and field visibility rules. Only users that have visibility for a given project and type receive e-mail notification for issues related to that project and type. In addition, e-mail notifications include only the fields they have permission to view.
- E-mail notifications are evaluated on the Integrity Server 's time zone; however, symbolic dates in rules are evaluated on the Integrity Client's time zone.
- Configuration management project and attachment fields cannot be specified.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--email=`*value*

  specifies the e-mail address for the user or group. To specify multiple e-mail addresses, separate them with semi-colons (;).
- `--notificationRule=`*rule*

  specifies the notification rule for this principal. For the rule syntax, see Specifying Rules on the options reference page.
- `--notificationRuleFile=`*filename*

  specifies the file that contains the rules to read. For the file content format, see Specifying Rules on the options reference page.

See Also

- Commands: im createuser, im edituser, im creategroup, im editgroup
- Miscellaneous: options

im setprefs

sets preferences

Synopsis

`im setprefs [--command=value] [--[no]resetToDefault] [--[no]save] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...`

Description

`im setprefs` sets preference options for Integrity. These settings are used to determine default behaviors for other commands - each command option has a preference key associated with it. The `im viewprefs` command lists the commands and preference keys. Changes to your preferences are either for the current client session (until `im exit` is used) or can be permanently saved in your system's `home` directory, in the file named `IntegrityClient.rc`, using the `--save` option.

For example:

```
im setprefs --command=viewissue --save showTimeEntries=true
```

sets the preferences for the `im viewissue` command to always display time entries for the issue.

Caution: Do not edit the `IntegrityClient.rc` file manually. Preferences that appear more than once in the `IntegrityClient.rc` file can cause unpredictable behavior in Integrity.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--command=`*value*

  identifies the command to be set.
- `--[no]resetToDefault`

  controls whether to revert specified settings to the default values as shipped with Integrity Client. If specifying `--resetToDefault`, you must not specify individual preferences.
- `--[no]save`

  controls whether changes should be permanently saved.
- `--[no]ask`

  controls prompts to the user for specific preferences. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:
  - `once`

    asks the user the first time only, and then uses the provided value every time after.
  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).
  - `element-last`

    asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.
- `--ui=[unspecified|gui|cli|api`

  controls whether to apply the preference to the graphical user interface, the command line interface, or when the interface is unspecified. By default, `--ui=cli` is implied when using `im setprefs`. To set preferences for GUI behavior, however, you should specify `--ui=gui`. For example, to set the `showHistory` preference to be true in the GUI for the `im printissue` command, you would type:

  ```
  im setprefs --command=printissue --ui=gui showHistory=true
  ```

  These correlate to settings in the`IntegrityClient.rc` file that have the `gui.im.` or `cli.im.` prefix, or the `im.` prefix when it is unspecified.
- *`string...`*

  identifies the preference string. If you specified the `--resetToDefault` option, then you only need to specify the preference name; otherwise specify a value for the preference. Use spaces to specify multiple preferences.

See Also

- Commands: im loadrc, im viewprefs
- Miscellaneous: options, preferences

im setproperty

configures workflow and document properties stored in the database

Synopsis

`im setproperty [--comment=value] [--restoreDefault] [--value=value] [-?|--usage] [-N|--no] [-Y|--yes] [--hostname=server] [--port=number] [--user=name] [--password=password] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [-Ffile|--selectionFile=file] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...`

Description

`im setproperty` configures workflow and document properties stored in the database. Properties specify information that affects the operation of the Integrity Server, workflows and documents, and configuration management. Other properties are stored and configured in properties files on the server's file system. For a complete list of configurable properties and possible values, see the *PTC Integrity Server Administration Guide*.

Note

- Some properties require the server to be restarted for the changes to take effect.
- Access to configuring properties is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document properties, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management properties, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all properties.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--comment=value`

  specifies a comment associated with the change made to the property. While PTC recommends specifying a comment for troubleshooting purposes, a comment is optional.
- `--restoreDefault`

  restores the property to the default value.
- `--value=value`

  specifies the new value of the property.
- *`string...`*

  specifies the name of the property you want to configure.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im admingui, integrity gui
- Miscellaneous: diagnostics, options, preferences

im settimeentries

creates, edits, or deletes time entries

Synopsis

```
im settimeentries [--actionDefinition=value] [--actionDefinitionFile=value] [--[no]processBatch] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]]
```

Description

Once an issue has been created and assigned to you, you can record time spent working on the issue by creating a *time entry*. A time entry records the duration of time spent working on the issue in hours for a specific day, and, optionally, any relevant notes.

If you have the `TimeTrackingAdmin` permission assigned to you, you can also create time entries on the behalf of other users. This is useful if your organization has a designated role for entering employee time entries.

You can also edit existing time entries to correct durations and notes, and delete time entries.

To create a time entry, note the following:

- Typically, you create a time entry for an issue assigned to you; however, you can create a time entry for any issue that you have visibility to.
- Your administrator configures which states allow the creation of time entries.
- The issue type must allow time entries. Your administrator configures which types allow time entries.

To edit or delete a time entry, note the following:

- Typically, you edit or delete a time entry for an issue assigned to you; however, you can edit or delete a time entry for any issue that you have visibility to.
- Your administrator configures which states allow the editing or deletion of time entries.
- The time entry must be created by you, or you must have the `TimeTrackingAdmin` permission assigned to you.

Options

This command takes the universal options available to all *im* commands, as well as some general options. See the options reference page for descriptions.

- `--actionDefinition=`*value*

  specifies a set of time entry actions to perform using XML tags.

  Note the following:

  - For create, edit, and delete operations, each time entry consists of the issue ID, user, entry date, and source. If these values are not specified, the source defaults to the built-in \u201cIntegrity\u201d source.
  - If the duration or notes are not included in the operation, existings value are ignored. If these values are included and empty, the value in the corresponding time entry is cleared.
  - `Duration`values can be any positive floating point number or 0.
  - `Notes` values can be any amount of text or an empty string.
  - Time entry actions are performed on the Integrity Server in the order they are written in the XML file. Errors occur if you attempt to edit or delete a time entry that does not exist (either created earlier in the XML file or in a previous command invocation).
  - The following filters can be specified in a delete all time entries operation: issue:*ID*, user:*user name*, fromDate:*date*, toDate:*date*, source:*source name*.
  - Dates are locale specific; the format you specify appears when you view the time entry. For example, if you are in Japan and you specify `yyyy/MM/dd`, the same date format displays when you view the time entry.

  To create a time entry, type the following:

  ```
  <ActionList>
   <Create issueId="issue ID" user="user name" entryDate="date">
    <Duration>hours</Duration>
    <Notes>text string</Notes>
   </Create>
  </ActionList>
  ```

  To edit a time entry, type the following:

  ```
  <ActionList>
   <Edit issueId="issue ID" user="user name" entryDate="date">
    <Duration>hours</Duration>
    <Notes>text string</Notes>
   </Edit>
  </ActionList>
  ```

  To delete a time entry, type the following:

  ```
  <ActionList>
   <Delete issueId="issue ID" user="user name" entryDate="date"/>
  </ActionList>
  ```

  Note
  This operation requires the`TimeTrackingAdmin` permission.

  ```
  <ActionList>
   <DeleteAll issueId="issue ID" filter="value"/>
  </ActionList>
  ```

  To perform multiple time entry actions, specify additional create, edit, or delete time entry tags within the <ActionList></ActionList> tags, for example:

  ```
  <ActionList>
   <Create issueId="50546" user="jriley" entryDate="7/25/2005" source=\u201dmks_integrity\u201d>
    <Duration>6.5</Duration>
    <Notes>finished print feature</Notes>
   </Create>
   <Create issueId="50546" user="jriley" entryDate="7/25/2005">
    <Duration>6</Duration>
    <Notes>sent out for code review</Notes>
   </Create>
   <Delete issueId="50545" user="user2" entryDate="7/25/2005"/>
   <DeleteAll issueId="issue ID" user="jriley" toDate="7/28/2006"/>
   <Edit issueId="50545" user="jriley" entryDate="7/25/2005">
    <Duration>10.5</Duration>
    <Notes>worked overtime</Notes>
   </Edit>
   <Edit issueId="50546" user="jriley" entryDate="7/25/2005">
    <Duration>8.5</Duration>
    <Notes></Notes>
   </Edit>
  </ActionList>
  ```

- `--actionDefinitionFile=`*value*

  specifies a file containing the XML time entry actions to perform. See `--actionDefinition=`*value* for the file format.

- `--[no]processBatch`

  specifies whether to submit the time entry actions as a batch operation.

See Also

- Commands: im timeentries
- Miscellaneous: options

im states

displays a list of states

Synopsis

`im states [--fields=field1[:width1],field2[:width2]... [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] state...`

Description

`im states` displays a list of states for workflows and documents. By default, all states are listed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the state. By default, only the name is shown.
  - `id`

    displays the database ID of the state. This is for PTC - Integrity Support only.
  - `description`

    displays a description of the state.
  - `image`

    displays if the state has an image.
  - `capabilities`

    displays the capabilities, if any, of the state (CLI only).
  - `position`

    displays the order number of the state.
  - `references`

    displays a list of object references.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields displayed in the CLI. The default is " ".
- *state...*

  specifies names of the states to display.

See Also

- Commands: im createstate, im editstate, im viewstate, im deletestate
- Miscellaneous: options

im timeentries

displays time entries

Synopsis

```
im timeentries [--entryUser=user] [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--filter-value] [--
height=value] [--width=value] [-x=value] [-y=value] [--mode=[view|edit|viewany|editany]] [--query=[user:]query] [--
queryDefinition=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [--hostname=value] [--port=value] [(-F value|--
selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [--forceConfirm=[yes|no]]
```

Description

The `im timeentries` command allows you to view all time entries created by you or another user. Time entries are not private and can be viewed by any user; however, only the user who created a time entry can edit or delete it. To create, edit, or delete time entries, see the *PTC Integrity User Guide*.

For example,

```
im timeentries --entryUser=jriley --fields=createdDate,duration,issue
```

displays the creation date, amount of time, and the item the time was recorded against for all time entries entered by `jriley`.

Note
Displayed time entries vary based on the time zone that a user is operating in.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--entryUser=`*`user`*

  specifies the user whose time entries you want to view or edit.

  Note
  You can specify only one of the following options in a command: `--entryUser`, `--filter`, `--query`, or `--queryDefinition`.

- `--fields=`*`field1[:width1],field2[:width2]...`*

  specifies the time entry fields to display and the width of each field in characters. If the output is directed to the GUI, the width is specified in pixels.

  The time entry fields you can specify are:

  - `createdBy`

    displays the name of the user who created the time entry.
  - `createdDate`

    displays the date the time entry was created.
  - `duration`

    displays the duration of the time entry. All time entries are represented in hours and formatted to two decimal places.
  - `entryDate`

    displays the date the time entry was recorded to.
  - `issue`

    displays the ID of the issue that the time entry was recorded to.
  - `modifiedBy`

    displays the name of the user who last modified the time entry.
  - `modifiedDate`

    displays the date that the time entry was last modified on.
  - `notes`

    displays notes added to the time entry. Notes are optional.
  - `sourceDisplayName`

    displays the display name of the source that entered the time entry, for example, `Integrity`.
  - `sourceName`

    displays the name of the source that entered the time entry, for example, `mks_integrity`.
  - `user`

    displays the name of the user whose time entries you are viewing.

- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between the fields in the display.

- `--filter=`*`value`*

  specifies a filter to display time entries by.

  Note
  You can specify only one of the following options in a command: `--entryUser`, `--filter`, `--query`, or `--queryDefinition`.

  The filters you can specify are:

  - `issue:`*`expression`*

    displays time entries in a specific issue ID or range of issue IDs. If you are specifying a range of issue IDs, use conditions and logical operators, for example,

    ```
    issue: > 1000 AND 2000.
    ```
  - `user:`*`name`*

    displays time entries belonging to one or more users, where *`name`* is the user's name. Use spaces to specify more than one user.
  - `entrydate:`*`date`*

    displays time entries recorded on a specific date or during a time period, where `date` is the specified date or time period. If you are specifying a time period, use conditions and logical operators, for example, `entrydate: between 11/01/2004 AND 11/01/2005`.
  - `source:`*`name`*

    displays time entries created by a specific source, where `name` is the source name.

- `creationdate:`*`date`*

  displays time entries created on a specific date or during a time period, where `date` is the specified date or time period. If you are specifying a time period, use conditions and logical operators, for example, `creationdate: between 11/01/2004 AND 11/01/2005`.
- `createdby:`*`name`*displays time entries created by one or more users, where `name` is the user's name. Use spaces to specify more than one user.
- `modificationdate:`*`date`*

  displays time entries modified on a specific date or during a time period, where `date` is the specified date or time period. If you are specifying a time period, use conditions and logical operators, for example, `modificationdate: between 11/01/2004 AND 11/01/2005`.
- `modifiedby:`*`name`*

  displays time entries modified by one or more users, where `name` is the specified user name. Use spaces to specify more than one user.

- `--mode=[view|edit|viewany|editany]`

  specifies whether to display or edit personal time entries (`--mode=view|edit`) or time entries belonging to another user (`--mode=viewany|editany`). The `--mode=edit` option can only be specified with the `-g` or `--gui` option.

Note
You can also edit a time entry when you edit an issue assigned to you (in the GUI or Web interface only).

- `--query=`*`[user:]query`*

  specifies the query used to filter time entries by issue. Includes the name of the user who the query belongs to and the query name, for example, `mchang:ActiveDefects`. You do not have to specify the user name, but you must specify the query name. This option is useful when multiple users have the same name for a query.

Note

- Integrity initially assumes that text before the first colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `mchang:ActiveDefects`, Integrity searches for the `Active Defects` query created by `mchang`. If Integrity cannot find the query and/or user, it searches for the `mchang:ActiveDefects` query created by any user.
- You can specify only one of the following options in a command: `--entryUser`, `--filter`, `--query`, or `--queryDefinition`.

- `--queryDefinition=`*`value`*

  specifies a string to define the query constraints for the query used to define the selection criteria for time entries. For details of the query format, see the `im createquery` reference page.

Note
You can specify only one of the following options in a command: `--entryUser`, `--filter`, `--query`, or `--queryDefinition`.

See Also

- Commands: im editissue
- Miscellaneous: options

im toggleinclude

allows you to change included subsegments to referenced subsegments within the Document view

Synopsis

`im toggleinclude [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)][(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=[none|gui|default]] `*`node id...`*

Description

`im toggleinclude`allows you to change included subsegments to referenced subsegments within the Document view.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

`node id...`

the ID of the included subsegment for which you want to change to a referenced subsegment in the Document view.

See Also

- Commands: im refmode, im insertsegment, im importcontent
- Miscellaneous: diagnostics, options, preferences

im triggers

displays a list of event triggers

Synopsis

`im triggers [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] trigger...`

Description

`im triggers` displays a list of event triggers for workflows and documents. By default, all triggers are selected to be displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the trigger. By default, only the name is shown.
  - `description`

    displays a description of the trigger.
  - `type`

    displays the trigger type.
  - `runAs`

    displays the name of the user a scheduled trigger runs as. All modifications appear as though they were made by the specified user (CLI only).
  - `frequency`

    displays the scheduled frequency of a trigger (CLI only).
  - `query`

    displays name of the query used for a scheduled trigger (CLI only).
  - `script`

    displays the file name for the scripts used for the trigger (CLI only).
  - `scriptParams`

    displays the list of script arguments (CLI only).
  - `scriptTiming`

    displays if the script should be run pre, post, or both pre and post on issue commit to the database (CLI only).
  - `assign`

    displays the list of field assignments (CLI only).
  - `position`

    displays the position of the trigger in the run order (CLI only).
  - `lastRunTime`

    displays date and time the trigger was last run (CLI only)
  - `rule`

    displays the rule associated with the trigger (CLI only).
  - `references`

    displays a list of object references.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields displayed in the CLI.
- `--height=value`

  used with the `-g` or `--gui` options, specifies the height in pixels of the window.
- `--width=value`

  used with the `-g` or `--gui` options, specifies the width in pixels of the window.
- `-x value`

  used with the `-g` or `--gui` options, specifies the x location in pixels of the window.
- `-y value`

  used with the `-g` or `--gui` options, specifies the y location in pixels of the window.
- *`trigger...`*

  specifies names of the triggers to display.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im echo
- Miscellaneous: options

im types

displays a list of issue types

Synopsis

`im types [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-F file|--selectionFile=file)] [--[no]asAdmin] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]] type...`

Description

`im types` displays a list of issue types. By default, all types are displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the type. By default, only the name is shown.
  - `id`

    displays the database ID of the type. This is for PTC - Integrity Support only.
  - `description`

    displays a description of the type.
  - `defaultReferenceMode`

    displays the default reference mode if a document was branched. `Reuse` points to the same shared item until the original node changes and a new shared item is created.`Share` points to the same shared item as the original branched node. Editability is not allowed on shared fields; this mode only allows observation of the changes made on the other item.

    displays the additional fields by type that, when edited, result in an update to the content revision.
  - `associatedType`

    displays the associated type for segments and nodes. If the document class is Segment, the associated type is a node. If it is node, the type is a shared item.
  - `image`

    displays whether or not there is an image for the type.
  - `fieldRelationships`

    displays the relationships between fields (CLI only).
  - `groupDocument`

    displays whether the segment item type can be used to group content.
  - `addLabel`

    displays whether or not users can add labels on the type.
  - `allowAttachments`

    displays whether or not to allow attachments for the type.
  - `allowChangePackages`

    displays whether or not the type allows change packages.
  - `backsProject`

    displays that this type backs a project.
  - `branch`

    displays whether branches exist for the type.
  - `branchEnabled`

    displays whether branching is allowed on the type for specific users or groups.
  - `canRelateToTestResult`

    displays whether the item type can be related to test results.
  - `copyTree`

    displays whether items of this type have had their trees copied (CLI only).
  - `copyTreeEnabled`

    displays whether issue trees can be copied for this type.
  - `copyFields`

    displays the list of fields to be copied by default for items of this type. Users can override these default fields by adding and removing any fields that they can view and edit.
  - `labelEnabled`

    displays whether labelling is allowed on the type for specific users or groups.
  - `deleteLabel`

    displays whether labels can be deleted for this type by specific users or groups.
  - `issueEditability`

    displays the issue editability for the type.
  - `mandatoryFields`

    displays the mandatory fields for the type (CLI only).
  - `modifyTestResultPolicy`

    displays the users or groups allowed to modify test results for the test session item type.
  - `versionEditFields`

    displays the fields that are configured as editable for document versions.
  - `visibleFields`

    displays the fields that are visible for the type (CLI only).
  - `stateTransitions`

    displays state transitions for this type (CLI only).
  - `systemManagedFields`

    displays fields with system managed visibility.
  - `testCaseResultFields`

    displays the fields on the test case item type that display in the test results.

- `testRole`

  displays the test role for the type.
- `testStepsEnabled`

  displays whether test steps are enabled for this type.
- `typeClass`

  displays whether the associated types are a part of a document domain. Available type classes are segment, node, and shared item.
- `createCPPolicy`

  displays the change package creation policy (CLI only).
- `editPresentation`

  displays the presentation template name for editing an issue of this type (CLI only).
- `notificationFields`

  displays a list of the fields to send on notification (CLI only).
- `permittedAdministrators`

  displays the Type Administrators for this type (CLI only).
- `permittedGroups`

  displays the groups that are permitted access to issues of this type (CLI only).
- `phaseField`

  displays the names of the phase fields in the type's workflow.
- `position`

  displays the sequential position of the type in the list of types.
- `printPresentation`

  displays the presentation template for printing an issue of this type (CLI only).
- `showHistory`

  displays whether or not the type shows the **History** tab in the **Edit Issue** view and **Issue Detail** view.
- `showWorkflow`

  displays whether or not the type displays the **Workflow** tab in the **Create Issue** view, **Edit Issue** view, and **Issue Detail** view.
- `timeTrackingEnabled`

  displays whether or not the type allows time entries.
- `viewPresentation`

  displays the presentation template for viewing an issue of this type (CLI only).
- `references`

  displays a list of object references.
- `properties`

  displays properties defined for the type.

Note

Fields or states containing overrides display regardless of whether the fields are visible or the states are included in the workflow.

- `deleteItemEnabled`

  displays whether or not the type can be deleted by specific users or groups identified in the delete item rule.
- `deleteItem`

  displays the delete item rule for the type.
- `additionalContentLockFields`

  displays the list of additional content fields under the control of document locking.
- `additionalLockFieldsRule`

  displays the rule that determines when additional fields (if allowed) are affected by locking.
- `additionalSegmentLockFields`

  displays the list of additional segment fields under the control of document locking.
- `allowAdditionalLockFields`

  displays whether or not document locking affect additonal fields beyond the type's significant fields.
- `allowDocumentLocks`

  displays whether or not document locking is enabled for the type.
- `documentLockPolicy`

  displays the document locking policy for the type.
- `documentUnlockGroup`

  displays the lock administration group for the type.
- `lockingRequired`

  displays whether or not a document of this type must be locked before the fields affected by locking can be edited.
- `lockingRequiredRule`

  displays the rule that determines when a document of this type must be locked before the appropriate fields can be edited.

- `--[no]asAdmin`

  allows non-admin access to types. This option is used specifically for third party integrations using the Integrity API. The default setting is `--asAdmin` which requires `ViewAdmin` permissions. Specifying `--noasAdmin` prevents user access to types. If a user attempts to view a type but does not have the applicable permissions they will receive a corresponding error message. Specifying `im types --fields'', visibleFieldsForMe` for a type displays permitted fields by user.

  To learn more about the API, see your *PTC Integrity Integrations Builder Guide*.
- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimier between fields displayed in the CLI.
- *`type...`*

  specifies names of the types to display.

See Also

- Commands: im createtype, im edittype, im viewtype, im deletetype
- Miscellaneous: options

im unlock

unlocks a document

Synopsis

`im unlock [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] issue id...`

Description

`im unlock` unlocks a locked document. To unlock a document, you (or a group to which you belong) must have a lock on it or you must be part of the Lock Administrator group.

Normally, when a document is unlocked, any user can edit the lockable fields of that document. However, the administrator can configure Integrity to require users to lock a document before they edit the lockable fields of that document. When such a requirement is in place, no one can edit the lockable fields of an unlocked document.

Examples

For example, the current user is a member of the Reviewers groups. Consider the following documents:

- Document 100 is an Integrity document of a type that uses locking. It is locked by the current user. Locking is not required to edit this document.
- Document 200 is an Integrity document of a type that uses locking. It is locked by the Reviewers group. Locking is required to edit this document.
- Document 300 is an Integrity document of a type that uses locking. It is locked by the Publishers group. Locking is required to edit this document.

Specifying:

```
im unlock 100
```

attempts to unlock Document 100. This succeeds because it is locked by the current user. Any user can now edit the lockable fields of this document.

Specifying:

```
im unlock 200
```

attempts to unlock Document 200. This succeeds because the current user belongs to the group that has the document locked. No user can now edit the lockable fields of this document without locking it first.

Specifying:

```
im unlock 300
```

fails because the current user does not belong to the group which has the document locked.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- *`issue id...`*

  the ID of the document you are unlocking. To specify more than one document, use a space separated list.

See Also

- Commands: im lock
- Miscellaneous: options

im updateclient

updates the Integrity Client

Synopsis

```
im updateclient [--[no|confirm]download] [--[no|confirm]shutdown] [--[no|confirm]rollback] [--[no|confirm]rollbackshutdown] [--
hostname=server] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--
quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet]
```

Description

`im updateclient` updates the Integrity Client with a service pack if one is available. A service pack may be designated as required to address a known issue, or may provide enhancements. Client side service pack numbers are designated with a "C", for example, `C04030003`.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]download`

  automatically downloads a service pack if one is available.
- `--[no|confirm]shutdown`

  automatically shutdowns the client if a service pack is downloaded.
- `--[no|confirm]rollback`

  automatically initiates a service pack rollback, if required to connect to the Integrity Server.
- `--[no|confirm]rollbackshutdown`

  automatically shutdowns the client if a service pack rollback is initiated.

See Also

- Commands: im about, im connect
- Miscellaneous: options

im users

displays a list of users

Synopsis

`im users [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] user...`

DESCRIPTION

`im users` displays a list of users for workflows and documents. By default, all users are displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `name`

    displays the name of the user. By default, only the name is shown.
  - `id`

    displays the database ID of the user. This is for PTC - Integrity Support only.
  - `isActive`

    displays whether or not the user is active.
  - `description`

    displays a description of the user.
  - `image`

    displays whether or not there is an image, and if that image is default or custom.
  - `fullname`

    displays the full name of the user.
  - `email`

    displays the email address of the user.
  - `notificationRule`

    displays the email notification rule associated with the user (CLI only).
  - `isInRealm`

    displays whether or not the user exists in the authentication realm (CLI only).
  - `references`

    displays a list of object references.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields
- *user...*

  specifies names of the users to display.
-

See Also

- Commands: im createuser, im edituser, im viewuser, im deleteuser
- Miscellaneous: options

im viewadminlock

view the status of an administrative lock on the Integrity Server

Synopsis

```
im viewadminlock [--[no]batch] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [-g|--gui][--user=name]
[--hostname=server] [(-N|--no)] [--password=password] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]
```

Description

`im viewadminlock` allows you to view the status of the administrative lock, if any, on the target Integrity Server. Locking is a mechanism for preventing changes on the target server while you are testing a workflow and document configuration on a Staging Server. The Admin Migration Wizard allows you to migrate your final workflow and document configuration from the test environment on the Staging Server to the work environment on your Production Server.

An administrative lock means that the administrative configuration of the server is no longer editable directly through the standard command set. Locking ensures that no other administrative changes can occur to the Production Server while the migration wizard is performing its update.

The administrator can manually lock either the Staging or Production Servers, or both, to prevent administrative changes from taking place. Locking both servers means that no one can make any administrative changes to the workflow and document system. For more information, see the *PTC Integrity Server Administration Guide*.

For example:

```
im viewadminlock --server=abcFinancial --port=7001
```

displays the status of the administrative lock on the *abcFinancial* server.

Options

This command takes the universal options available to all `im` commands. See the options reference page for descriptions.

See Also

- Commands: im obtainadminlock, im releaseadminlock
- Miscellaneous: options

im viewauditlog

view a record of operations performed on the Integrity Server

Synopsis

`im viewauditlog [--fields=field1[:width1],field2[:width2]...] [--filter=value:<expression>] [--[no]batch] [--height=value] [--maxRows=value] [--width=value] [-x value] [-y value] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [-g|--gui][--user=name] [--hostname=server] [(-N|--no)] [--password=password] [--[no]persist] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]`

Description

`im viewauditlog` allows you to view a record of operations performed on the Integrity Server

Auditing categories are organized into independent hierarchies, with an individual Integrity Server component at the root of each category (workflows and documents–`im`, configuration management–`si`, or the Integrity Server–`is`). These root component categories are independent of one another.

Once auditing is enabled on the Integrity Server, you can use the command line interface to view the available audit records according to filters set by you. To access the audit log, use the `im viewauditlog` command and select from the available filters to find the required audit records.

For example, the following command would return all records for administrative operations in Integrity:

```
im viewauditlog --filter=operation:im.admin
```

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...]`

  where `fieldn` can be any of:

  - `id`

    specifies the ID number of an individual audit log detail. This is the ID number of an individual record found in a previous search.
  - `parentid`

    specifies the ID of the audit detail that spawned the specified operation (that is, all operations that have been spawned by the audit detail with the specified ID number).
  - `user`

    specifies a user ID. This is the login ID of the user who performed the operation.
  - `state`

    specifies the status of the operation being audited. Valid states are `completed` or `failed`.
  - `category`

    specifies the root component for Integrity. For example, workflows and documents is `im`, configuration management is `si`, and the Integrity Server is `is`.
  - `operation`

    specifies the operation or command. This can also include the operation that spawned the audit or sub-operation.
  - `contexttype`

    applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project`, `si.buildProject`, and `si.variantProject`.
  - `context`

    applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project` in the format *<project id>*, `si.buildProject` in the format *<project id>(<revision ID>*), and `si.variantProject` in the format *<project id>(<variant name>)*.
  - `date`

    specifies the timestamp information for the target operation.
  - `detail`

    specifies the audit log detail.
  - `targettype`

    specifies the type of object the specified operation is acting on. For example, `si.member`, `im.state`, and `im.issue`.
  - `target`

    specifies the object that the specified operation acts on.
  - `parameters`

    specifies the parameters defined in the operation. Parameters take the form of `key=value` pairs in a comma-separated string. Parameter keys and values depend on the operation being recorded, for example, `label=Version2,saveTimestamp=true`. For configuration management, arguments should match those created for the corresponding trigger scripts.
  - `resulttype`

    specifies the type of operation result. Valid result types include `exception` in the format *<exception class>*;, `im.issue` in the format *<issue ID>*, `si.revision` in the format *<revision ID>*.
  - `result`

    specifies the operation result. Valid results include `exception`, `im.issue`, and `si.revision`.
- `--filter[=value:<expression>]`

  specifies filter to apply when viewing the audit log results. Valid filters include the following:

  - `id:<expression>`

    specifies the ID number of an individual audit log detail. This allows you to search for an individual record found in a previous search.
  - `parentid:<expression>`

    specifies the ID of the audit detail that spawned the specified operation (that is, all operations that have been spawned by the audit detail with the specified ID number).
  - `user:<expression>`

    specifies a user ID to search for when filtering the audit log. This is the login ID of the user who performed the operation.
  - `state:<expression>`

    specifies the status of the operation being audited. Valid states are `completed` or `failed`.
  - `timestamp:<expression>`

    specifies the timestamp information for the target operation. Valid filter options are `today¦tomorrow¦yesterday`, `last¦next<total number of days`, `between <date>` and *<date>*.
  - `category:<expression>`

    specifies the root component of Integrity you want to filter for. For example, workflows and documents is `im`, configuration management is `si`, and the Integrity Server is `is`. To refine the level of filtering, you can also specify a subcomponent. For example, to filter the audit log for workflow and document administrative operations use `im.admin` as the expression. To filter for member operations, use `si.member` as the expression.
  - `operation:<expression>`

    specifies the operation or command being filtered in audit log. This can also include the operation that spawned the audit or sub-operation.
  - `contexttype:<expression>`

applies for applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project`, `si.buildProject`, and `si.variantProject`.

- `context:<expression>`

  applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project` in the format *<project id>*, `si.buildProject` in the format *<project id>(<revision ID>)*, and `si.variantProject` in the format *<project id>(<variant name>)*.

- `targettype:<expression>`

  specifies the type of object the specified operation is acting on. For example, `si.member`, `im.state`, and `im.issue`.

- `target:<expression>`

  specifies the object that the specified operation acts on.

- `parameters:<expression>`

  specifies the parameters defined in the operation. Parameters take the form of `key=value` pairs in a comma-separated string. Parameter keys and values depend on the operation being recorded, for example, `label=Version2,saveTimestamp=true`. For configuration management, arguments should match those created for the corresponding trigger scripts.

- `resulttype:<expression>`

  specifies the type of operation result to filter for. Valid result types include `exception` in the format *<exception class>*;, `im.issue` in the format *<issue ID>*, `si.revision` in the format *<revision ID>*.

- `result:<expression>`

  specifies the operation result to filter for. Valid results include `exception`, `im.issue`, and `si.revision`.

- `--maxRows=value`

  specifies the maximum number of audit log view records to return from the server.

- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

See Also

- Commands: im purgeauditlog, si purgeauditlog, si viewauditlog
- Miscellaneous: options

im viewchart

displays the properties of an existing Integrity chart

Synopsis

```
im viewchart [--[no]showHistory] [--[no]showReferences] [--hostname=value] [--port=value] [--password=value] [--user=value] [--
height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=
[yes|no]] [username:]chart...
```

Description

`im viewchart` displays the properties of a selected Integrity chart.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the chart.
- `--[no]showReferences`

  specifies whether to display all system provided and user objects that reference the chart.
- `[user:]chart`

  specifies the name of the chart to view, and the user who created it. If you are viewing a chart you created, you do not have to specify the user name, but you must specify the chart name.

Note

Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a chart name. If Integrity fails to find a matching user name and chart name, it searches for a chart name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` chart created by `jhoyt`. If Integrity cannot find the chart and/or user, it searches for the `jhoyt:CosmosDefects` chart created by any user.

See Also

- Commands: im copychart, im createchart, im deletechart, im editchart,im charts,imrunchart
- Miscellaneous: options

im viewcolumnset

displays the properties of a column set

Synopsis

```
im viewcolumnset [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [(-g|--gui)] [--forceConfirm=[yes|no]] columnset...
```

Description

Note

Column sets are no longer supported for the Integrity client. This command can only be used to view the properties of a column set for the MKS Worktray used in integrations. For more information on integrations, see the *PTC Integrity Integrations User Guide*.

`im viewcolumnset` displays the properties of a column set. The properties of a column set are: issue fields, name of the column set, what order the specified field is set to (ascending or descending order), and the field that issues are sorted by.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- *columnset...*

  specifies the column set whose properties you want to view. You can only specify one column set.

See Also

- Commands: im copycolumnset, im createcolumnset, im editcolumnset, im deletecolumnset, im columnsets
- Miscellaneous: options

im viewcp

displays the details of a change package

Synopsis

```
im viewcp[ --attributes=attribute1,attribute2...] [--entryAttributes=attribute1,attribute2...] [--[no]showEntries] [--
filter=type:name] [--height=value] [--width=value] [(-g|--gui)] [-x value] [-y value] [--[no]batch] [--hostname=value] [--port=value]
[--password=value] [--user=value] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--cwd=value] [--
forceConfirm=[yes|no]] [--quiet] [--settingsUI=[gui|default]][--status=[none|gui|default]] issue|issue:change package id...
```

Description

`im viewcp` allows you to view attribute and change package entry details on any change package you select. You can select the change package using an issue ID or change package ID. The selected change package does not have to be assigned to you. You can list and display multiple change packages.

For example,

```
im viewcp --attributes=type,summary --entryAttributes=revision,project 123
```

displays the type and summary for the change packages, and the revision and project for the change package entries, for issue 123.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--attributes=`*`attribute1,attribute2...`*

  specifies the change package attributes to display.
- `--entryAttributes=`*`attribute1,attribute2...`*
- specifies the change package entry attributes to display.

Note

The attribute `configpath` displays the configuration path of the change package entry (repository location of the member/subproject).

The following change package entry types display more than one configuration path:

- `Move Subproject`
  - `from entry`is the original configuration path of the subproject
  - `to entry` is the new destination configuration path
- `Configure Subproject`
  - `from` and `to` entries contain same configuration path
- `Move Member`
  - `from` entry is the original configuration path of the member
  - `to` entry is the new destination configuration path
- `Rename Member`
  - `from` and to entries contain same configuration path

- `--[no]showEntries`

  specifies whether to show the individual change package entries.
- `--filter=type:`*`name`*

  specifies a change package type to refine the change package selection. Acceptable values are `si` (configuration management change package), `implementer` (Implementer change package), or a custom change package type.
- *`issue...`*
- *`issue:change package id...`*
- issue identifies a specific issue that contains all change packages that you want to view; use spaces to specify more than one issue.

  `issue:change package id` identifies a specific change package to view; use a space separated list to specify more than one change package.

See Also

- Miscellaneous: ACL, options, preferences

im viewdashboard

displays the properties of an existing Integrity dashboard

Synopsis

```
im viewdashboard [--[no]showHistory] [--[no]showReferences] [--hostname=value] [--[no]showLayout] [--port=value] [--password=value]
[--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=
[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--height=value] [--width=value]
[-x value] [-y value] [user:]dashboard
```

Description

`im viewdashboard` displays the properties of an Integrity dashboard.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the dashboard.
- `--[no]showReferences`

  specifies whether to display all system provided and user objects that reference the dashboard.
- `--[no]showLayout`

  specifies whether to show the XML dashboard layout definition.
- *`[username:]dashboard`*

  specifies the name of the dashboard to view, and the user who created it. If you are viewing a dashboard you created, you do not have to specify the user name, but you must specify the dashboard name.

Note

Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a dashboard name. If Integrity fails to find a matching user name and dashboard name, it searches for a dashboard name matching the exact text. For example, if you type `jhoyt:ProjectOverview`, Integrity searches for the dashboard named `ProjectOverview` created by `jhoyt`. If Integrity cannot find the dashboard and/or user, it searches for the dashboard named `jhoyt:ProjectOverview` created by any user.

See Also

- Commands: im copydashboard, im createdashboard, im deletedashboard, im editdashboard, im dashboards, im rundashboard
- Miscellaneous: options

im viewduplicates

displays a list of potential duplicate items

Synopsis

```
im viewduplicates [--issueID=value] [--searchText=value] [--searchType=type] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

`im viewduplicates` runs a search to display a list of potential duplicate items. You can use duplicate detection to find possible duplicate items before submitting a new item, or to identify and clean up existing duplicates in the Integrity database.

For example, specify the search text and item type to search:

```
im viewduplicates --searchType=Defect --searchText="engine problems"
```

to display a list of potential duplicate defects containing the text string "engine problems".

Or run the search according to a single item ID number:

```
im viewduplicates --issueID=314159
```

to display a list of items that contain similar search text to the text contained in Item 314159. All potential duplicates are of the same item type as Item 314159.

Note the following:

- To identify existing duplicates, the Integrity administrator must first enable duplicate detection for the item type you want to search. If duplicate detection is not enabled for the type, an error occurs when running the `im viewduplicates` command. For more information, contact your Integrity administrator.
- Duplicate item searches are limited to items of the same Integrity type.
- Only the text you enter in the defined search field is used as the basis for the search. The search field is defined by your Integrity administrator (for example, the Summary field could be used).
- When searching for duplicates, Integrity returns a maximum of 20 items by default.
- If your administrator makes duplicate detection mandatory for the item type you are creating, the search runs automatically when you are finished entering text in the defined search field (when you move focus away from the defined search field). If duplicates are found, the list is displayed in the command output.
- When you use the `-g` or `--gui` option, Integrity displays the search results in the Potential Duplicates View.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--searchType=`*type*

  specifies the name of the item type you want to search for potential duplicates. Integrity uses the value you set here to obtain the search settings from the type (such as the defined search field and any mandatory settings). When searching for potential duplicates, Integrity will only search items that are of the type you specify here. This option must be specified together with the `--searchText` option.
- `--searchText="`*value*`"`

  specifies the specific text string you want to search for. This option must be specified together with the `--searchType` option.
- `--issueID=`*value*searches for potential duplicates based on the configured settings for the individual item you specify here. Integrity uses the text in the defined search field of the specified item and searches for similar text in items of the same type. This option must be specified on its own, and cannot be specified with either the `--searchType` or `--searchText` options.

See Also

- Miscellaneous: options

im viewdynamicgroup

displays the properties of a dynamic group

Synopsis

```
im viewdynamicgroup [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--[no]showReferences] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--quiet] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] dynamic group...
```

Description

`im viewdynamicgroup` displays the properties of a dynamic group. The details of the dynamic group are not editable in view mode.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the dynamic group.

  The following information is displayed:
  - `Object Type` displays the type of object referencing the dynamic group. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the dynamic group.
    - `Creator` displays the user ID that was logged as creating the object reference.
- *`dynamic group...`*

  specifies the name of the dynamic group.

See Also

- Commands: im createdynamicgroup, im editdynamicgroup, im deletedynamicgroup, im dynamicgroups
- Miscellaneous: options

im viewfield

displays the attributes of a field

Synopsis

```
im viewfield [--overrideForType=type] [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--
[no]showReferences] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--
selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] field...
```

Description

`im viewfield` lists detailed information about specified fields for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--overrideForType=`*type*

  specifies that the command applies to overridden attributes for the specified type. The *value* is the name of the target type.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.

- `--[no]showReferences`

  used to display all objects that reference the field. The following information is displayed:

  `Object Type` displays the type of object referencing the user. Possible objects include:

  - Admin Change Package Type
  - Admin Chart
  - Admin Dynamic Group
  - Admin Project
  - Admin Query
  - Admin Type (issue)
  - Chart (user charts)
  - Column Set
  - Field (includes computed fields, as well as editability, relevance, and visibility rules)
  - Query (user queries)
  - Trigger (includes trigger and notification rules)
  - User Notification Rule
  - User name
  - `Name` displays the actual name of the object referencing the field.
  - `Creator` displays the user ID that was logged as creating the object reference.

- *field...*

  specifies the names of the fields you want to view.

See Also

- Commands: im createfield, im editfield, im fields
- Miscellaneous: options

im viewgroup

displays the attributes of a group

Synopsis

```
im viewgroup [--height=value] [--width=value] [-x value] [-y value] [--quiet] [--[no]showHistory] [--[no]showReferences] [--
user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-
Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] group...
```

Description

`im viewgroup` lists detailed information about specified groups for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the group.

  The following information is displayed:
  - `Object Type` displays the type of object referencing the group. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the group.
    - `Creator` displays the user ID that was logged as creating the object reference.
- *`group...`*

  specifies the names groups you want to view.

See Also

- Commands: im creategroup, im editgroup, im deletegroup, im groups
- Miscellaneous: options

im viewissue

displays the issue information of one or more Integrity issues

Synopsis

```
im viewissue [--asOf=[<date>|label:<label>] [--[no]showAttachments] [--[no]showAttachmentDetails] [--[no]showAnnotations] [--[no]showBranches] [--[no]showLabels] [--[no]showLock] [--[no]showChangePackages] [--[no]showHistory] [--[no]showHistoryAscending] [--[no]showHistoryWithIndirectEdits] [--[no]showRelationships] [--[no]showWorkflow] [--[no]showTimeEntries] [--[no]showHistoryWithComputedField] [--[no]showRichContent] [--[no]showXHTML] [--[no]showTestResults] [--[no]substituteParams] [--[no]showSourceTraceDetails] [--[no]showSourceLinkDetails] [--[no]showDecorators] [--height=value] [--width=value] [-x value] [-y value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--[no]showIncomingExternalReferences] issue id...
```

Description

`im viewissue` displays the issue information of one or more Integrity issues.

For example,

```
im viewissue --showTimeEntries --showAnnotations --asOf="January 8, 2007 10:00:00 AM EST" 123
```

displays information for issue `123`, including time entries and annotations, as of January 8, 2007.

Note the following:

- Displayed date fields do not change based on the time zone that a user is operating in; however, displayed date/time fields vary based on the time zone that a user is operating in.
- Relevance and editability rules are evaluated on the Integrity Client's time zone.
- Computed expressions return dates/times in the Integrity Client's time zone and perform calculations in the Integrity Server's time zone where appropriate.

When you use the `-g` or `--gui` option, Integrity displays an issue selection dialog box. If you browse for issues from the issue selection dialog box, any changes you make to the columns in the Issues View will not be applied to that view when it is accessed through the Integrity Client GUI.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--asOf=<date>|label:<label>`

  allows you to view an issue as of a historical date or label. If a value is not provided the issue displays as of the server's current time. This field is optional.

  Note
  If you specify a `WEST` or `IST` time zone, the time is not displayed correctly in the issue history. Instead, use the time zone GMT+/-*hours*:*minutes*.

- `--[no]showAttachments`

  specifies whether to display attachment file names. The default is to show attachments.

  Note
  This option controls the visibility of the built in attachment field only. Custom attachment fields are always visible.

- `--[no]showAttachmentDetails`

  specifies whether to display the following attachment details: added user, added date, summary and Mime type, and name and size of attachment. The default is not to display attachment details.

- `--[no]showAnnotations`

  specifies whether to display annotations or a audit record for an issue. The default is not to show annotations.One line per annotation in the following format

  ```
  Annotations:
  Operation by: John Smith on Thu Aug 10 08:59:35 EDT 2006
  Add Label: mylabel against time Thu Aug 10 08:59:35 EDT 2006
  ```

- `--[no]showBranches`

  specifies whether to show branch issue numbers if branches exist. A separate line indicates if an issue was branched from a parent. If the branch was performed against an `--asOf` time the annotation indicates the target `--asOf` time. The default is not to show branches.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the issue. The default is to not show the history. If you display the issue history, the information displays in chronological order (the most recent changes appear at the bottom) by default; however, you can configure the chronological ordering of history information using the `--[no]showHistoryAscending` option.

  Note
  Depending on the settings applied by your Integrity administrator, the item's history may not be visible. If `showHistory` is disabled, an error message displays when you use the `--showHistory` option.

- `--[no]showHistoryAscending`

  specifies whether to display the issue history in ascending or descending chronological order. In the GUI, the issue history displays in descending order by default. In the CLI and API, the issue history displays in ascending order by default. This option requires the `--[no]showHistory` option.

- `--[no]showHistoryWithIndirectEdits`

  specifies whether to include indirect edits in the item's history. An indirect edit is an edit made to the backing field of a field value attribute (FVA) field. When `--showHistoryWithIndirectEdits` is specified, the results of such edits are shown in the item's history. When this option is not specified or `--noshowHistoryWithIndirectEdits` is specified, the results of such edits are not included.

- `--[no]showLabels`specifies whether to display the labels created on an issue against a specific date and time. The default is not to show labels.

- `--[no]showLock`

  specifies whether to display the document locking information about an issue. The default is not to show locking information.

- `--[no]showChangePackages`

  specifies whether to display change package information. For more information on using change packages, refer to the *PTC Integrity User Guide*. The default is to show change packages.

- `--[no]showRelationships`

  specifies whether to display related issue IDs from the Forward and Backward relationship fields. The default is to show these fields.

  Note
  This option controls the visibility of the related issue IDs from the built in relationship fields only. Related issue IDs from custom relationship fields are

always visible.

- `--[no]showWorkflow`

  specifies whether to display the workflow for the issue type, if your administrator has enabled it. This option can only be specified with `-g` or `--gui`. Viewing the `Workflow` panel is useful for determining where you can progress in the workflow. The `Workflow` panel displays the complete workflow for the issue type, unvisited states, visited states, the current state, other state transitions, and phases, as indicated by the Legend.

- `--[no]showTimeEntries`

  specifies whether to display time entries for the issue type, if enabled by your administrator. Time entries indicate time spent working on an issue. Time entries display the entry date, user, source, duration, and notes. Time entries are sorted in descending order of entry date.

- `--[no]showHistoryWithComputedField`

  specifies whether to display the changes to computed fields in the issue history. By default, computed fields are not displayed in the issue history.

- `--[no]showRichContent`

  specifies whether to display rich text information. The data displays in raw rich text, with HTML elements and attributes. You can display rich text for the issue as of now, for the issue as of a historical date or label (using the `--asOf option`), or for the log of changes made to the item (using the `--showHistory option`).

Note
This option is intended only for use with the CLI or API and does not work when used with the`-g` option. By default, the data displays in plain text.

- `--[no]showXHTML`

  specifies whether to display rich text field data in XHMTL format. Use with the `--[no]showRichContent` option. The `--[no]showXHTML` option is intended for use in triggers, scripts, or custom integrations. Special characters are included in the CLI output, but not the API output (API output is entity protected, for example < is converted to <).

- `--[no]showTestResults`

  specifies whether to display test results for the issue, if enabled by your administrator for the issue type. The data displays as a list of test result IDs in the following format: `test session ID : test case ID`.

- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.

- `--[no]showSourceTraceDetails`

  specifies whether to display source trace details for any source link fields, with trace enabled, on the issue. By default, source link fields with trace enabled display the file name, revision and line range (start and end) for each trace. If details are displayed, the following information is displayed for each trace: member, revision, line range, trace name, server, project, variant (developent path), created date, modified date, suspect trace (true or false), and follow tip (true or false). Details are separated by a tab.

  Follow tip indicates that the trace always points to the head revision of the source file.

- `--[no]showSourceLinkDetails`

  specifies whether to display source link details for any source link fields on the issue. By default, source link fields display the file name, revision and line range (start and end) for each link.

- `--[no]showDecorators`

  specifies whether to display the `!` decorator to indicate a computed field in a versioned item contains an ambiguous computation. By default, the `--showDecorators` option is specified.

- `--[no]showIncomingExternalReferences`

  specifies whether to show incoming external references (IERs) that are related to a requirement. This option is only valid if one or more IER fields have been created.

- *issue id...*

  specifies the ID of the issue you want to view. Use a space separated list to specify more than one issue ID, for example, `240 241 242`.

  If document versioning is enabled, you can also specify versioned items. To type the ID of a versioned item, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

See Also

- Commands: im copyissue, im createcolumnset, im createissue, im editcolumnset, im editissue, im extractattachments
- Miscellaneous: options

im viewpendingimports

displays the view in which you can see all pending imports that await your review

Synopsis

```
im viewpendingimport [--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...] [--hostname=value] [--port=value] [--
password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]]
[--status=[none|gui|gui.actions|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

`im viewpendingimports` displays the view in which you can see all pending imports that await your review.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

For more information about the `--fields` option, see the im issues man page.

See Also

- Commands: im issues
- Miscellaneous: options

im viewprefs

displays preferences

Synopsis

`im viewprefs [--[no]global] [--command=`*value*`] [--[no]showValidValues] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [(-F `*value*`|--selectionFile=`*value*`)] [--[no]batch] [--cwd=`*value*`] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`im viewprefs` displays preferences and configuration options for Integrity. These settings are used to determine default behaviors for other commands. You can only view one set of preferences at a time.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the `options` reference page for descriptions. For an easy way to see a list of commands and values that may be set, type the `im viewprefs` command, either piped through `|more` or redirected to a file, for example:

```
im viewprefs --global --showValidValues >prefs.txt
```

Alternatively, the `--gui` option presents a dialog box that lets you view and configure the preferences.

- `--[no]global`

  specifies whether to show all preferences.
- `--command=`*value*

  identifies the command that preferences are to be viewed for.
- `--[no]showValidValues`

  specifies whether to display a list of valid values for the preferences.
- `--[no]ask`

  specifies whether to show the ask preference. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:
  - `once`

    asks the user the first time only, and then uses the provided value every time after.
  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).
  - `element-last`

    asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.
- `--ui=[unspecified|gui|cli|api`

  controls whether to view the preference for the graphical user interface, the command line interface, or an unspecified interface. By default, `--ui=cli` is implied when using `im viewprefs`. To view preferences for GUI behavior, however, you should specify `--ui=gui`. For example, to view the preference for the im printissue command, type:

  ```
  im viewprefs --command=printissue --ui=gui
  ```

  These correlate to settings in the`IntegrityClient.rc` file that have the `gui.im.` or `cli.im.` prefix, or the `im.` prefix when it is unspecified.

See Also

- Commands: im loadrc, si setprefs
- Miscellaneous: ACL, options

im viewproject

displays detailed project information

Synopsis

`im viewproject [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--[no]showReferences] [--user=name] [--hostname=server] [--password=password] [--port=number] [--quiet] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] project...`

Description

`im viewproject` lists detailed information about specified projects for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the project. The following information is displayed:
  - `Object Type` displays the type of object referencing the project. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the project.
    - `Creator` displays the user ID that was logged as creating the object reference.
- *`project...`*

  specifies the projects you want to view. Include a forward slash (/) when specifying a high level project. For subprojects, include the full path to the subprojects, for example, `/AuroraProject/BorealisProject`.

See Also

- Commands: im createproject, im editproject, im deleteproject, im projects
- Miscellaneous: options

im viewquery

displays the properties of an existing Integrity query

Synopsis

`im viewquery [--[no]showHistory] [--[no]showReferences] [--hostname=`*value*`] [--port=`*value*`] [--password=`*value*`] [--user=`*value*`] [--height=`*value*`] [--width=`*value*`] [-x `*value*`] [-y `*value*`] [(-?|--usage)] [(-g|--gui)] [(-F `*value*`|--selectionFile=`*value*`)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*value*`] [--forceConfirm=[yes|no]] `*[user:]query...*

Description

`im viewquery` displays the properties of an Integrity query. The properties of an Integrity query are: name, who created the query, description, image type, users and groups shared with, query definition, associated column set, and last modified date.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the query.
- `--[no]showReferences`

  specifies whether to display all system provided and user objects that reference the query.
- *`[user:]query...`*

  specifies the name of the user who the query belongs to and the query name, for example, `jhoyt:Cosmos Critical Defects`. If you want to view queries that have the same name, but different users, you must specify `[user:]query`. The query name is mandatory.

Note

Integrity initially assumes that text before the colon (:) is a user name and text after it is a query name. If Integrity fails to find a matching user name and query name, it searches for a query name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` query created by `jhoyt`. If Integrity cannot find the query and/or user, it searches for the `jhoyt:CosmosDefects` query created by any user.

See Also

- Commands: im copyquery, im createquery, im deletequery, im editquery, im queries
- Miscellaneous: options

im viewreport

displays report information

Synopsis

```
im viewreport [--[no]showHistory] [--[no]showReferences] [--[no]showRecipeParams] [--hostname=value] [--port=value] [--
password=value] [--user=value] [(-?|--usage)] [--height=value] [--width=value] [-x value] [-y value] [(-F value|--
selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [(-g|--gui)] [--forceConfirm=[yes|no]] [user:]report
```

Description

`im viewreport` displays the properties of an Integrity report. Using the command line interface, you can select more than one report to view at a time.

Options

This command takes the universal options available to `im` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the report.
- `--[no]showReferences`

  specifies whether to display all system provided and user objects that reference the report.
- `--[no]showRecipeParams`

  specifies whether to display the report recipe parameters.
- *`[user:]report`*

  specifies the name of the report and the user who created the report. If you are viewing a report you created, you do not have to specify the user name, but you must specify the report name.

Note
Integrity initially assumes that text before the colon (`:`) is a user name and text after it is a report name. If Integrity fails to find a matching user name and report name, it searches for a report name matching the exact text. For example, if you type `jhoyt:CosmosDefects`, Integrity searches for the `CosmosDefects` report created by `jhoyt`. If Integrity cannot find the report and/or user, it searches for the report created by any user.

See Also

- Commands: im createreport, im editreport, im copyreport, im runreport, im reports
- Miscellaneous: options

im viewsegment

displays a segment in the Document view

Synopsis

```
im viewsegment [--[no]applyDisplayPattern] [--asOf=[<date>|label:<label>] [--[no]displayOutline] [--
fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...] [--filterRule=value] [--filterRuleFile=value] [--
filterQueryDefinition=value] [--filterQueryDefinitionFile=value] [--[no]linkWithTable] [--perspective] [--[no]recurseInclude] [--
[no]recurseReference] [--sessionID=value] [--[no]showParentage] [--[no]displayOutline] [--[no]showTallRows] [--[no]inlineEditMode] [-
-currentContext=value] [--focusIssueID=value] [--expandLevel=value] [--
outlineColumns=field[:width[:rich|plain]],field[:width[:rich|plain]],...] [--structureFieldIconDisplayField=field] ][--
structureFieldDisplayFormat=value] [--outlineItemFormat=value] [--outlineIconField=field] [--outlineItemFormat=value] [--
[no]substituteParams] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--
selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [--forceConfirm=[yes|no]] [--height=value] [--width=value] [-x value] [-y value] issue id... [--[no]showEditPanel] [--
[no]confirmSave] [--[no]showIssueView] [--[no]multiEditSession]
```

Description

`im viewsegment` displays a segment in the Document view. A segment can be a document root or a subsegment. If you specify the item ID of the reference to a subsegment, the Document View displays the parent document that contains the subsegment.

This command is only supported with the `-g` or `--gui` option. When you use the `-g` or `--gui` option, Integrity displays an issue selection dialog box. If you browse for issues from the issue selection dialog box, any changes you make to the columns in the Issues View will not be applied to that view when it is accessed through the Integrity Client GUI.

Note

Certain command options cannot be used with `-g` or `--gui`. These include `--filterQueryDefinition`, and `--filterQueryDefinitionFile`. These options are primarily useful for building integrations through the API.

Options

When calling the `im viewsegment` command using the API, you can use `--filterRule` or `--filterQueryDefinition` when you need to filter the content on the segment to be returned. For example:

- To provide a supplier with only the `Technical Requirement` content from a document, use:

  ```
  --filterQueryDefinition=(field[Category]="Technical Requirement")
  ```

- To retrieve all document content nodes that were changed in the last two days, use:

  ```
  --filterQueryDefinition=(field[Modified Date] in the last 2 days)
  ```

- To retrieve only `Comment` nodes where the most recent change was made by the user "`Me`", use:

  ```
  --filterRule=((field[Category]="Comment") and field[Modified By]="me"))
  ```

When working with `--filterRule` and `--filterQueryDefinition`, note the following key considerations:

- The `--filterRule` and `--filterQueryDefinition` options have different rules for filtering content (see detailed descriptions of the two options below).
- The `--filterQueryDefinition` option always returns the top-most segment (that is, the segment that was passed as an argument to this command). The `--filterRule` option only returns the segment if it matches the specified filter.
- The `im viewsegment` command may perform better on larger documents when using `--filterQueryDefinition`, because only the partial or filtered document is retrieved from the repository. When using the `--filterRule` option, the complete document is retrieved from the repository and is then filtered.

Note

: Prior to Integrity 10.3, the `--filterRule` option was known as `--filter`.

For larger filter definitions, you can use the `--filterRuleFile` or `--filterQueryDefinitionFile` options. For further information on the available options, see the detailed descriptions that follow.

This command also takes the universal options available to `im` commands, as well as some general options. See the options reference page for descriptions.

- `--filterRule=`*value*

  specifies a rule to filter the content when viewing the segment from the API.

  A *<rule>* rule is defined as an *<expression>*

  An *<expression>* is defined as one of the following:(*<expression>* and *<expression>* and *<expression>*)(*<expr>* or *<expr>* or *<expr>*)(*<user op>*)(*<field>* *<operator>* *<value>*)

  where for users and groups:

  <user op> rule is defined as a user is not a member of "*<group>*"*<user op>* rule is defined as a user is a member of "*<group>*"*<value>* rule is defined as a *<field>* | "*<text>*" | "*<number>*"*<field>* rule is defined as a field[*<fieldname>*]*<fieldname>* rule is defined as a ID | Summary | Priority | ...*<group>* rule is defined as a (everyone | im-dev | ...)*<operator>* rule is defined as a (= | > | >= | <= | < | <>)*<number>* rule is defined as a (.. | -1 | 0 | 1 | ..)

  For example:

  ```
  ((field[Occurrences] > 4) and (field[Created By]="ProjectManager"))
  ```

  When using `--filterRule` with the `-g` or `--gui` option, note the following:

  - Field filters:
    - For single expression with single field filter, the *<expression>* is converted into a field filter. For example, `(field[Input Revision Date] in the last 30 days)`.
    - For multiple expressions with and operations the *(<expression> and <expression> and <expression>)* is converted into multiple field filters. For example, `((field[Input Revision Date] in the last 30 days) and (field[Category]="Heading"))`

      Note

      `OR` operation is not supported.

      This rule does not work for multiple criteria on same field. For example,`(field [Input Revision Date]in the last 30 days) and ,(field [Input Revision Date]> in the last 60 days)`
  - Advanced filters:

    To convert an expression into advanced filter, user needs to specify extra brackets.

    For example: `(((field[Input Revision Date]<'in the last 30 days) or (field[Category]="Heading")))`
- `--filterRuleFile=`*value*

  specifies the name of the file that filters the content of the segment you are viewing.
- `--filterQueryDefinition=`*value*

specifies a query definition rule to filter the content when viewing the segment from the API. The filtering applies to all segment nodes, as well as to all included subsegment nodes. Filtering does not apply to the segment (that is, the segment is always returned in the API result whether or not it matches the filter query definition). This option is not valid from the CLI or GUI.

The filter query definition uses the same format as the `--queryDefinition` option for the `im createquery` command. See the im createquery reference page for details.

Note

To reference the original query in the new query so that any changes to the original query are reflected in the new query, define *<subquery>* as *subquery[OriginalQuery]*.

- `--filterQueryDefinitionFile=`*value*

  specifies the name of the file where the query definition is stored. You can use this option for larger filter query definitions. This option is not valid from the CLI or GUI.
- `--perspective`

  specifies the perspective name to be applied.

Note

`--perspective` works independently. This option cannot be used with `--filterQueryDefinition`, `--filterRule`, `--filterRuleFile`, `--filterQueryDefinitionFile`, and `--fields`.

- `--[no]recurseInclude`

  specifies whether or not to display included subsegments for the segment in the tree pane. When a subsegment is included into the parent, the entire contents of the subsegment are exposed as if they were a sequential part of the parent.
- `--[no]recurseReference`

  specifies to display referenced segments in the tree pane. When a subsegment is inserted into a parent segment, only the reference to the subsegment is exposed. You must open a subsegment to manage its contents.
- `--sessionID=`*value*

  specifies the item ID of a test session. Including a test session ID provides a test session as context when viewing a document and counts as one of the layers in parameter substitution.
- `--[no]showParentage`

  shows the parentage of the content in a filtered document. This option is only useful if a filter is enabled. By default, parentage is not shown. When enabled, all ancestors in the exported document are shown, from filtered items to the document root.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `--[no]displayOutline`

  specifies to display the Outline, a tree-like relationship hierarchy stemming from the segment you used to launch the view.
- `--[no]linkWithTable`

  specifies that when a segment or node is selected in the Outline pane in the Document view, the same portion of the segment or node is highlighted in the tree pane.
- `--outlineColumns=`*field[:width[rich|plain]],field[:width[rich|plain]],...*specifies the columns to display in the Outline.
- `--[no]applyDisplayPattern`

  specifies whether to apply a display pattern to numeric fields. Display patterns are configured by your administrator and allow you to quantify integer and floating point field values, for example, as currency or percentages. `--applyDisplayPattern` is enabled by default.

Note

If you use scripts, PTC recommends using the `--noapplyDisplayPattern` option to avoid being impacted by administrative changes to display patterns.

- `--asOf=`*[<date>|label:<label>]*specifies the date to use for the segment. To specify a date and time, type `MM/dd/yyyy h:mm:ss [AM|PM]`. To specify the current date, type `today`. To specify the current date and time, type `now`. All segment content included in the view is as of the date and time specified.
- `--fields=`*field[:width[:rich|plain]],field[:width[:rich|plain]],...]*specifies the fields, and their respective widths, to display in the Document view. Your administrator defines the available fields. Fields can include `ID`, `Type`, `Assigned User`, `Assigned Group`, `Summary`, and others. Use commas to specify more than one field.
- `--focusIssueID=`*value*

  specifies the item ID to highlight in the tree pane when the segment displays. This colour is defined in the `View > Options` dialog in the context of the Document view.
- `--expandLevel=`*value*

  specifies to expand the nodes to a specified level, for example, 1, 2, 3. The default is one.
- `--outlineColumns=`*field[:width[:rich|plain]],field[:width[:rich|plain]],...]*specifies the fields, and their respective widths, to display in the outline pane of the Document view. Your administrator defines the available fields. Fields can include `ID`, `Type`, `Assigned User`, `Assigned Group`, `Summary`, and others. Use commas to specify more than one field.
- `--currentContext=`*value*

  specifies the current task context in the Document view columns. If the value for `--currentContext` is not a valid context, it is ignored and the default task context will be used.
- `--[no]inlineEditMode`

  enables inline editing. You can edit issue fields from the Document view fields without requiring the `Item > Edit` command or its subsequent dialog box.
- `--[no]showTallRows`

  allows you to expand the height of a row to fit the data in the columns.
- `--structureFieldDisplayFormat=`*value*

  specifies the fields and style that should be displayed for the tree nodes.

  Defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  Uses the same values as `--fields`, but similar to a JAVA Message Format string (that is, it requires { } to surround each field). For example:

  ```
  im viewsegment --structureFieldDisplayFormat="{ID},{Summary}"
  ```
- `--structureFieldIconDisplayField=`*field*

  specifies the field from which the icon is taken. This is the icon that will be displayed for each node in the Outline pane.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- *issue id...*

specifies the ID of the issue of the segment you want to view. Use spaces to specify more than one issue, for example `34 23`.

If document versioning is enabled, you can also specify versioned documents. To type the ID of a versioned document, use the format *Live Item ID-major.minor*, for example, `184-1.2`.

Note

For versioned documents, referencing a content node ID is not supported for the`im viewsegment` command. When specifying a document version, the ID must refer to a segment or document. In addition, you can only view a versioned document in the Integrity Client GUI.

- `--[no]showEditPanel`

  specifies whether to show the embedded item edit pannel. This option can be used in conjunction with the `--[no]showIssueView` option. If conflicts arise, the`--[no]showEditPanel` option always wins.
- `--[no]confirmSave`

  specifies whether to prompt you to confirm changes are to be saved. This turns on and off whether you a confirmation window oepns when you save in this view.
- `--[no]showIssueView`

  specifies whether to show the embedded item view. This option can be used in conjunction with the `--[no]showEditPanel` option. If conflicts arise, the `--[no]showEditPanel` option always wins.
- `--[no]multiEditSession`

  specifies whether to use single-row editing or multiple-row editing in the Document view.

See Also

- Commands: im createsegment
- Miscellaneous: diagnostics, options, preferences

im viewserveralert

displays Integrity Server alert messages for a target server and all related servers

Synopsis

`im viewserveralert [--height=value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`im viewserveralert` displays Integrity Server server alert messages for a target sever and all related servers, for example, a configuration management server and a proxy server. The alert message displays who sent the message, the server it came from, when it was sent, and the message. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.
- A connection to the target server is required for the `im viewserveralert` command to display alert messages.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=server`

  specifies the host name of the target Integrity Server to retrieve alert messages from.
- `--port=number`

  specifies the port of the target Integrity Server to retrieve alert messages from.

See Also

- Commands: im serveralerts, im admingui, im servers
- Miscellaneous: options

im viewsourcetraces

displays trace information for one or more source files

Synopsis

```
im viewsourcetraces [--devpath=path] [--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...] [--fieldsDelim=value] [--
asOf=[<date>|label:<label>] [--height=value] [(-P project|--project=project)] [--projectRevision=value] [(-r rev|--revision=rev)] [--
scmHost=value] [--scmPort=value] [--sortAscending] [--sortField=field] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-F
value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--hostname=value] [--
port=value] [--password=value] [--user=value] [(-g|--gui)] source file...
```

Description

`im viewsourcetraces`
displays trace information for one or more source files.

For example,

```
im viewsourcetraces -g --devpath=SP1 -r 1.30.1.1 -p "/Release1/project.pj" dataStructure.txt
```

displays information for all traces for revision 1.30.1.1 of dataStructure.txt in the SP 1 variant of the Release 1 project.

Options

This command takes the universal options available to `im` and `si`commands, as well as some general options. See the options reference page for descriptions.

- `--fields=`*`field[:width[:rich|plain]],field[:width[:rich|plain]],...`*

  specifies the issue fields, and their respective widths, to be displayed. Your administrator defines the fields in an issue type. Use commas to specify more than one field. The default fields displayed in the CLI are independent of the default fields (columns) specified in the GUI view.
- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between the fields in the display.
- `--asOf=`*`<date>|label:<label>`*

  allows you to view source traces as of a specific date or label. This field is optional. If a value is not provided the source traces are viewed as of the current time.
- `--sortField=`*`field`*

  specifies the field to sort issues by, for example, `ID`.

  By default, issues are sorted by ID.
- `--sortAscending`

  specifies whether to sort the specified fields in ascending order.
- `--scmHost=`*`value`*

  the host name of the configuration management server where the traced source files are stored.
- `--scmPort=`*`value`*

  the port number of the configuration management server where the traced source files are stored.
- *`source file...`*

  identifies a specific source file; use spaces to specify more than one source file.

See Also

- Commands: im copyissue, im createissue, im viewissue, im editissue
- Miscellaneous: options

im viewstate

displays detailed state information

Synopsis

`im viewstate [--overrideForType=type] [--height=value] [--width=value] [-x value] [-y value] [--quiet] [--[no]showHistory] [--[no]showReferences] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] state...`

Description

`im viewstate` lists detailed information about specified states for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--overrideForType=type`

  specifies that the command applies to overridden attributes for the specified type. The *value* is the name of the target type.
- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the state. The following information is displayed:
  - `Object Type` displays the type of object referencing the state. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the state.
    - `Creator` displays the user ID that was logged as creating the object reference.
-
- *state...*

  specifies the names of the states you want to view.

See Also

- Commands: im createstate, im editstate, im deletestate, im states
- Miscellaneous: options

im viewtrigger

displays the attributes of an event trigger

Synopsis

`im viewtrigger [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--[no]showReferences] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] trigger...`

Description

`im viewtrigger` displays the attributes of an event trigger for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--height=value`

  used with the `-g` or `--gui` options, specifies the height of the GUI window, in pixels; *value* must be a whole number.
- `--[no]showHistory`

  used to display the history of changes for the selected system provided object in Integrity. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the trigger.

  The following information is displayed:

  - `Object Type` displays the type of object referencing the trigger. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the trigger.
    - `Creator` displays the user ID that was logged as creating the object reference.
- *trigger...*

  specifies the name of the event trigger you want to view.

See Also

- Commands: im createtrigger, im edittrigger, im runtrigger, im deletetrigger, im triggers, im echo
- Miscellaneous: options

im viewtype

displays the attributes of an issue type

Synopsis

```
im viewtype [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--[no]showReferences] [--[no]showProperties]
[--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-
N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] (--
status=[none|gui|default]] type...
```

Description

`im viewtype` lists detailed information about specified types.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the type. The following information is displayed:
  - `Object Type` displays the type of object referencing the type. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the type.
    - `Creator` displays the user ID that was logged as creating the object reference.

Note

Fields or states containing overrides display regardless of whether the fields are visible or the states are included in the workflow.

- `--[no]showProperties`

  specifies whether to show all properties defined for the type.
- *`type...`*

  specifies the name of the type you want to view.

See Also

- Commands: im createtype, im edittype, im deletetype, im types, im copytype.
- Miscellaneous: options

im viewuser

displays the attributes of a user

Synopsis

```
im viewuser [--height=value] [--width=value] [-xvalue] [-yvalue] [--[no]showHistory] [--[no]showReferences] [--quiet] [--user=name]
[--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [-
-[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] user...
```

Description

`im viewuser` lists detailed information about specified users for workflows and documents.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  used to display the history of changes for the selected system provided object in Integrity. By default, the history of changes is not shown when running the view command for the target object.
- `--[no]showReferences`

  used to display all objects that reference the user.

  The following information is displayed:

  - `Object Type` displays the type of object referencing the user. Possible objects include:
    - Admin Change Package Type
    - Admin Chart
    - Admin Dynamic Group
    - Admin Project
    - Admin Query
    - Admin Type (issue)
    - Chart (user charts)
    - Column Set
    - Field (includes computed fields, as well as editability, relevance, and visibility rules)
    - Query (user queries)
    - Trigger (includes trigger and notification rules)
    - User Notification Rule
    - User name
    - `Name` displays the actual name of the object referencing the user.
    - `Creator` displays the user ID that was logged as creating the object reference.
- *`user...`*

  specifies the users you want to view.

See Also

- Commands: im createuser, im edituser, im deleteuser, im users
- Miscellaneous: options

integrity about

displays product information

Synopsis

`integrity about [--[no]batch] [--cwd=`*`directory`*`] [(-g|--gui)] [ --quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)]`

Description

`integrity about` displays version information for the Integrity release, build, service pack (if installed), API, and HotFixes (if installed).

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity exit, integrity gui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: options

integrity acv

displays running commands

Synopsis

`integrity acv [--[no]batch] [--cwd=`*`directory`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)] [--height=`*`value`*`] [--width=`*`value`*`] [-x=`*`value`*`] [-y=`*`value`*`] [--forceConfirm=`*`[yes|no]`*`] [(-Y|--yes)] [(-N|--no)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [--[no]persist]`

Description

`integrity acv` displays the currently running commands.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]persist` controls the persistance of CLI views.

See Also

- Miscellaneous: diagnostics, options

integrity admin

opens the Integrity Administration window

Synopsis

`integrity admin [--height=`*value*`] [--width=`*value*`] [--user=`*name*`] [--hostname=`*server*`] [--password=`*password*`] [--port=`*number*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [-g|--gui] [(-F`*file*`|--selectionFile=`*file*`)] [--settingsUI=`*[gui|default]*`] [--quiet] [--status=`*[none|gui|default]*`]`

Description

`integrity admin` launches an Administration window instance. Only one Administration window instance can be opened using this command, and the window does not include the full functionality of the Integrity Administration Client.

Note

The Integrity Administration Client GUI interface essentially acts as a container for several administration applications (workflow and document management, configuration management). From the client interface, you can specify one or more servers to administer, allowing you to have multiple Administration windows open. In addition, you can specify application preferences. CLI commands that offer full administrative functionality include: `aa admingui`, `si admingui`, `im admingui`, and `integrity admingui`. To open the Administration client, see integrity admingui. For information on using the Administration window, see the PTC Integrity Server Administration Guide.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=`*server*

  identifies the name of the host server where the Integrity Server is located.
- `--port=`*number*

  identifies the port on the host server where the Integrity Server is located.
- `--password=`*password*

  identifies the password to use for connecting to the Integrity Server.
- `--user=`*name*

  identifies the user to use for connecting to the Integrity Server. This typically defaults to the name you have used to log into your client machine.

See Also

- Commands: integrity about, integrity disconnect, integrity exit, integrity admingui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: options

integrity admingui

launches the Integrity Administration Client

Synopsis

`integrity admingui [--height=value] [--[no]restoreDesktop] [-x value] [-y value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`integrity admingui` launches the Integrity Administration Client GUI. The client interface provides a single, centralized access point for the most common administration tasks. More specifically, you can manage Access Control Lists (ACLs), manage and distribute ViewSets, set up workflows and documents, configure configuration management policies, and send alert messages.

Note

The Integrity Administration Client GUI interface essentially acts as a container for several administration applications (workflow and document management, configuration management). From the client interface, you can specify one or more servers to administer, allowing you to have multiple Administration windows open. In addition, you can specify application preferences. Similar CLI commands that offer full administrative functionality include: `aa admingui`, `si admingui`, and `im admingui`. These commands and `integrity admingui` should not be confused with the `integrity admin` command, which launches an Administration window instance for a specific server. While you can administer all of the administrative applications for that server, you cannot connect to other servers or specify application preferences.

For information on using the Integrity Administration Client, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--height=value`

  specifies the height in pixels of the window.
- `--width=value`

  specifies the width in pixels of the window.
- `--[no]restoreDesktop`

  controls whether to restore any windows that were active when the graphical user interface was closed.

See Also

- Commands: integrity about, integrity disconnect, integrity exit, integrity gui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: options

integrity changemksdomainuserpassword

changes your MKS Domain user password

Synopsis

`integrity changemksdomainuserpassword [--confirmNewPassword=`*value*`] [--newPassword=`*value*`] [--oldPassword=`*value*`] [(-F`*file*`|--selectionFile=`*file*`)] [--hostname=`*server*`] [--port=`*number*`] [--password=`*password*`] [--user=`*name*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [(-F`*file*`|--selectionFile=`*file*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`]`

Description

`integrity changemksdomainuserpassword` changes your password for the MKS Domain, if you are a user in that domain. If you are logged into the Integrity Server, but authenticated against a realm other than MKS Domain, you can still change your password if the username for both realms is the same. This command does not require any permissions assigned to the user.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--confirmNewPassword=`*value*

  specifies a repeat of the contents entered with the `--newPassword` option for accuracy confirmation.
- `--newPassword=`*value*`=`*value*

  specifies your new password for the MKS Domain.
- `--oldPassword=`*value*

  specifies your existing password (for the MKS Domain) that will be changed.

Note

This password may differ from the password you used to connect to the Integrity Server, since you may be authenticated against a domain other than the MKS Domain.

-

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im connect
- Miscellaneous: ACL, diagnostics,options, preferences

Installing Integrity Clients

For information on installing and configuring Integrity Clients, see the *PTC Integrity User Guide*.

Related Links

Server Installation

integrity createmksdomaingroup

creates a new MKS Domain group

Synopsis

`integrity createmksdomaingroup [--description=value] [--members=value] [--name=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-Ffile|--selectionFile=file)] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]markRestricted]`

Description

`integrity createmksdomaingroup` creates a new group in the MKS Domain. For example:

```
integrity createmksdomaingroup --name=Development --members=u=jriley,u=nsingh
```

creates the MKS domain group *Development*, adding the members *jriley* and *nsingh*.

Note

- A user from another domain can be part of an MKS Domain group.
- An MKS Domain group can contain another MKS Domain group as a member.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--description=value`

  specifies an optional description of the group.
- `--members=value`

  specifies members to add to the group, where *value=u=user1,u=user2,...,g=group1,g=group2,...*
- `--name=value`

  specifies the name of the group being created for the MKS Domain.
- `--[no]markRestricted`

  marks the MKS Domain Group as restricted group. Default value is false, if not specified. Principals having *RestrictGroup* permission granted for the *mks:system:mksdomain* ACL can only mark a group as restricted group.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_editmksdomaingroup, integrity_deletemksdomaingroup, integrity_mksdomaingroups, integrity_viewmksdomaingroup
- Miscellaneous: ACL, diagnostics,options, preferences

integrity createmksdomainuser

create a new MKS Domain user

Synopsis

`integrity createmksdomainuser [--email=value] [--fullName=value] [--loginID=value] [--userPassword=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)]`

Description

`integrity createmksdomainuser` creates a user in the MKS Domain. For example:

```
integrity createmksdomainuser --fullname="James Riley" --login=jriley --userPassword=jriley
```

creates the user James Riley in the MKS Domain, with a login and password.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--email=value`

  specifies an e-mail address for the user.
- `--fullName=value`

  specifies the name of the user, including first and last name (user names are stored in the same field).
- `--loginID=value`

  specifies the alphanumeric string the user will use as the login ID to connect to the Integrity Server. Login IDs cannot contain commas, and must not exceed 50 characters in length.

  Caution

  The login ID cannot be edited after the user is created. To change the login ID, the user must be deleted from the domain and created using a different login ID.

- `--userPassword`

  specifies the alphanumeric string the user will use as the password with the Login ID to connect the Integrity Server. Empty passwords for users are not permitted. Users can change their own passwords through the GUI and CLI interfaces.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_viewmksdomainuser, integrity_editmksdomainuser, integrity_changemksdomainuserpassword, integrity_deletemksdomainuser, integrity_mksdomainusers
- Miscellaneous: ACL, diagnostics,options, preferences

integrity deletemksdomaingroup

deletes an MKS Domain group

Synopsis

`integrity deletemksdomaingroup [--[no]confirm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`group1, group2...`*

Description

`integrity deletemksdomaingroup` deletes a group from the MKS Domain.

Caution

You cannot undo a group deletion.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies if to confirm deletion of each group. The default is to confirm.

  Note

  A restricted group can only be deleted by principals having *RestrictGroup* permission granted for the `mks:system:mksdomain` ACL. While deleting a restricted group, if a principal does not have *RestrictGroup* permission, `MKS 1039956` error is displayed.

- *group1, group2...*

  specifies MKS Domain groups to delete from the authentication realm.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomaingroup, integrity_editmksdomaingroup, integrity_mksdomaingroups, integrity_viewmksdomaingroup
- Miscellaneous: ACL, diagnostics,options, preferences

integrity deletemksdomainuser

deletes MKS Domain users

Synopsis

`integrity deletemksdomainuser [--[no]confirm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`user1, user2...`*

Description

`integrity deletemksdomainuser` deletes users from the MKS Domain.

⚠ Caution

You cannot undo a user deletion from the realm. However, you can recreate a user with the same information as one you have deleted.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  controls if to confirm the MKS Domain user deletion.
- *`user1, user2...`*

  specifies the login IDs for users to delete from the MKS Domain.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomainuser, integrity_viewmksdomainuser, integrity_editmksdomainuser, integrity_changemksdomainuserpassword, integrity_mksdomainusers
- Miscellaneous: ACL, diagnostics,options, preferences

integrity disconnect

disconnects Integrity Administration Client from Integrity Server

Synopsis

```
integrity disconnect [--[no]confirm] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [(-Ffile|--selectionFile=file)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`integrity disconnect` disconnects the Integrity Administration Client connection to the host Integrity Server.

Note

When disconnecting a connection that is the current connection, all open client interfaces will use the first initialized connection as the new current connection.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  controls whether to implement the Integrity Server disconnection confirmation policy.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity admingui, integrity exit, integrity gui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: ACL, diagnostics, options

integrity echo

displays a string in UI

Synopsis

`integrity echo [(-?|--usage)] [(-F` *`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`string...`*

Description

`integrity echo` displays a string in the appropriate user interface when used in an event trigger.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- *`string...`*

  specifies a string to display in the appropriate user interface.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers
- Miscellaneous: options

integrity editmksdomaingroup

edits an MKS Domain group

Synopsis

`integrity editmksdomaingroup [--addMembers=value] [--description=value] [--removeMembers=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-Ffile|--selectionFile=file)] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]markRestricted] group...`

Description

`integrity editmksdomaingroup` edits a group from the MKS Domain. The group name is not editable. For example:

```
integrity editmksdomaingroup --addMembers=u=jriley,u=nsingh Development
```

adds the members *jriley* and *nsingh* to the MKS Domain group *Development*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--addMembers=`*value*

  specifies members to add to the group, where *value=u=user1,u=user2,...,g=group1,g=group2,...*

Note

The list of members to add cannot include any of the same members as specified with the `--removeMembers` option.

- `--description=`*value*

  specifies an optional description of the group.
- `--removeMembers=`*value*

  specifies members to remove from the group, where *value=u=user1,u=user2,...,g=group1,g=group2,...*
- `--[no]markRestricted`

  changes the restricted attribute of the group. A restricted group can only be edited by principals having RestrictGroup permission granted for the *mks:system:mksdomain* ACL.

Note

While editing a restricted group, if principal does not have the `RestrictGroup` permission granted for the `mks:system:mksdomain` ACL, the following error is displayed:

MKS 1039956: You must have *RestrictGroup* permission granted for the *mks:system:mksdomain* ACL to be able to administer restricted MKS domain group.

- *group...*

  specifies the selection of a group to edit.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomaingroup, integrity_deletemksdomaingroup, integrity_mksdomaingroups, integrity_viewmksdomaingroup
- Miscellaneous: ACL, diagnostics,options, preferences

integrity editmksdomainuser

edits an MKS Domain user

Synopsis

`integrity editmksdomainuser [--email=`*value*`] [--fullName=`*value*`] [--userPassword=`*value*`] [--hostname=`*server*`] [--port=`*number*`] [--password=`*password*`] [--user=`*name*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [(-F`*file*`|--selectionFile=`*file*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`] `*user...*

Description

`integrity editmksdomainuser` edits the details of a user in the MKS Domain. For example:

```
integrity editmksdomainuser --email=jriley@abcFinancial.com jriley
```

specifies an e-mail address for jriley.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--email=`*value*

  specifies a new e-mail address for the user.
- `--fullName=`*value*

  specifies a new name for the user.
- `--userPassword=`*value*

  specifies a new password for the edited user.
- *user...*

  specifies the login ID of the user you are editing.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomainuser, integrity_viewmksdomainuser, integrity_changemksdomainuserpassword, integrity_deletemksdomainuser, integrity_mksdomainusers
- Miscellaneous: ACL, diagnostics,options, preferences

integrity exit

exits the current Integrity Administration Client session

Synopsis

`integrity exit [--[no]abort] [--[no|confirm]shutdown] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`integrity exit` exits the current Integrity Administration Client session. When you run any `integrity` command from the CLI, or when you open the Administration Client GUI, you start a client session. Only one client session is running at a time, regardless of how many CLIs you are using. To close the GUI you use the appropriate menu commands, and to close the CLI you can use the `integrity exit` command.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]abort`

  controls whether to shut down any other commands that may be running. Some commands allow you to specify a `--persist` option which keeps those commands active during a client session. Using `--abort` with `integrity exit` is recommended for stopping all persistent views that have been specified with another command's `--persist` option.

- `--[no|confirm]shutdown`

  controls the shutting down of the Integrity Administration Client without getting a prompt.

Note

Specifying `--noshutdown` with `integrity exit` is essentially a non-operation: it does nothing.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity gui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: ACL, diagnostics, options

integrity fetchviewset

retrieves a published ViewSet

Synopsis

`integrity fetchviewset [--destination=`*`value`*`] [--[no|confirm]overwriteExisting [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [--[no|Confirm=`*`[yes|no]`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--user=`*`name`*`] [--status=`*`[none|gui|default]`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] `*`viewset...`*

Description

`integrity fetchviewset` retrieves ViewSets that are available on the Integrity Server and copies them to a specified directory. For details about the administration of ViewSets, see the *PTC Integrity Server Administration Guide*. For example:

```
integrity fetchviewset --destination=C:/ViewSets UserViewSet
```

Note

The `integrity fetchviewset` command is useful for examining the contents of a ViewSet from a particular Integrity Server and is not intended to be used for publishing the ViewSet to a different Integrity Server. ViewSets contain references to other server objects by their IDs instead of their names. The consequence is that a ViewSet fetched from one server cannot be safely published to another server since the same ID may not represent the same object on both servers. For example, a ViewSet from server A may have a reference to an Integer field with the ID 71. If the ViewSet was published to server B where the field with ID 71 is a User field, the User field is referenced in the ViewSet instead of the Integer field.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--destination=`*`value`*

  The destination directory for the ViewSet files.
- `--[no|confirm]overwriteExisting`

  overwrites the existing ViewSet with the definitions from the retrieved ViewSet.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity viewsets
- Miscellaneous: ACL, diagnostics, options, preferences

integrity getdbfile

retrieves a configuration file from the database

Synopsis

`integrity getdbfile [--encoding=value] [--output=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...`

Description

`integrity getdbfile` retrieves a configuration file from the database, such as a ViewSet or item presentation template (IPT). Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing. Once you are finished editing these files, you can store them in the database using the `integrity putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=value`

  specifies the code set to save the file in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--output=value`

  specifies the name of the file to store the output to on the local file system.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag=listdbfiles` or `im diag --diag=listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\`*`templatename`*`.xml`, for example, `integrity getdbfile --outputfile=c:/defect.xml data/im/issue/templates/defect.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

integrity gui

launches the Integrity Client graphical user interface

Synopsis

`integrity gui [--height=`*value*`] [--width=`*value*`] [-x`*value*`] [-y `*value*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [(-F `*file*`|--selectionFile=`*file*`)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [(-g|--gui)] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`]`

Description

`integrity gui` launches the graphical user interface (GUI) for the Integrity Client. For information on using the client, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--height=`*value*

  used with the `-g` or `--gui` options, specifies the height of the GUI window, in pixels; `value` must be a whole number.
- `--width=`*value*

  used with the `-g` or `--gui` options, specifies the width of the GUI window, in pixels; `value` must be a whole number.
- `-x`*value*

  used with the `-g` or `--gui` options, specifies the x location in pixels of the window.
- `-y`*value*

  used with the `-g` or `--gui` options, specifies the y location in pixels of the window.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity exit, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: diagnostics, options, preferences

integrity licenses

reports on Integrity licensing usage

Synopsis

```
integrity licenses [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|gui.actions|default]]
```

Description

`integrity licenses` reports on Integrity licensing usage, including license counts per license type within a reporting period, as well as peak and average usage over a period of time when license monitoring is enabled.

License reporting can be helpful to view metrics for current usage to determine whether additional seat licenses are required or whether it is necessary to convert existing floating/concurrent licenses to named/seat licenses. For more information on Integrity licensing, see the *PTC Integrity Server Administration Guide*. For example:

```
integrity licenses
```

returns the following license usage statistics:

- Counts for Floating and Seat license types for each of Configuration Management (SI) and Workflow and Document Management (IM) Integrity components, and a list of principal users who currently have each license type checked out.
- Server License and Flex License usage
  - The usage reporting period identified by the date and time that the server statistics were last reset (which typically corresponds to the date and time of the last server restart), and the date and time that the usage statistics were last collected (e.g., the date and time the command is last run).
  - Usage statistics per license type and component within the recording period that includes a count of the number of licenses current allocated, and the minimum, maximum, and average number of licenses used.
- Licenses Denied: The number of times each license type is not granted.

Note

Users on multiple Integrity Servers show licenses checked out for each server. When multiple Integrity Servers are connected to the same license server, only licenses used by the current server are recorded and reported.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

See Also

- Commands: si connect
- Miscellaneous: options

integrity loadrc

loads the Integrity Administration Client preferences file

Synopsis

integrity loadrc [--[no]merge] [--rc=*value*] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-F*file*|--selectionFile=*file*)] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*]

Description

`integrity loadrc` loads the user's `IntegrityClient.rc` file, which contains your personal preferences for configuring the Integrity Administration Client. If for some reason your personal `IntegrityClient.rc` file has changed, this command will reload your preferences.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]merge`

  controls whether settings from the loaded file should be merged into existing preferences.
- `--rc=`*`value`*

  identifies the file containing settings for running the Integrity Administration Client. The default is the `IntegrityClient.rc` file in your home directory.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity exit, integrity gui, integrity servers, integrity updateclient
- Miscellaneous: ACL, diagnostics, options

integrity logging

modifies Integrity logging levels

Synopsis

`integrity logging [--category=value] [--debug] [--level=value] [--off] [--on] [--target=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`integrity logging` modifies Integrity logging levels.

In addition to modifying the `logger.properties` file, you can use the `integrity logging` command to increase or decrease logging levels for several different categories (as outlined below). You can run this command from a client or server machine, and use it to set client or server side logging. Any category included in the `logger.properties` file can be used. For example:

`integrity logging --target=client --category=CACHE --level=5`

changes *cache* logging on the client to level *5*.

Note

- The client and server do not need to be restarted after making changes.
- Logging changes last only until the Integrity Server or Integrity Client is restarted.
- You need the `DebugServer` permission in the `mks:integrity` ACL to run this command.
- `server.log` logs information about this command when it runs.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--category=value`

  where *value* specifies the category name. Possible values are:

  - `DEBUG`

    Logs debug messages.

    Note

    This command overrides the settings in `logger.properties`, but only until the Integrity Server is restarted. Additionally, this option does not explicitly log debug exceptions. To log exceptions, open `logger.properties`, uncomment `mksis.logger.exception.includeCategory.DEBUG`, and set the value to 10.

  - `SQL`

    Logs all SQL commands to `server.log`. For example, if you view an issue, the SELECT statement is logged. When editing an issue, the INSERT, UPDATE, and SELECT statements are logged. `--level=5` logs all SQL commands. `--level=10` adds additional information such as Rollback time.

    If SQL logging is enabled on the Integrity Server, you can set SQL logging levels for specific users. Setting the logging level can assist in isolating specific user commands and improving server performance. For example, if logging levels are high and server performance is impacted, reducing logging levels may improve performance.

    To set SQL logging levels for a user, type:

    `im/integrity/si --diag=sqllogging username logginglevel`

    where

    username is the user ID of the user whose commands you want to log.

    logginglevel is one of the following number or text values: `high` (`0`), `medium` (`5`), `low` (`10`), or `off` (`-1`).

    For example, typing:

    `integrity diag --diag=sqllogging jriley high`

    logs all Integrity SQL commands for the user `jriley` to the category `SQL-jriley`.

  - `CACHE`

    Logs cache operation information. Levels 0–3 are not verbose. Levels 4–15 are verbose.

  - `SMTP`

    Logs communication between the Integrity Server and SMTP server.

  - `IM-NOTIFICATION`

    Logs e-mail notification.

  - `ACL`

    Logs permission checking to `server.log`. For example, the add label command logs the following:

    ```
    2007-08-16 13:45:17,140 INFO [mksis.IntegrityServer] ACL(5): Check user administrator for permission CreateProject against acl mks:integrity. Resolved ACL: mks:integrity Decision: GRANTED
    ```

  - `TRANSACTION`

    Logs all transactions performed by a specific user, for example, `integrity logging --category=TRANSACTION-jdoe --on` logs all transactions performed by jdoe. To log all cache transactions, type TRANSACTION-system

  - `SILIB`

    Similar to TRANSACTION, this option logs more information about a single command performed by a user, for example, `integrity logging --category=SILIB-jdoe --on`.

  - `SICIAGENT`

    Provides communications between functionality for workflows and documents, and configuration management, and communications between the Integrity Client and Integrity Server.

  - `REALM`

    Logs NT realm information.

  - `API`

    Logs Integrity API information.

  - `HLL`

    Logs Integrity HLL server information.

  - `LDAP`

    Logs LDAP security scheme information.

- `--debug`

is equivalent to `--category=DEBUG`.

- `--level=`*`value`*

  where `value` specifies the log level (-1 disables logging, 10 logs all messages)
- `--off`

  is equivalent to --level=-1 (no reporting in this category).
- `--on`

  is equivalent to --level=10 (logs all messages in this category).
- `--target=`*`value`*

  specifies the target the debugging is enabled for. Valid values for `--target` are `client`, `server`, and `proxy`. The default value for `--target` is `server`.

See Also

- Commands: im purgeauditlog, im viewauditlog, si purgeauditlog, si viewauditlog
- Miscellaneous: options

integrity mksdomaingroups

displays MKS Domain groups

Synopsis

`integrity mksdomaingroups --fields=field1[:width1],field2[:width2]...] [--height=value] [--width=value] [-x value] [-y value] [(-F file|--selectionFile=file)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] group1 group2...`

Description

`integrity mksdomaingroups` displays groups in the MKS Domain.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `description`

    displays the description of the group.
  - `name`

    displays the name of the group.
  - `restricted`

    displays the restricted attribute of the group. A user does not require `RestrictGroup` permission to request this field.
- *group1 group2...*

  specifies names of the groups to display.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomaingroup, integrity_editmksdomaingroup, integrity_deletemksdomaingroup, integrity_viewmksdomaingroup
- Miscellaneous: ACL, diagnostics,options, preferences

integrity mksdomainusers

displays MKS Domain users

Synopsis

`integrity mksdomainusers --fields=field1[:width1],field2[:width2]...] [--height=value] [--width=value] [-x value] [-y value] [(-F file|--selectionFile=file)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] user1 user2...`

Description

`integrity mksdomainusers` displays users in the MKS Domain. If the `--fields` option is not specified, only the login IDs are displayed.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `email`

    displays the email address of the user.
  - `fullname`

    displays the full name of the user, including surname.
  - `loginID`

    displays the login ID of the user.
- *user1 user2...*

  specifies the login IDs of the users to display.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomainuser, integrity_viewmksdomainuser, integrity_editmksdomainuser, integrity_changemksdomainuserpassword, integrity_deletemksdomainuser
- Miscellaneous: ACL, diagnostics,options, preferences

Investigating Integrity Objects Impacting Server Performance

If the Integrity Server is experiencing performance issues, an Integrity object (query, chart, report, dashboard) may be the cause. For example, a report that returns a large amount of data can impact server performance.

While you can notify a user about objects they created that you may suspect are impacting server performance, the user may be busy or unavailable. As an administrator, you can impersonate active users to investigate which object may be the cause of the performance issue. Once you impersonate a user, you have full access to objects created by them. Specifically, you can view, edit, and delete objects created by the user you impersonate.

Impersonating a user is also useful if a user leaves your organization. Before deactivating the user in Integrity, you can impersonate the user to review what objects they created. For example, you could share queries to other users or delete charts that are no longer useful.

Note

To impersonate a user, the Admin permission is required. For more information on impersonation, see the *PTC Integrity Integrations Builder Guide*.

For example, jriley (a developer) frequently runs reports that are suspected to be the cause of server performance issues.

1. To investigate jriley's reports, nsingh (the administrator) creates an impersonate ACL to impersonate jriley:

   ```
   aa addaclentry --acl=""mks:impersonate:user:jriley""
   u=nsingh:Impersonate
   ```

2. nsingh views jriley's reports:

   ```
   im reports -g --impersonateuser=jriley --user=nsingh
   --password=password
   ```

3. nsingh views the report he suspects is causing the performance issue:

   ```
   im viewreport -g --impersonateuser=jriley --user=nsingh
   --password=password jrileyQuery
   ```

4. nsingh edits the report to reduce the amount of data the report returns:

   ```
   im editreport -g --impersonateuser=jriley --user=nsingh
   --password=password jrileyQuery
   ```

Related Links

Server Troubleshooting

integrity publishviewset

publishes a ViewSet to Integrity Server

Synopsis

`integrity publishviewset [--canImport=user:<user>|group:<group>,..] [--canModify=user:<user>|group:<group>,..] [--[no]customizable] [--description=value] [--[no]mandatory] [--name=value] [--[no|confirm]overwriteExisting] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-F file|--selectionFile=file)] file`

Description

`integrity publishviewset` publishes a ViewSet to Integrity Server. For detailed information on publishing ViewSets, see the *PTC Integrity Server Administration Guide*.

This command is available to support ViewSet backward compatibility for ViewSets from MKS Integrity 2006, intended for use by Integrity Client 2006 users. To publish ViewSets from later releases, it is recommended that you use the Integrity Administration Client. For information on support for MKS Integrity 2006 ViewSets, see the *PTC Integrity Upgrading Guide*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--canImport=user:<user>|group:<group>,..`

  specifies users or groups in the security realm who can import the ViewSet.

  Tip

  When publishing a 2006 ViewSet, Integrity Client 2009 users are also able to view and import that ViewSet. If you only desire Integrity Client 2006 users to import the ViewSet, create a group in the security realm for those users and then specify it using `--canImport`.

- `--canModify=user:<user>|group:<group>,..`

  specifies the users or groups in the security realm who can modify the ViewSet.

- `--[no]customizable`

  specifies if the ViewSet can be customized by users after it is imported. The default is customizable.

  Note

  If `--mandatory` is specified, this option is not valid and the ViewSet is not customizable.

- `--description=value`

  specifies an optional description for the ViewSet. Use quotation marks for the description if it includes spaces.

- `--[no]mandatory`

  specifies if the ViewSet is mandatory. For a detailed description of mandatory ViewSet behaviour and their impact on users, see the *PTC Integrity Server Administration Guide*. By default, ViewSets are not mandatory. This option cannot be used with `--customizable`.

- `--name=value`

  specifies a new name for the ViewSet you are publishing. Use quotation marks if the name includes spaces.

- `--[no|confirm]overwriteExisting`

  determines the action to take if overwriting an existing ViewSet on the Integrity Server. If none of the following is specified, ViewSets are overwritten without Integrity prompting you.

  - `no` causes the command to fail without prompting.
  - `confirm` prompts you for a decision (default).

- *`file`*

  specifies the location and filename of the ViewSet you are publishing. For example, `c:/temp/ViewSet.vs`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity loadrc, integrity setprefs
- Miscellaneous: ACL, diagnostics, options, preferences

integrity putdbfile

puts a configuration file into the database

Synopsis

```
integrity putdbfile [--encoding=value] [--input=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...
```

Description

Although PTC recommends creating and editing ViewSets and IPTs in the GUI, you can retrieve ViewSets and IPTs from the database for manual editing using the `integrity getdbfile` command. Once you are finished editing these files, you can store them in the database again using the `integrity putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=`*`value`*

  specifies the code set the file is saved in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--input=`*`value`*

  specifies the name of the file on the local file system containing the input.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag=listdbfiles` or `im diag --diag=listdbfiles`. For example, a valid file could be `data\im\issue\templates\defect.xml`, which is the IPT for the Defect type. For each IPT in Integrity, there is an XML file under `data\im\issue\templates\`*`templatename`*`.xml`, for example `integrity putdbfile --input=c:/defect.xml data/im/issue/templates/defect.xml`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

integrity serveralerts

displays Integrity Server alert messages on all currently connected servers

Synopsis

```
integrity serveralerts [--height=value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]
```

Description

`integrity serveralerts` displays Integrity Server alert messages for all servers that you are currently connected to. The alert message displays who sent the message, the server it came from, when it was sent, and the message. If you are not connected to any servers, a message informs you that there are no alert messages. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

Note

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

See Also

- Commands: integrity viewserveralert, integrity setserveralert, integrity admingui, integrity servers
- Miscellaneous: options

integrity servers

displays the current connections to an Integrity Server

Synopsis

`integrity servers [--[no]showVersion] [--height=value] [--width=value] [-xvalue] [-yvalue] [-?|--usage] [-N|--no] [-Y|--yes] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [-Ffile|--selectionFile=file] [--[no]persist]] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`integrity servers` displays active server connections in the format *user@host_name:port*.

The default server connection is indicated by *user@host_name:port(default)*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showVersion`

  controls whether to show build version information for the connected server. The presentation of this information is in the format *[Build: 4.5.0.4652.7.1]*.

- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity exit, integrity gui, integrity loadrc, integrity updateclient
- Miscellaneous: diagnostics, options, preferences

integrity setprefs

sets Integrity Administration Client preferences

Synopsis

`integrity setprefs [--command=value] [--[no]resetToDefault] [--[no]save] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] pref[=value]...`

Description

`integrity setprefs` sets Integrity Administration Client preferences. These settings are used to determine default behaviors for other commands - each option on each command has a preference key associated with it. The integrity viewprefs command lists the commands and preference keys. Changes to your preferences are either for the current client session (until integrity exit is used) or may be permanently saved in your system's `home` directory, into a file named `IntegrityClient.rc`, using the `--save` option.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--command=value`

  identifies the command to be set. For an easy way to see a list of commands and values that may be set, simply type the integrity viewprefs command, either piped through `|more` or redirected to a file, for example:

  ```
  integrity viewprefs --global --showValidValues >prefs.txt
  ```

  The commands and preference keys are also listed on the preferences reference page.

- `--[no]resetToDefault`

  controls whether to revert specified settings to the default values as shipped with the Integrity client. If specifying `--resetToDefault`, you must not specify `=value` for each preference.

- `--[no]save`

  controls whether changes should be permanently saved.

- `--[no]ask`

  controls prompts to the user for specific preferences. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:

  - `once`

    asks the user the first time only, and then uses the provided value every time after.

  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).

  - `element-last`
  - asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.

- For example, to set the server host for Integrity to connect to a specific host name, you specify something like:

  ```
  integrity setprefs --command=connect
  server.hostname=specific.hostname.com
  ```

  but to set the preference to ask for a host name, you specify something like:

  ```
  integrity setprefs --command=connect --ask
  server.hostname=element-last
  ```

- `--ui=[unspecified|gui|cli|api]`

  controls whether to apply the preference to the graphical user interface, the command line interface, the application programming interface, or when the interface is unspecified. By default, `--ui=unspecified` is used and applies any changes you make to all interfaces unless there is already an interface specific preference configured.

  These correlate to settings in the `IntegrityClient.rc` file, which can be seen as having the `gui.integrity.`, `api.integrity.` or `cli.integrity.` prefix, or simply the `integrity.` prefix when it is unspecified.

- `pref[=value]...`

  identifies the preference string. If you specified the `--resetToDefault` option, then you only need to specify the preference name; otherwise specify a value for the preference. Use spaces to specify multiple preferences.

-

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity loadrc, integrity viewprefs
- Miscellaneous: diagnostics, options, preferences

integrity setproperty

configures Integrity properties stored in the database

Synopsis

`integrity setproperty [--comment=`*value*`] [--restoreDefault] [--value=`*value*`] [-?|--usage] [-N|--no] [-Y|--yes] [--hostname=`*server*`] [--port=`*number*`] [--user=`*name*`] [--password=`*password*`] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [-g|--gui] [-F`*file*`|--selectionFile=`*file*`] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`] `*string...*

Description

`integrity setproperty` configures Integrity properties stored in the database. Properties specify information that affects the operation of the Integrity Server, workflows and documents, and configuration management. Other properties are stored and configured in properties files on the server's file system. For a complete list of configurable properties and possible values, see the *PTC Integrity Server Administration Guide*.

Note

- Some properties require the server to be restarted for the changes to take effect.
- Access to configuring properties is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document properties, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management properties, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all properties.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--comment=`*value*

  specifies a comment associated with the change made to the property. While PTC recommends specifying a comment for troubleshooting purposes, a comment is optional.
- `--restoreDefault`

  restores the property to the default value.
- `--value=`*value*

  specifies the new value of the property.
- *string...*

  specifies the name of the property you want to configure.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity admin, integrity gui
- Miscellaneous: diagnostics, options, preferences

integrity setserveralert

creates and clears Integrity Server alert messages

Synopsis

`integrity setserveralert [--clear] [--message=value] [--messageFile=value] [--height=value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`integrity setserveralert` sends alert messages to all users currently logged in to the Integrity Server, or clears the current alert message. The alert message displays who sent the message, the server it came from, when it was sent, and the message. Sending alert messages is useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

Note

- To create and clear alert messages, you need one of the following permissions: `mks:AdminServer`, `mks:im:Admin`, `mks:im:AdminServer`, `mks:si:AdminServer`.
- You can create one message for a given server at a single time. Creating a new message deletes the previous message.
- Alert messages are stored in the server's memory. If the server shuts down, the message is deleted.
- If a user connects to the server after a message is sent, the user still receives the message.
- For users working in the Integrity Client GUI with the system tray or Web interface for workflows and documents, alert messages appear in a window at the bottom of the interface. Users working in the CLI and Integrity Client GUI without the system tray must manually check for alert messages from the CLI.
- Alert messages are plain text only and allow a maximum 4000 KB.
- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--clear=server`

  clears the current alert message.
- `--message=value`

  specifies the alert message to send to all users currently logged in to the Integrity Server.
- `--messageFile=server`

  specifies a file containing the alert message to send to all users currently logged in to the Integrity Server.
- `--hostname=server`

  specifies the name of the Integrity Server to send the alert message to.
- `--port=number`

  specifies the port of the Integrity Server to send the alert message to.

See Also

- Commands: integrity serveralerts, integrity viewserveralert, integrity admingui, integrity servers
- Miscellaneous: options

integrity stats

reports on Integrity Server statistics

Synopsis

```
integrity stats [--[no]deltaOnly] [--[no]description] [--csv] [--startdate=value] [--enddate=value] [--filter=value] [--file=value] [--reset] [--target=client,proxy,server] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-Ffile|--selectionFile=file)] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--password=password] [--user=name]
```

Description

To monitor and diagnose issues that may arise when performing operations on the Integrity Client and Integrity Server, the Integrity Server stores statistics about these operations at defined intervals, storing the deltas (differences) in a database table. You can then import the delta information into Microsoft Excel to create a histogram of a specific statistic tracked over time, allowing you to monitor and diagnose areas of degradation.

Note

To define the intervals at which statistics are recorded and stored in the database, configure the `mksis.statisticsInterval` policy, described in the *PTC Integrity Server Administration Guide*.

The `integrity stats` command offers the following modes (using specific options) to display statistics:

- Normal mode (no options) displays all statistics as of the current time.
- Interval mode displays all statistics for the specified interval. For each interval, new statistics display, showing the difference between the latest and previous statistics.
- Historic mode (the `--startdate` option) displays the two statistics closest to the specified start date, showing the differences between them.
- Histogram mode (the `--startdate` and `--enddate` options) displays the statistics closest to the starting date and after the ending date, and all snapshots between are displayed. Differences display for each adjacent statistic. For example, for 10 snapshots, 9 deltas display.
- File mode (the `--file` option) uses a file containing a single raw statistics dump, via a support package. For example, you could view the statistics from either the Statistics entry, or one of the Stats/time entries found in the support package.
- File Histogram mode (multiple `--file` options) uses each specified file as a statistics/time file extracted from the support package.

  Note

  The actual time the snapshot was collected is not actually stored in the snapshot; some output formats use this time. If the filename is not parsable, then the time in those formats will not be available.

- Reset mode (the `--reset` option) clears the collected statistics.

Note

- You must have the `AdminServer` permission, or one of the following permissions for configuration management or workflows and documents: `AdminServer` or `DebugServer`.
- This command is subject to removal or change without notice, and may expose internal information that is not documented.
- If `--startdate`/`--enddate`, or `--file` is not specified, the statisics displayed are those from the time the server started (or last reset) to the current time.
- If `--startdate`/`--enddate` is specified, statistics data recorded in the database is used. The interval that statistics are stored is specified by the `mksis.statisticsInterval` policy.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]deltaOnly`

  specifies if to show only deltas when an interval is specified. By default, two rows display: the delta row and the actual value row. Specifying `--deltaOnly` displays the delta row only.

- `--[no]description`

  specifies if to display a description of the statistics columns on each line. This option is applicable to non-CSV mode only.

- `--csv`

  displays the output as comma-separated data. A single header line is always produced with the column headers. Each value is in its appropriate comma-delimited column, or is reserved with an empty comma.

  By default, the following columns display: *startdate,enddate,groupname,statisticsname,count,total,average*. The complete list of CSV columns you can display include:

  - startdate
  - enddate
  - groupname
  - statisticname
  - kind (includes the keywords `data`, `ratio`, or `simple`)
  - unit (statistic dependant: Ms (microseconds), ms (milliseconds), #issues, #bytes, objects, listeners, revs
  - count
  - total
  - total
  - sum
  - min
  - max
  - average
  - mode (includes the keywords `delta` or `cumulative`)

  For example, output filtered by the Agent group displays as:

  ```
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,getCurrentUser,1,0,0,0,0
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,getIssue,1,39,39,39,39
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,getIssues,1,312,312,312,312
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,getServerVersion,1,0,0,0,0
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,getUserPreferences,1,175,175,175,175
  Wed Jun 27 15:50:43 EDT 2009,Wed Jun 27 15:51:43 EDT 2009,Agent,upsync,2,260,130,248,12
  Wed Jun 27 15:51:43 EDT 2009,Wed Jun 27 15:52:43 EDT 2009,Agent,getQueries,1,138,138,138,138
  Wed Jun 27 15:51:43 EDT 2009,Wed Jun 27 15:52:43 EDT 2009,Agent,upsync,1,0,0
  ```

```
Wed Jun 27 15:54:43 EDT 2009,Wed Jun 27 15:55:43 EDT 2009,Agent,getIssue,6,117,19
Wed Jun 27 15:54:43 EDT 2009,Wed Jun 27 15:55:43 EDT 2009,Agent,getIssues,6,48,8
Wed Jun 27 15:54:43 EDT 2009,Wed Jun 27 15:55:43 EDT 2009,Agent,upsync,6,1,
```

Tip

Import the resulting data into Microsoft Excel and use the pivot table functionality to create a histogram that can be further filtered.

- `--startdate=`*value*

  specifies the first date in a date range, or a single date for a single histogram point. This option can be used with the `--enddate=`*value* option. For example, If you specify 11:30 as the start date and 3:30 as the end date and the statistics are generated hourly, statistics generated at 11:00, 12:00, 1:00, 2:00, 3:00, and 4:00 display.

Note

If you specify a start date, but no end date, a pair of statistics display for the specified time. For example, if you specify 11:30 and the statistics are generated hourly, statistics generated at 11:00 and 12:00 display.

- `--enddate=`*value*

  specifies the last date in a date range. This option must be used with the `--startdate=value` option.
- `--filter=`*value*

  specifies a statistic to display, using the form `--filter=`*groupname* or `--filter=`*groupname.statisticsname*. This option can be specified multiple times to output multiple groups. If a filter is not specified, all statistics display.
- `--file=`*value*

  specifies to read raw statistics from a file. A file containing raw statistics may be obtained from the support package; the support package has files in the Stats directory for various times. This setting may be specified multiple times: thus invoking it against the support package directory will allow you to obtain histograms of usage.
- `--reset`

  resets the statistics.

Note

Statistics on the servers are global; any statistics not recorded to the database are deleted when the `--reset` option is specified.

- `--target=`*client*`,`*proxy*`,`*server*

  specifies to operate against a specific target, such as the client, proxy, or server.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity setprefs
- Miscellaneous: diagnostics, options, preferences

integrity updateclient

updates the Integrity Administration Client with a service pack

Synopsis

`integrity updateclient [--[no|confirm]download] [--[no|confirm]shutdown] [--[no|confirm]rollback] [--[no|confirm]rollbackshutdown] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`integrity updateclient` updates theIntegrity Administration Client with a service pack from the server if one is available. A service pack may be designated as required to address a known issue, or may provide enhancements.

Options

This command takes some of the universal options available to `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]download`

  automatically downloads a service pack if one is available.
- `--[no|confirm]shutdown`

  automatically shut downs the client if a service pack is downloaded.
- `--[no|confirm]rollback`

  automatically initiates a service pack rollback, if required to connect to the Integrity Server.
- `--[no|confirm]rollbackshutdown`

  automatically shutdowns the client if a service pack rollback is initiated.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity about, integrity admin, integrity disconnect, integrity exit, integrity gui, integrity loadrc, integrity servers
- Miscellaneous: ACL, diagnostics, options

integrity viewmksdomaingroup

displays MKS Domain group details

Synopsis

`integrity viewmksdomaingroup [--height=value] [--width=value] [-x value] [-y value] [(-F file|--selectionFile=file)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] group1, group2...`

Description

`integrity viewmksdomaingroup` displays the details of groups in the MKS Domain. The details include the name, restricted attribute, and description of the group, as well as a list of group members.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- *group1, group2...*

  specifies the names of the groups to view.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomaingroup, integrity_editmksdomaingroup, integrity_deletemksdomaingroup, integrity_mksdomaingroups
- Miscellaneous: ACL, diagnostics, options, preferences

integrity viewmksdomainuser

displays MKS Domain user details

Synopsis

`integrity viewmksdomainuser [--height=`*value*`] [--width=`*value*`] [-x `*value*`] [-y `*value*`] [(-F `*file*`|--selectionFile=`*file*`)] [--hostname=`*server*`] [--port=`*number*`] [--password=`*password*`] [--user=`*name*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*directory*`] [--forceConfirm=`*[yes|no]*`] [(-g|--gui)] [--quiet] [--settingsUI=`*[gui|default]*`] [--status=`*[none|gui|default]*`] `*user1, user2...*

Description

`integrity viewmksdomainuser` displays the details of a user in the MKS Domain. The details include the login ID, full name, and email address of each user.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- *`user1, user2...`*

  specifies the login IDs of the users to view.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity_createmksdomainuser, integrity_editmksdomainuser, integrity_changemksdomainuserpassword, integrity_deletemksdomainuser, integrity_mksdomainusers
- Miscellaneous: ACL, diagnostics,options, preferences

integrity viewprefs

displaysIntegrity Administration Client preferences

Synopsis

`integrity viewprefs [--[no]global] [--command=`*`value`*`] [--[no]showValidValues] [--[no]ask] [--ui=`*`[unspecified|gui|cli|api]`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`integrity viewprefs` displays Integrity Administration Client preferences.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions. For an easy way to see a list of commands and values that may be set, simply type the `integrity viewprefs` command, either piped through `|more` or redirected to a file, for example:

```
integrity viewprefs --global --showValidValues >prefs.txt
```

Alternatively, the `--gui` option presents a simple-to-use dialog box that lets you view and configure the preferences.

- `--[no]global`

  controls whether to view all preferences. By default, all preferences are displayed, including those marked as global. Specifying `no` displays preferences that are not marked as global.
- `--command=`*`value`*

  identifies the command preference to be viewed.
- `--[no]showValidValues`

  controls whether to list valid values for the preferences.
- `--[no]ask`

  controls whether to view the ask control over the preference options. See the description for `--[no]ask` on the integrity setprefs command.
- `--ui=[`*`unspecified`*`|`*`gui`*`|`*`cli`*`|`*`api`*`]`

  controls whether to apply the preference to the graphical user interface, the command line interface, the application programming interface, or when the interface is unspecified. By default, `--ui=`*`unspecified`* is used and applies any changes you make to all interfaces unless there is already an interface specific preference configured.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity loadrc, integrity setprefs
- Miscellaneous: ACL, diagnostics, options, preferences

integrity viewserveralert

displays Integrity Server alert messages for a target server and all related servers

Synopsis

`integrity viewserveralert [--height=value] [--width=value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`integrity viewserveralert` displays Integrity Server server alert messages for a target sever and all related servers, for example, a configuration management server and a proxy server. The alert message displays who sent the message, the server it came from, when it was sent, and the message. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.
- A connection to the target server is required for the `integrity viewserveralert` command to display alert messages.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=server`

  specifies the host name of the target Integrity Server to retrieve alert messages from.
- `--port=number`

  specifies the port of the target Integrity Server to retrieve alert messages from.

See Also

- Commands: integrity serveralerts, integrity setserveralert, integrity admingui, integrity servers
- Miscellaneous: options

integrity viewsets

displays available ViewSets

Synopsis

`integrity viewsets[--fields=field1[:width1],field2[:width2]... [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-Ffile|--selectionFile=file)] [(-g|--gui)] [--fieldsDelim=value] [--quiet] [--settingsUI=[gui|default]] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [--status=[none|gui|default]]`

Description

`integrity viewsets` displays a list of all ViewSets available on the Integrity Server and the local machine. For details about publishing ViewSets, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `integrity` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field*n* can be any of the following:

  - `creator`

    displays the name of the user who created (if a personal or unpublished ViewSet) or published the ViewSet.
  - `customizable`

    displays if users can change the contents of ViewSet menus and toolbars.
  - `description`

    displays a description of the ViewSet as added by the creator.
  - `filename`

    displays the path to where ViewSet resides on your machine.
  - `mandatory`

    displays mandatory ViewSets. Mandatory ViewSets are automatically returned by the Integrity Client and updated when new versions become available on the Integrity Server.
  - `modifieddate`

    displays the date the ViewSet was last modified.
  - `name`

    displays the name of the ViewSet. By default, only the name is shown.
  - `publishedstate`

    displays the published state of the ViewSet.
- `--fieldsDelim=value`

  specifies the string to be used as a delimiter between fields displayed in the CLI.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: integrity fetchviewset
- Miscellaneous: diagnostics, options, preferences

options

applicable to CLI commands

Description

Some CLI commands (`si`, `im`, `aa`, `integrity`) share general options, while all take certain universal options. This reference page is provided as a guide to these common options.

This reference page contains the following information:

- Specifying Members, Sandboxes and Projects for Configuration Management Commands
- Specifying Sandboxes Explicitly or Implicitly for Configuration Management Commands
- Specifying Configuration Management Projects
- Specifying Rules
- General Options
- Universal Options

Specifying Members, Sandboxes and Projects for Configuration Management Commands

There are three types of configuration management commands, and therefore the way you specify the *member* varies:

1. Some commands can only be executed on members in the context of a Sandbox, because they manipulate the member's working file. These are noted as requiring a *sandbox member...*, such as `si ci`, `si co`, and `si merge`. These take a Sandbox member, and you may not perform the operation against a project member. If you try to specify a project for these commands, you will see an error message.
2. Other commands do not manipulate a member's working file, and therefore can be performed directly in the context of a project. If you specify a Sandbox, it is simply used as a pointer to find the project itself. The fact that you specify the Sandbox is only incidental. These are noted as requiring a *project member...*, and most of the commands that say *member...* fall into this category; for example, `si updaterevision`, `si updatearchive`, and `si addlabel`.
3. There are also some commands that perform differently depending on whether they are given a *sandbox* or a *project*. Examples for this would be `si diff` and `si edit`.

Specifying Sandboxes Explicitly or Implicity for Configuration Management Commands

You can explicitly specify either `-P` or `-S` options for most commands. For `-P` you must specify a project or subproject, the `-P` option does not accept the filename of a Sandbox. For `-S` you must specify a Sandbox or sub Sandbox, `-S` does not accept the filename of a project.

Integrity also allows for implicit Sandbox selection. This means that you can use commands without explicitly specifying a `-S` *sandbox* option, and Integrity determines the Sandbox to operate on based on the directory you are working in.

For example, suppose you are working in the directory `C:/test/sbx/sub1/sub2`, and suppose you have a Sandbox only in the `/sbx` directory. Using the `-S` option to explicitly specify the Sandbox, you might enter the following to check in a file:

```
si ci -S c:/test/sbx/project.pj header.c
```

Using implicit Sandbox location to check in a file, you might enter:

```
si ci header.c
```

If the Sandbox is not specified explicitly through `-S`, Integrity tries to locate one by starting in the current working directory and seeing if there is a Sandbox registered in that directory. If there isn't, it searches up the directory tree until it either finds one, or until it reaches the root of the drive. In the example provided, Integrity first determines there is no Sandbox in `/sub2`, then `/sub1`, then finds the Sandbox in `/sbx` and uses that Sandbox.

Now suppose you have Sandboxes in each of `/sbx`, `/sub1`, and `/sub2`. This implicit Sandbox selection allows you to work with multiple Sandboxes in one command line entry, and without having to explicitly specify lengthy locations for each one. If you're working in the `C:/test/sbx` directory and decide to check in files to each Sandbox, for example, you might enter:

```
si ci header.c sub1/comp.c sub1/sub2/img.c
```

This checks in the file `header.c` to the Sandbox at `C:/test/sbx` , checks in the file `comp.c` to the Sandbox at `C:/test/sbx/sub1`, and checks in the file `img.c` to the Sandbox at `C:/test/sbx/sub1/sub2`.

A requirement for implicit Sandbox location to operate correctly is that there can be no more than one Sandbox (or sub Sandbox) in a single directory. If you create two or more Sandboxes in the same directory, the implicit Sandbox location algorithm cannot unambiguously determine which Sandbox to use in that directory, and it prompts you to clarify by specifying the name of the Sandbox that you want to use in that case. In general, you shouldn't create multiple Sandboxes in the same directory.

Note

Certain `si` commands do not operate on Build Sandboxes, which are created as read-only for the purpose of building a programming artifact. Using inappropriate commands with a Build Sandbox causes error messages to appear.

Specifying Configuration Management Projects

This section provides information on the two available syntaxes for specifying projects, followed by examples of their usage. The following two syntaxes are available:

- Source Configuration Path

  A keyword-based string that provides the ability to specify subprojects within the context of a project tree (see examples that follow).
- Flat Path

  The legacy syntax that may not be supported in future releases. It takes the form of a simple pathname string, possibly accompanied by a development path name or a project checkpoint.

WELL FORMED PROJECTS: PTC recommends using well formed projects wherever possible. A well formed project is one where every directory contains a subproject (if not possible, then all members should belong to the nearest enclosing subproject), and that subproject is named `project.pj`.

BENEFIT: Well formed projects use more compact paths. Using a well formed project elminates the need to use hash (#) values for specifying projects (refer to Path ambiguity example).

When specifying Sandbox subproject scope, to avoid potential ambiguities, PTC recommends that the subproject name is specified using a Sandbox-relative source configuration path to specify subprojects within the context of a project tree. Use explicit non-compact keyword string syntax to match the path definition defined in the GUI instead of using compact hash syntax.

For example, the following source configuration path:

```
#s=sub/project.pj#s=sub2/project.pj#s=sub3/project.pj
```

is the same as the following compact source configuration path:

```
#sub/sub2/sub3 or #sub#sub2#sub3
```

however, the non-compact keyword string syntax is recommended to account for cases where project paths are not necesarily well-formed. For more details on specifying Sandbox subproject scope, see the `si createsandbox` command man page.

SCENARIOS WHERE SOURCE CONFIGURATION PATH IS SUPERIOR TO FLAT PATH SPECIFICATION

The following scenarios are documented in this section, and illustrate how using the source configuration path syntax is the superior choice compared to flat path specification:

- Subproject not on main development path
- Path ambiguity
- Ambiguous co-located subprojects

The following keywords are used in the examples:

- The # keyword specifies the well-formed project or subproject name. Well-formed project and subproject names end with `project.pj`.
- The `#a` keyword specifies the date of the project configuration.
- The `#d` keyword specifies the development path name.
- The `#s` keyword specifies the subproject in a poorly-formed project tree. A poorly-formed project tree has co-located subprojects or subprojects located more than one directory level deep. Using this keyword, you can only specify one subproject for each occurrence of the keyword.

For a description of all keywords, see the `-P` option under General Options.

SUBPROJECT NOT ON MAIN DEVELOPMENT PATH

The flat path cannot handle the case where the object (in this case a sub project on a variant) Integrity is locating does not exist in the main project tree (on the main devpath).

In the following diagram, user needs to specify project `sub2` on devpath Dev.

- Tree 1

  /aurora_project/project.pj

  |

  |

  --- sub/project.pj
- Tree 2

  /aurora_project/project.pj (Development Path Dev)

  |

  |

  ---sub/project.pj

  |

  |

  ---sub2/project.pj

Flat Path: `-P /aurora_project/sub2/project.pj --devpath Dev`

This syntax does not work because Integrity attempts to find `/aurora_project/sub2/project.pj` on the main devpath first, and then jumps from there into the development path, but in this case that subproject does not exist in the main devpath.

Source Configuration Path: `-P"#/aurora_project#d=Dev#sub2"`

This syntax works because it instructs Integrity to start with /aurora_project (which does exist in main devpath), then jump to devpath Dev, then move into `sub2`.

Summary: In this scenario, there is no way to specify sub2 using the flat path syntax because it is not on the main development path. You must use the source configuration path syntax to specify `sub2`.

PATH AMBIGUITY

In some cases, the flat path lacks the ability to specify the desired project with no ambiguity.

In the following diagram, the user needs to specify subproject `beta/project.pj` (but only from the location in Tree 1 below).

- Tree 1

  /aurora_project/project.pj

  |

  |

  ---codebase/project.pj (Development Path Dev1)

  |

  |

  ---beta/project.pj (Development Path Dev2)
- Tree 2

  /aurora_project/codebase/project.pj [Reg as top-level proj]

  |

  |

  ---beta/project.pj

Flat Path: `-P/aurora_project/codebase/beta/project.pj`

This syntax is ambiguous, because it could be specifying the project on either node. Integrity picks the first project it finds in the registry, which may not be the one desired. The contents of each project are not the same because one is on devpath Dev2, while the other is on the main devpath.

Source Configuration Path:

`-P "#/aurora_project#codebase#beta"`

`-P "#/aurora_project/codebase#beta"`

This syntax method is unambiguous. It is completely clear which beta subproject location is specified.

Note

The short form is used because the projects are assumed to be well formed. The long form example would be: `-P #p=/aurora_project/project.pj#s=codebase/project.pj#s=beta/project.pj`

`-P #p=/aurora_project/codebase/project.pj#s=beta/project.pj`

Summary: In this scenario, the flat path method is ambiguous while the source configurion path is not. The source configuration path can never be ambiguous.

AMBIGUOUS CO-LOCATED SUBPROJECTS

The flat path syntax cannot specify co-located suprojects (subprojects located in same directory).

In the following diagram, two subprojects are located in the same directory.

- Tree 1

  `/aurora_project/source_code/root.pj`

  `/colocatedsub.pj`

  Note

  Project colocatedsub.pj is in the same directory as root.pj but is the subproject of root.pj in hierarchy.

- Tree 2

  `/aurora_project/source_code/base.pj`

/colocatedsub.pj

Note

Project colocatedsub.pj is in the same directory as base.pj but is the subproject of base.pj in the hierarchy.

In this scenario, root.pj, base.pj, and colocated.pj are all in the same directory.

File Path:-P /aurora_project/source_code/colocatedsub.pj

This syntax method is ambigious because it is unclear which project the co-located project belongs to. There might be different policies that affect how colocatedsub.pj is administered or used.

Source Configuration Path -P #/aurora_project/source_code/root.pj#s=colocatedsub.pj

This syntax method clearly specifies a registered project and a subproject where there are co-located subprojects

Summary: In this scenario, the flath path syntax method is unable to specify the co-located subproject. Only the source configuration path syntax can specify the desired path.

Rules for Jumps

When jumping to a specific configuration in a project path, the following rules apply:

- You cannot jump anywhere from a build project
- You can jump from a normal project to a variant only if it is the root of the variant (the project through which the development path was created)
- You cannot jump to a variant if it differs from the closest variant higher in the project hierarchy (if there is a higher variant). When no subprojects are configured as variants in the hierarchy, the closest variant is the variant of the top-level project. When at least one subproject in the hierarchy is configured as a variant, the closest variant is the variant of the lowest configured subproject. This does not include the variant of the subproject on which the jump is specified, if it is currently configured as a variant.

The last two rules are verified based on the type of the parent project. You can always jump to the current configuration of a subproject, even if it violates the rules listed above.

The following provides examples of how jump rules are applied when jumping to a variant. If you had the following project setup:

/projects/aurora_project/source_code/savings_tool/project.pj

where source_code is a subproject currently configured as beta_variant and savings_tool is a shared subproject currently configured as normal.

The following jump would be allowed:

-P #/projects/aurora_project#source_code/savings_tool#d=beta_variant

The following jump would not be allowed:

-P #/projects/aurora_project#source_code/savings_tool#d=prod_variant

You can specify a jump to beta_variant from the subproject savings_tool because it is the same as the variant for source_code, and because as a shared subproject it is accepted as the local variant root (the project through which the development path was created). You cannot jump to prod_variant because it is different than the variant of source_code.

The following jumps would also be allowed:

-P #/projects/aurora_project#d=SP4#source_code#d=SP4

-P #/projects/aurora_project#d=SP4#source_code#d=beta_variant

The following jump would not be allowed:

-P #/projects/aurora_project#d=SP4#source_code#d=prod_variant

You can specify a jump to SP4 from the subproject source_code because it is the same as the variant for aurora_project. You can specify a jump to beta_variant because source_code is currently configured as beta_variant. You cannot jump to prod_variant because it is different than the variant of aurora_project.

Note

If you are using a case-insensitive database repository, you can use case-insensitive keyword-based strings.

Tip

If the path contains a hash character (#), use a second hash character to escape it. For example:

-P #/projects/C##/aurora_project#d=SP4#source_code#d=SP4

Specifying Rules

Some CLI commands share the same rule syntax. The rule is of the following form:

A <rule> rule is defined as an <expression>

An <expression> is defined as one of the following:

(<expression> and <expression> and <expression>)

(<expr> or <expr> or <expr>)

(<user op>)

(<field> <operator> <value>)

where for users and groups:

<user op>rule is defined as a user is not a member of "<group>"

<user op>rule is defined as a user is a member of "<group>"

<value>rule is defined as a <field> | "<text>" | "<number>"

<field> rule is defined as a field[<fieldname>]

<fieldname> rule is defined as a ID | Summary | Priority | ...

<group> rule is defined as a (everyone | im-dev | ...)

<operator> rule is defined as a (= | > | >= | <= | < | <>)

<number> rule is defined as a (.. | -1 | 0 | 1 | ..)

For example:

(((field[Summary] = "Hello") or (user is a member of "everyone"))

and

((field[ID] = "1") or (field[ID] = "2") or (field[ID] = "5"))

and

(field[Summary] <> field[Description]))

For trigger rules, same syntax applies but:

<value> is defined as <field> | "<text>" | "<number>"

<field> is defined as field'[<fieldname>]

<field> is defined as field[<fieldname>]

<fieldname> is defined as ID | Summary | Priority | ...

<operator> is defined as (= | > | >= | <= | < | <>)

<number> is defined as (.. | -1 | 0 | 1 | ..)

For example:

(((field[Summary] = "Hello") or (field'[Summary] = field[Summary]))

and

((field[ID] = "1") or (field[ID] = "2") or (field[ID] = "5"))

and

(field[Priority] <> field'[Priority]))

To prevent the field from being displayed, specify the rule value to be "(false)". This option is useful if you want to hide read-only custom fields (i.e. phase, range, computed) in Issue Detail and Issues views, but still be able to query on them and use them in column sets. Enabling this option replaces any existing rules with "(false)".

To specify a date and time for a date field, use the *MM/dd/yyyy h:mm:ss [AM|PM]* format. You can specify a time only if the date field is configured to display the time. To specify the current date for a date or date/time field, type *today*. To specify an empty value for the date field, type *none*.

For rules specified in commands that use the --notificationRule, --notificationRuleFile, --rule, and --ruleFile options, you can also compare the value of a field with another field or constant, by specifying an apostrophe ('). For example, Created Date' compares the value of the field with another field (Created Date), but Created Date'="01/26/2009 1:45:33 PM" compares the value of a field with a constant (an actual date).

When specifying a user, you can choose yourself by specifying "me". "me" is a symbolic user which refers to the currently logged in user. For example, you could create a relevance rule that specifies the Requirements field is visible only if the currently logged in user is one of the users defined in the multi-valued Stakeholders field.

Specifying Rules for Live and Versioned Document Model Items

If document versioning is enabled, you can specify conditions for live and versioned document model items. For example, you can create an event trigger rule to run on versioned items only or an e-mail notification rule that sends an e-mail when a user edits a specific live item.

With items, you can:

- define a rule to match live items only. For example, "(item is live)" matches live items only.
- define a rule to match versioned items only. For example, "(item is versioned)" matches versioned items only.

Note

As a best practice, PTC recommends including the (item is live) condition in all rules for live items. This improves the accuracy of rules.

With item IDs, you can:

- define a rule using a live item ID to match a single live item. For example, "((field[ID]=123) and (item is live))" matches 123.
- define a rule using a versioned item ID to match a single versioned item. For example, "(field[ID]=123-1.0)" matches 123-1.0.

Note

- You cannot define a rule using a live item ID to match the live item and all versions of the item.
- You cannot define a rule using live or versioned item IDs to match a range of live or versioned items, for example, "(field[ID]>123-1.0 and 128-1.0)".

General Options

Some CLI commands share the following general options.

- --devpath=*path*

  identifies the development path of a variant project. This is a label that was associated with a branch of the project by si createdevpath. Paths that include spaces must be enclosed by quotes. The following characters may not be used in a development path: \n, \r, \t, :, ['], #.

  Note

  This option is always used in conjunction with the -P option and a flat string project path. It cannot be used if you specify a project using a keyword string for the -P option. It is also mutually exclusive with the --projectRevision and --sandbox options.

- --changePackageId=*ID*

  identifies a change package that is notified of this action, for example, 1452:1. Note the following about using this option:

  - This option can only be specified if change packages are enabled.
  - You must specify this option if you have requested to obtain a lock and your administrator has set up locks to be tracked in change packages.
  - You must specify this option if your administrator has made change packages mandatory.
  - If your administrator has given you permission, you can bypass mandatory change packages by specifying --changePackageId=*:bypass*.
  - If change packages are enabled but it is not mandatory to specify a change package, or if no change package is applicable, you must specify --changePackageId=*:none*.

- --[no]failOnAmbiguousProject

  if you specify the project using a flat string for the -P option, this option displays an error message when multiple projects correspond to the specified path.

- --filter=*filteroptions*

  allows you to select members for all commands that take a list of members, using filteroptions, which can be one or more of the following:

  - archivesharedselects members that share another member's archive.
  - attribute*:name[=value]*

    selects members based on an attribute name and, optionally, value.
  - changed *[:working|:sync|:newer|:size|:missing|:newmem|:all]*

    selects changed members based on: changes to working files, those that are out of sync with the project, those where a newer revision exists in the project, or based on all changes.
  - rule *[:memberrevdiffers|:defined|:invalid]*

    selects members based on a revision rule filter. :memberrevdiffers selects all members for which the rule does not match the member revision. :defined selects all members with a revision rule. :invalid selects all members for which the rule does not expand to any existing revision.
  - file*:expression*

    selects members with a specific file name. This allows you to specify wild cards for file naming, such as the asterisk (*) to match any number of characters, and the question mark (?) to match a single character. For example, *.java or *RB.properties would be valid expressions.

- caseinsensitivefile*:expression*

  selects members with a specific case-insensitive file name. This allows you to specify wild cards for file naming, such as the asterisk (*) to match any number of characters, and the question mark (?) to match a single character. For example, *.java or *rb.properties would be valid expressions.
- frozen

  selects frozen members.
-
- label*[:name]*

  selects any member whose member revision has the specified label.
- anylabel*[:name]*

  selects any member that contains a revision that has the specified label.
- locked*[:name]*

  selects all locked members or those locked by a particular user.
- locktype*[:exclusive|:nonexclusive|:any]*

  selects members that are locked with the specified lock type. If no lock type or any is specified, all locked members are displayed.
- outofscope

  selects members that are outside of the scope definition of the Sandbox (if the Sandbox is a Scoped Sandbox). Specifying a Sandbox scope allows you to define what project members are included in a Sandbox, transferring specific members from the Integrity Server to the Sandbox directory when the Sandbox is created and controlling what members display in the Sandbox view.
- state*[:name]*

  selects members based on state.
- format*[:text|:binary]*

  selects members based on storage format.
- workingbranch

  selects members where the working file is on a branch from a given development path that is not the trunk development path.

Note

This filter applies only to sandboxes.

- deferred*[:add|:addfromarchive|:checkin|:drop|:import|:move|:rename|:updaterevision|:all]*

  selects deferred members based on: add, addfromarchive, checkin, drop, import, move, rename, updaterevision, or all operations.
- memberonbranch

  shows only members that are off the main development trunk.
- unresolvedmerges

  selects members affected by unresolved merges.
- pending*[:add|:addfromarchive|:drop|:import|:movememberfrom|:movememberto|:renamefrom|:renameto|:update|:updaterevision|:all]*

  selects pending members based on add, addfromarchive, drop, import, movememberfrom, movememberto, renamefrom, renameto, update, updaterevision, or all operations.
- workinprogress

  combines the deferred (all), locked (all), and changed (all) filters to select members that are considered work in progress.
- sparsecontents

  shows only existing working files and deferred operations in a sparse sandbox.

Using commas between the *filteroptions* serves to build logical "OR" statements between them, allowing you to create powerful filters. You may also specify multiple --filter=*filteroptions* on the command line, which effectively creates logical "AND" statements between them.

For example, you can resynchronize all modified JAVA files through:

```
si resync --filter=changed --filter=file:*.java
```

or you can resynchronize all files with label a or b through:

```
si resync --filter=label:a,label:b
```

You can also negate a filter using the ! character.

For example, you can check out all JAVA files that are not labelled Beta by typing:

```
si co --filter=file:*.java --filter=!label:Beta
```

- --hostname=*server*

identifies the name of the host server where the Integrity Server is located.

-
- --issueid=*id*

for configuration management commands, specifies the issue ID that corresponds to the change package that records the changes(s). This option can only be specified if the integration between configuration management, and workflow and document management is enabled. See your administrator for details on using the integration.

Note

The terms item and issue refer to the same object and are indistinguishable. Issue is a term embedded in legacy command and option names; therefore, item and issue are used interchangeably in the CLI documentation.

Note

If you have an issue assigned to you that contains only one open change package, you can specify the issue ID instead of the change package ID.

- --password=*password*

identifies the password to use for connecting to the Integrity Server.

-
- --port=*number*

identifies the port on the host server where the Integrity Server is located.

-
- -P *project*
- --project=*project*

specifies the path and name of a project. You can specify the project using a flat string or a keyword string. It is recommended that you use a keyword string, especially when you are writing scripts, since flat strings can be ambiguous as to which project is being specified. Use the following keywords to identify the project.

  - #value

#wp=value

#wproject=value

specifies the well-formed project or subproject name. Well-formed project or subproject names end with project.pj. For example, #/aurora_project/source_code. Do not specify a trailing project.pj.

- #p=value

  #project=value

  specifies the full name of the project, when it does not end with project.pj. For example, #p=/aurora_project/source_code/root.pj
- #value

  #ws=value

  #wsubs=value

  specifies the subproject in a well-formed project tree. A well-formed project tree has one subproject per directory. Using this keyword, you can specify several levels of subprojects at the same time. For example, #/aurora_project/source_code/#applications/savings_tool. Do not specify a trailing project.pj.
- #s=value

  #sub=value

  specifies the subproject in a poorly-formed project tree. A poorly-formed project tree has co-located subprojects or subprojects located more than one directory level deep. Using this keyword, you can only specify one subproject for each occurrence of the keyword. For example:

  #/aurora_project/source_code/#s=applications/savings_tool/project.pj#s=colocated.pj.

Note

The #s and # keywords do not interpret sub subprojects in the same way. For example, #/aurora_project#source_code/applications is not the same as #/aurora_project#s=source_code/applications/project.pj but is the same as #/aurora_project#s=source_code/project.pj#s=applications/project.pj

---

- #d=value

  #devpath=value

  specifies the development path name, for example, #/aurora_project/source_code/#applications/savings_tool#d=beta_variant. You can only jump to a variant for a subproject if the subproject is the root of the variant (the project rhough which the development path was created). You cannot jump to a variant if it differs from the closest variant higher in the project hierarchy (if there is a higher variant).

Note

This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

- #a=value

  specifies the project configuration as of a date (timestamp), for example, #p=aurora/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00". Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-*hours*:*minutes*. The following information illustrates North American timestamps recognized by Integrity:

  US GMT -5 (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, h is the time in hours, m is the time in minutes, s is the time in seconds, a is the AM or PM indicator, z is the timezone difference from Greenwhich Mean Time.

  EEEE, MMMM d, yyyy h:mm:ss a z | Monday, April 28, 2014 3:33:45 AM GMT-05:00

  EEEE, MMMM d, yyyy h:mm:ss a | Monday, April 28, 2014 3:33:45 AM

  EEEE, MMMM d, yyyy h:mm a | Monday, April 28, 2014 3:33 AM

  MMMM d, yyyy h:mm:ss a z | April 28, 2014 3:33:45 AM GMT-05:00

  MMMM d, yyyy h:mm:ss a | April 28, 2014 3:33:45 AM

  MMMM d, yyyy h:mm a | April 28, 2014 3:33 AM

  MMM d, yyyy h:mm:ss a z | Apr 28, 2014 3:33:45 AM GMT-05:00

  MMM d, yyyy h:mm:ss a | Apr 28, 2014 3:33:45 AM

  MMM d, yyyy h:mm a | Apr 28, 2014 3:33 AM

  M/d/yy h:mm:ss a z | 4/28/14 3:33:45 AM GMT-05:00

  M/d/yy h:mm:ss a | 4/28/14 3:33:45 AM

  M/d/yy h:mm a | 4/28/14 3:33 AM

  h:mm:ss a z | 3:33:45 AM GMT-05:00

  h:mm:ss a | 3:33:45 AM

  h:mm a | 3:33 AM

  EEEE, MMMM d, yyyy | Monday, April 28, 2014

  MMMM d, yyyy | April 28, 2014

  MMM d, yyyy | Apr 28, 2014

  M/d/yy | 4/28/14

  MMM d, yyyy - h:mm:ss a | Apr 28, 2014 - 3:33:45 AM

  MMM d, yyyy - h:mm a | Apr 28, 2014 - 3:33 AM

  Because keywords are processed from left to right, ensure the placement of the #a keyword with other keywords will provide the desired result. The following example

  -P#p=aurora/project.pj#s=sub/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00"

  specifies to start at aurora/project.pj as it is configured now, then move into its subproject sub/project.pj as it exists in that configuration, and then access that configuration as of April 28, 2014 3:33:45 AM GMT-05:00. The result is that sub/project.pj is the configuration it has now, but includes the project contents as of April 28. A potential consequence is that if sub/project.pj is currently configured as a variant subproject in the parent aurora/project.pj project, it is the variant project contents as of April 28 that is returned, even if sub/project.pj was configured as a mainline subproject in the parent aurora/project.pj on April 28. Instead, following example,

  -P#p=aurora/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00"#s=sub/project.pj

  specifies specifies to start at aurora/project.pj as it is configured now, then move into its subproject sub/project.pj as it exists in that configuration, and then access that configuration as of April 28, 2014 3:33:45 AM GMT-05:00.

Note

Project configurations specified using #a appear as date-based build Project views and date-based build Sandbox views in the GUI.

- #b=value

  #build=value

  specifies the number, label or symbolic of the revision, for example,

  #/aurora_project/source_code/#applications/savings_tool#b=:head

Note

Timestamps may also display in project paths that specify the #b keyword; for the date format, see the documentation for the #a keyword. The date may also appear in the form: ts=*timestamp.number*.

This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

- #n=

  #normal=

  specifies that the subproject is a normal subproject. Do not enter a value.

Note

This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

- #l=value

  #location=value

  specifies the absolute path of the target subproject (rather than the configuration path). This keyword can be used for commands where the subproject context is not needed and the subproject is not part of any configuration (si configuresubproject and si sharesubproject).

  Note the following about the use of keywords:

  - The order of the keywords is important. Keywords are processed from left to right to build the project specification.
  - If you need to specify a '#' or '=' symbol in a keyword value, specify the symbol twice ('##', '==').
  - If you are specifying a variant subproject, you must specify its path starting at the root of the variant project (the project through which the development path was created).
- `--projectRevision=`*rev*

  identifies a particular revision of a build project.

Note

This option cannot be used if you specify a project using a keyword string for the `-P` option. This option is also mututally exclusive with the `--devpath` option.

This option can also take the "asof:" identifier with a date, to return the project configuration as of a specific date. For example, --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00".

It is possible to specify the branch with the identifier, for example --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00"@1.3.1. Specifying the branch returns contents of the project at the specified date on the specified project branch.

It is possible to specify the development path with the identifier using the form asof:*date*:@:*devpath*, for example --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00"@:Release2. Specifying the developmenth path returns contents of the project at the specified date on the specified project development path.

If the branch or development path is not specified, the current branch for the specified project configuration is used.

The following are additional forms may be displayed for configuration paths, and are represented here for information purposes:

```
asof:ts=timestamp.number
 asof:ts=timestamp.number@branch
 asof:ts=timestamp.number@:devpath
```

- `-R`

  `--[no|confirm]recurse`

  controls whether to recursively apply this command to any subprojects; used in all commands which take a list of members.
- `-r` *rev*

  `--revision=`*rev*

  uses a specified revision for the member. `rev` can be a valid revision number or a label. You may also facilitate automation with special keyword identifiers, specified using a colon (:) prefix (except for the state keyword). Acceptable identifiers are:

  - :head

    identifies the head revision.
  - :member

    identifies the member revision.
  - :locked

    identifies locked revisions.
  - :master

    identifies the member revision in the master project. This option is only applicable to variant projects.
  - time:*timestamp*

    uses the most recent revision on any branch at the specified timestamp. For example, `-rtime:December 22, 2007 3:33:34 PM GMT-05:00`. Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-`hours:minutes`. The following examples illustrate North American and German timestamps recognized by Integrity:

    Example 1: US GMT -5 (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, h is the time in hours, m is the time in minutes, s is the time in seconds, a is Greenwhich Mean Time (GMT), z is the timezone difference from Greenwhich Mean Time.

    `EEEE, MMMM d, yyyy h:mm:ss a z | Wednesday, April 28, 2007 3:33:45 AM GMT-05:00`

    `EEEE, MMMM d, yyyy h:mm:ss a z | Wednesday, April 28, 2007 3:33:45 AM GMT-05:00`

    `EEEE, MMMM d, yyyy h:mm:ss a | Wednesday, April 28, 2007 3:33:45 AM`

    `EEEE, MMMM d, yyyy h:mm a | Wednesday, April 28, 2007 3:33 AM`

    `MMMM d, yyyy h:mm:ss a z | April 28, 2007 3:33:45 AM GMT-05:00`

    `MMMM d, yyyy h:mm:ss a z | April 28, 2007 3:33:45 AM GMT-05:00`

    `MMMM d, yyyy h:mm:ss a | April 28, 2007 3:33:45 AM`

    `MMMM d, yyyy h:mm a | April 28, 2007 3:33 AM`

    `MMM d, yyyy h:mm:ss a z | Apr 28, 2007 3:33:45 AM GMT-05:00`

    `MMM d, yyyy h:mm:ss a z | Apr 28, 2007 3:33:45 AM GMT-05:00`

    `MMM d, yyyy h:mm:ss a | Apr 28, 2007 3:33:45 AM`

    `MMM d, yyyy h:mm a | Apr 28, 2007 3:33 AM`

    `M/d/yy h:mm:ss a z | 4/28/04 3:33:45 AM GMT-05:00`

    `M/d/yy h:mm:ss a z | 4/28/04 3:33:45 AM GMT-05:00`

    `M/d/yy h:mm:ss a | 4/28/04 3:33:45 AM`

M/d/yy h:mm a | 4/28/04 3:33 AM

h:mm:ss a z | 3:33:45 AM GMT-05:00

h:mm:ss a z | 3:33:45 AM GMT-05:00

h:mm:ss a | 3:33:45 AM

h:mm a | 3:33 AM

EEEE, MMMM d, yyyy | Wednesday, April 28, 2007

MMMM d, yyyy | April 28, 2007

MMM d, yyyy | Apr 28, 2007

M/d/yy | 4/28/04

MMM d, yyyy - h:mm:ss a | Apr 28, 2007 - 3:33:45 AM

MMM d, yyyy - h:mm a | Apr 28, 2007 - 3:33 AM

Example 2: Germany CEST (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, H is the time in hours, m is the time in minutes, s is the time in seconds, Uhr 'z is Central European Summer Time (CEST).

EEEE, d. MMMM yyyy H.mm' Uhr 'z | Montag, 26. Juli 2007 22.26 Uhr CEST

EEEE, d. MMMM yyyy HH:mm:ss z | Montag, 26. Juli 2007 22:26:29 CEST

EEEE, d. MMMM yyyy HH:mm:ss | Montag, 26. Juli 2007 22:26:29

EEEE, d. MMMM yyyy HH:mm | Montag, 26. Juli 2007 22:26

d. MMMM yyyy H.mm' Uhr 'z | 26. Juli 2007 22.26 Uhr CEST

d. MMMM yyyy HH:mm:ss z | 26. Juli 2007 22:26:29 CEST

d. MMMM yyyy HH:mm:ss | 26. Juli 2007 22:26:29

d. MMMM yyyy HH:mm | 26. Juli 2007 22:26

dd.MM.yyyy H.mm' Uhr 'z | 26.07.2007 22.26 Uhr CEST

dd.MM.yyyy HH:mm:ss z | 26.07.2007 22:26:29 CEST

dd.MM.yyyy HH:mm:ss | 26.07.2007 22:26:29

dd.MM.yyyy HH:mm | 26.07.2007 22:26

dd.MM.yy H.mm' Uhr 'z | 26.07.04 22.26 Uhr CEST

dd.MM.yy HH:mm:ss z | 26.07.04 22:26:29 CEST

dd.MM.yy HH:mm:ss | 26.07.04 22:26:29

dd.MM.yy HH:mm | 26.07.04 22:26

H.mm' Uhr 'z | 22.26 Uhr CEST

HH:mm:ss z | 22:26:29 CEST

HH:mm:ss | 22:26:29

HH:mm | 22:26

EEEE, d. MMMM yyyy | Montag, 26. Juli 2007

d. MMMM yyyy | 26. Juli 2007

dd.MM.yyyy | 26.07.2007

dd.MM.yy | 26.07.04

MMM d, yyyy - h:mm:ss a | Jul 26, 2007 - 10:26:29 PM

MMM d, yyyy - h:mm a | Jul 26, 2007 - 10:26 PM

- timeonbranch:*timestamp*@*branchnumber*

  uses the most recent revision on a specific branch at a specific timestamp. `-rtimeonbranch:timestamp` uses the most recent revision on the branch where the member revision currently resides. For example, `-rtimeonbranch:December 22, 2007 3:33:34 PM GMT-05:00`. `-rtimeonbranch:timestamp@branchnumber` uses the most recent revision on the specified branch at the specified timestamp. For example, `-rtimeonbranch:December 22, 2007 3:33:34 PM GMT-05:00@1.5.1`. Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-`hours:minutes`. For timestamp examples, see the :time:`timestamp` option.

Note

Updating a revision by timestamp makes the most recent revision at the specified timestamp the member revision.

- :memberbranchtip

  identifies the tip revision on the member revision branch.
- :working

  identifies the working revision.
- :trunktip

  identifies the tip revision on the trunk.
- state:*statename*

  identifies the state, for example, `Beta`. This option is useful when you want to select revisions in a project that are in a specific state.

  For each project member, Integrity searches from the member revision on the development path to the root of the archive to find a revision that corresponds to the specified state. If the member revision is on a branch, Integrity starts from the tip revision and searches to the root of the archive; other branches in the archive are *not* searched. If no revision on the development path matches the specified state, the command fails, stating "Revision does not exist."
- devpath:*devpathname*

  identifies the development path. This keyword only operates on member commands.
- build:*revisionnumber*

  identifies the build revision number, which must be a valid project checkpoint number or project label in which a given member is contained. Must specify a registered project. This keyword only operates on member commands.
- :rule

  identifies a rule defined with the `si setmemberrule` command.
-
- link:p=*project*[::d=*devpath*][::m=*member*][::recurse][::b=*buildrevnumber*]

  allows you to set the member revision to whatever is the member revision for the corresponding member in a specific external project configuration (normal, variant, build). Links the project that the member belongs to (the target project) with the master project where:

  - `project` is the master project
  - `devpath` is the development path for the master project
  - `member` is a member in the target project. If not provided, the project is searched for a member with the same backing archive. If recurse is specified, the search is recursive throughout the subprojects. There must be exactly one backing archive for each member.

A possible application is to update all members to the same revision, even if they do not have the same backing archive.

- `-S sandbox`

  `--sandbox=sandbox`

  specifies the location of a Sandbox. In some cases, the commands that take this option do something with the Sandbox contents themselves. In other cases, specifying the Sandbox location is simply a way to locate, or "point to", the corresponding project file. This option is mutually exclusive with `-P project|--project =project`.

Note

Locations that include spaces must be enclosed by quotes.

- `--user=name`

  identifies the user to use for connecting to the Integrity Server.

- `--devpath=path`

  identifies the development path of a variant project. This is a label that was associated with a branch of the project by `si createdevpath`. Paths that include spaces must be enclosed by quotes. The following characters may not be used in a development path: \n, \r, \t, :, ['], #.

Note

This option is always used in conjunction with the `-P` option and a flat string project path. It cannot be used if you specify a project using a keyword string for the `-P` option. It is also mutually exclusive with the `--projectRevision` and `--sandbox` options.

- `--changePackageId=ID`

  identifies a change package that is notified of this action, for example, `1452:1`. Note the following about using this option:

  - This option can only be specified if change packages are enabled.
  - You must specify this option if you have requested to obtain a lock and your administrator has set up locks to be tracked in change packages.
  - You must specify this option if your administrator has made change packages mandatory.
  - If your administrator has given you permission, you can bypass mandatory change packages by specifying `--changePackageId=:bypass`.
  - If change packages are enabled but it is not mandatory to specify a change package, or if no change package is applicable, you must specify `--changePackageId=:none`.

- `--[no]failOnAmbiguousProject`

  if you specify the project using a flat string for the `-P` option, this option displays an error message when multiple projects correspond to the specified path.

- `--filter=filteroptions`

  allows you to select members for all commands that take a list of members, using `filteroptions`, which can be one or more of the following:

  - `archivesharedselects` members that share another member's archive.
  - `attribute:name[=value]`

    selects members based on an attribute name and, optionally, value.
  - `changed [:working|:sync|:newer|:size|:missing|:newmem|:all]`

    selects changed members based on: changes to working files, those that are out of sync with the project, those where a newer revision exists in the project, or based on all changes.
  - `rule [:memberrevdiffers|:defined|:invalid]`

    selects members based on a revision rule filter. `:memberrevdiffers` selects all members for which the rule does not match the member revision. `:defined` selects all members with a revision rule. `:invalid` selects all members for which the rule does not expand to any existing revision.
  - `file:expression`

    selects members with a specific file name. This allows you to specify wild cards for file naming, such as the asterisk (*) to match any number of characters, and the question mark (?) to match a single character. For example, `*.java` or `*RB.properties` would be valid expressions.
  - `caseinsensitivefile:expression`

    selects members with a specific case-insensitive file name. This allows you to specify wild cards for file naming, such as the asterisk (*) to match any number of characters, and the question mark (?) to match a single character. For example, `*.java` or `*rb.properties` would be valid expressions.
  - `frozen`

    selects frozen members.
  -
  - `label[:name]`

    selects any member whose member revision has the specified label.
  - `anylabel[:name]`

    selects any member that contains a revision that has the specified label.
  - `locked[:name]`

    selects all locked members or those locked by a particular user.
  - `locktype[:exclusive|:nonexclusive|:any]`

    selects members that are locked with the specified lock type. If no lock type or `any` is specified, all locked members are displayed.
  - `outofscope`

    selects members that are outside of the scope definition of the Sandbox (if the Sandbox is a Scoped Sandbox). Specifying a Sandbox scope allows you to define what project members are included in a Sandbox, transferring specific members from the Integrity Server to the Sandbox directory when the Sandbox is created and controlling what members display in the Sandbox view.
  - `state[:name]`

    selects members based on state.
  - `format[:text|:binary]`

    selects members based on storage format.
  - `workingbranch`

    selects members where the working file is on a branch from a given development path that is not the trunk development path.

    Note

    This filter applies only to sandboxes.

  - `deferred[:add|:addfromarchive|:checkin|:drop|:import|:move|:rename|:updaterevision|:all]`

    selects deferred members based on: add, addfromarchive, checkin, drop, import, move, rename, updaterevision, or all operations.
  - `memberonbranch`

    shows only members that are off the main development trunk.
  - `unresolvedmerges`

    selects members affected by unresolved merges.

- pending*[:add|:addfromarchive|:drop|:import|:movememberfrom|:movememberto|:renamefrom|:renameto|:update|:updaterevision|:all]*

  selects pending members based on add, addfromarchive, drop, import, movememberfrom, movememberto, renamefrom, renameto, update, updaterevision, or all operations.
- workinprogress

  combines the deferred (all), locked (all), and changed (all) filters to select members that are considered work in progress.
- sparsecontents

  shows only existing working files and deferred operations in a sparse sandbox.

Using commas between the *filteroptions* serves to build logical "OR" statements between them, allowing you to create powerful filters. You may also specify multiple --filter=*filteroptions* on the command line, which effectively creates logical "AND" statements between them.

For example, you can resynchronize all modified JAVA files through:

```
si resync --filter=changed --filter=file:*.java
```

or you can resynchronize all files with label a or b through:

```
si resync --filter=label:a,label:b
```

You can also negate a filter using the ! character.

For example, you can check out all JAVA files that are not labelled Beta by typing:

```
si co --filter=file:*.java --filter=!label:Beta
```

- --hostname=*server*

identifies the name of the host server where the Integrity Server is located.

-
- --issueid=*id*

for configuration management commands, specifies the issue ID that corresponds to the change package that records the changes(s). This option can only be specified if the integration between configuration management, and workflow and document management is enabled. See your administrator for details on using the integration.

Note

The terms item and issue refer to the same object and are indistinguishable. Issue is a term embedded in legacy command and option names; therefore, item and issue are used interchangeably in the CLI documentation.

Note

If you have an issue assigned to you that contains only one open change package, you can specify the issue ID instead of the change package ID.

- --password=*password*

identifies the password to use for connecting to the Integrity Server.

-
- --port=*number*

identifies the port on the host server where the Integrity Server is located.

-
- -P *project*
- --project=*project*

specifies the path and name of a project. You can specify the project using a flat string or a keyword string. It is recommended that you use a keyword string, especially when you are writing scripts, since flat strings can be ambiguous as to which project is being specified. Use the following keywords to identify the project.

  - #value

    #wp=value

    #wproject=value

    specifies the well-formed project or subproject name. Well-formed project or subproject names end with project.pj. For example, #/aurora_project/source_code. Do not specify a trailing project.pj.
  - #p=value

    #project=value

    specifies the full name of the project, when it does not end with project.pj. For example, #p=/aurora_project/source_code/root.pj
  - #value

    #ws=value

    #wsubs=value

    specifies the subproject in a well-formed project tree. A well-formed project tree has one subproject per directory. Using this keyword, you can specify several levels of subprojects at the same time. For example, #/aurora_project/source_code/#applications/savings_tool. Do not specify a trailing project.pj.
  - #s=value

    #sub=value

    specifies the subproject in a poorly-formed project tree. A poorly-formed project tree has co-located subprojects or subprojects located more than one directory level deep. Using this keyword, you can only specify one subproject for each occurrence of the keyword. For example:

    #/aurora_project/source_code/#s=applications/savings_tool/project.pj#s=colocated.pj.

    Note

    The #s and # keywords do not interpret sub subprojects in the same way. For example, #/aurora_project#source_code/applications is not the same as #/aurora_project#s=source_code/applications/project.pj but is the same as #/aurora_project#s=source_code/project.pj#s=applications/project.pj

    ---
  - #d=value

    #devpath=value

    specifies the development path name, for example, #/aurora_project/source_code/#applications/savings_tool#d=beta_variant. You can only jump to a variant for a subproject if the subproject is the root of the variant (the project rhough which the development path was created). You cannot jump to a variant if it differs from the closest variant higher in the project hierarchy (if there is a higher variant).

    Note

    This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

  - #a=value

    specifies the project configuration as of a date (timestamp), for example, #p=aurora/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00". Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-*hours*:*minutes*. The following information illustrates North American timestamps recognized by Integrity:

US GMT -5 (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, h is the time in hours, m is the time in minutes, s is the time in seconds, a is the AM or PM indicator, z is the timezone difference from Greenwhich Mean Time.

`EEEE, MMMM d, yyyy h:mm:ss a z | Monday, April 28, 2014 3:33:45 AM GMT-05:00`

`EEEE, MMMM d, yyyy h:mm:ss a | Monday, April 28, 2014 3:33:45 AM`

`EEEE, MMMM d, yyyy h:mm a | Monday, April 28, 2014 3:33 AM`

`MMMM d, yyyy h:mm:ss a z | April 28, 2014 3:33:45 AM GMT-05:00`

`MMMM d, yyyy h:mm:ss a | April 28, 2014 3:33:45 AM`

`MMMM d, yyyy h:mm a | April 28, 2014 3:33 AM`

`MMM d, yyyy h:mm:ss a z | Apr 28, 2014 3:33:45 AM GMT-05:00`

`MMM d, yyyy h:mm:ss a | Apr 28, 2014 3:33:45 AM`

`MMM d, yyyy h:mm a | Apr 28, 2014 3:33 AM`

`M/d/yy h:mm:ss a z | 4/28/14 3:33:45 AM GMT-05:00`

`M/d/yy h:mm:ss a | 4/28/14 3:33:45 AM`

`M/d/yy h:mm a | 4/28/14 3:33 AM`

`h:mm:ss a z | 3:33:45 AM GMT-05:00`

`h:mm:ss a | 3:33:45 AM`

`h:mm a | 3:33 AM`

`EEEE, MMMM d, yyyy | Monday, April 28, 2014`

`MMMM d, yyyy | April 28, 2014`

`MMM d, yyyy | Apr 28, 2014`

`M/d/yy | 4/28/14`

`MMM d, yyyy - h:mm:ss a | Apr 28, 2014 - 3:33:45 AM`

`MMM d, yyyy - h:mm a | Apr 28, 2014 - 3:33 AM`

Because keywords are processed from left to right, ensure the placement of the #a keyword with other keywords will provide the desired result. The following example

-P#p=aurora/project.pj#s=sub/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00"

specifies to start at aurora/project.pj as it is configured now, then move into its subproject sub/project.pj as it exists in that configuration, and then access that configuration as of April 28, 2014 3:33:45 AM GMT-05:00. The result is that sub/project.pj is the configuration it has now, but includes the project contents as of April 28. A potential consequence is that if sub/project.pj is currently configured as a variant subproject in the parent aurora/project.pj project, it is the variant project contents as of April 28 that is returned, even if sub/project.pj was configured as a mainline subproject in the parent aurora/project.pj on April 28. Instead, following example,

-P#p=aurora/project.pj#a="April 28, 2014 3:33:45 AM GMT-05:00"#s=sub/project.pj

specifies specifies to start at aurora/project.pj as it is configured now, then move into its subproject sub/project.pj as it exists in that configuration, and then access that configuration as of April 28, 2014 3:33:45 AM GMT-05:00.

Note

Project configurations specified using #a appear as date-based build Project views and date-based build Sandbox views in the GUI.

- `#b=value`

  `#build=value`

  specifies the number, label or symbolic of the revision, for example,

  #/aurora_project/source_code/#applications/savings_tool#b=:head

---

Note

Timestamps may also display in project paths that specify the #b keyword; for the date format, see the documentation for the #a keyword. The date may also appear in the form: ts=*timestamp.number*.

This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

- #n=

  #normal=

  specifies that the subproject is a normal subproject. Do not enter a value.

Note

This keyword is not supported on Sandbox commands that specify subproject scope using a Sandbox-relative source configuration path.

- #l=value

  #location=value

  specifies the absolute path of the target subproject (rather than the configuration path). This keyword can be used for commands where the subproject context is not needed and the subproject is not part of any configuration (si configuresubproject and si sharesubproject).

  Note the following about the use of keywords:

  - The order of the keywords is important. Keywords are processed from left to right to build the project specification.
  - If you need to specify a '#' or '=' symbol in a keyword value, specify the symbol twice ('##', '==').
  - If you are specifying a variant subproject, you must specify its path starting at the root of the variant project (the project through which the development path was created).
- `--projectRevision=`*rev*

  identifies a particular revision of a build project.

Note

This option cannot be used if you specify a project using a keyword string for the `-P` option. This option is also mututally exclusive with the `--devpath` option.

This option can also take the "asof:" identifier with a date, to return the project configuration as of a specific date. For example, --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00".

It is possible to specify the branch with the identifier, for example --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00"@1.3.1. Specifying the branch returns contents of the project at the specified date on the specified project branch.

It is possible to specify the development path with the identifier using the form asof:*date*:@:*devpath*, for example --projectrevision asof:"April 28, 2014 3:33:45 AM GMT-05:00"@:Release2. Specifying the developmenth path returns contents of the project at the specified date on the specified project development path.

If the branch or development path is not specified, the current branch for the specified project configuration is used.

The following are additional forms may be displayed for configuration paths, and are represented here for information purposes:

```
asof:ts=timestamp.number
 asof:ts=timestamp.number@branch
 asof:ts=timestamp.number@:devpath
```

- `-R`

  `--[no|confirm]recurse`

  controls whether to recursively apply this command to any subprojects; used in all commands which take a list of members.
- `-r rev`

  `--revision=rev`

  uses a specified revision for the member. `rev` can be a valid revision number or a label. You may also facilitate automation with special keyword identifiers, specified using a colon (:) prefix (except for the state keyword). Acceptable identifiers are:
  - :head

    identifies the head revision.
  - :member

    identifies the member revision.
  - :locked

    identifies locked revisions.
  - :master

    identifies the member revision in the master project. This option is only applicable to variant projects.
  - time:*timestamp*

    uses the most recent revision on any branch at the specified timestamp. For example, `-rtime:December 22, 2007 3:33:34 PM GMT-05:00`. Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-`hours:minutes`. The following examples illustrate North American and German timestamps recognized by Integrity:

    Example 1: US GMT -5 (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, h is the time in hours, m is the time in minutes, s is the time in seconds, a is Greenwhich Mean Time (GMT), z is the timezone difference from Greenwhich Mean Time.

    ```
    EEEE, MMMM d, yyyy h:mm:ss a z | Wednesday, April 28, 2007 3:33:45 AM GMT-05:00
    EEEE, MMMM d, yyyy h:mm:ss a z | Wednesday, April 28, 2007 3:33:45 AM GMT-05:00
    EEEE, MMMM d, yyyy h:mm:ss a | Wednesday, April 28, 2007 3:33:45 AM
    EEEE, MMMM d, yyyy h:mm a | Wednesday, April 28, 2007 3:33 AM
    MMMM d, yyyy h:mm:ss a z | April 28, 2007 3:33:45 AM GMT-05:00
    MMMM d, yyyy h:mm:ss a z | April 28, 2007 3:33:45 AM GMT-05:00
    MMMM d, yyyy h:mm:ss a | April 28, 2007 3:33:45 AM
    MMMM d, yyyy h:mm a | April 28, 2007 3:33 AM
    MMM d, yyyy h:mm:ss a z | Apr 28, 2007 3:33:45 AM GMT-05:00
    MMM d, yyyy h:mm:ss a z | Apr 28, 2007 3:33:45 AM GMT-05:00
    MMM d, yyyy h:mm:ss a | Apr 28, 2007 3:33:45 AM
    MMM d, yyyy h:mm a | Apr 28, 2007 3:33 AM
    M/d/yy h:mm:ss a z | 4/28/04 3:33:45 AM GMT-05:00
    M/d/yy h:mm:ss a z | 4/28/04 3:33:45 AM GMT-05:00
    M/d/yy h:mm:ss a | 4/28/04 3:33:45 AM
    M/d/yy h:mm a | 4/28/04 3:33 AM
    h:mm:ss a z | 3:33:45 AM GMT-05:00
    h:mm:ss a z | 3:33:45 AM GMT-05:00
    h:mm:ss a | 3:33:45 AM
    h:mm a | 3:33 AM
    EEEE, MMMM d, yyyy | Wednesday, April 28, 2007
    MMMM d, yyyy | April 28, 2007
    MMM d, yyyy | Apr 28, 2007
    M/d/yy | 4/28/04
    MMM d, yyyy - h:mm:ss a | Apr 28, 2007 - 3:33:45 AM
    MMM d, yyyy - h:mm a | Apr 28, 2007 - 3:33 AM
    ```

    Example 2: Germany CEST (where E is the day of the week, M is the month, d is the numerical day of the month, y is the year, H is the time in hours, m is the time in minutes, s is the time in seconds, Uhr 'z is Central European Summer Time (CEST).

    ```
    EEEE, d. MMMM yyyy H.mm' Uhr 'z | Montag, 26. Juli 2007 22.26 Uhr CEST
    EEEE, d. MMMM yyyy HH:mm:ss z | Montag, 26. Juli 2007 22:26:29 CEST
    EEEE, d. MMMM yyyy HH:mm:ss | Montag, 26. Juli 2007 22:26:29
    EEEE, d. MMMM yyyy HH:mm | Montag, 26. Juli 2007 22:26
    d. MMMM yyyy H.mm' Uhr 'z | 26. Juli 2007 22.26 Uhr CEST
    d. MMMM yyyy HH:mm:ss z | 26. Juli 2007 22:26:29 CEST
    d. MMMM yyyy HH:mm:ss | 26. Juli 2007 22:26:29
    d. MMMM yyyy HH:mm | 26. Juli 2007 22:26
    dd.MM.yyyy H.mm' Uhr 'z | 26.07.2007 22.26 Uhr CEST
    dd.MM.yyyy HH:mm:ss z | 26.07.2007 22:26:29 CEST
    dd.MM.yyyy HH:mm:ss | 26.07.2007 22:26:29
    dd.MM.yyyy HH:mm | 26.07.2007 22:26
    dd.MM.yy H.mm' Uhr 'z | 26.07.04 22.26 Uhr CEST
    dd.MM.yy HH:mm:ss z | 26.07.04 22:26:29 CEST
    dd.MM.yy HH:mm:ss | 26.07.04 22:26:29
    dd.MM.yy HH:mm | 26.07.04 22:26
    H.mm' Uhr 'z | 22.26 Uhr CEST
    HH:mm:ss z | 22:26:29 CEST
    HH:mm:ss | 22:26:29
    ```

HH:mm | 22:26

EEEE, d. MMMM yyyy | Montag, 26. Juli 2007

d. MMMM yyyy | 26. Juli 2007

dd.MM.yyyy | 26.07.2007

dd.MM.yy | 26.07.04

MMM d, yyyy - h:mm:ss a | Jul 26, 2007 - 10:26:29 PM

MMM d, yyyy - h:mm a | Jul 26, 2007 - 10:26 PM

- timeonbranch:*timestamp*@*branchnumber*

  uses the most recent revision on a specific branch at a specific timestamp. `-rtimeonbranch:timestamp` uses the most recent revision on the branch where the member revision currently resides. For example, `-rtimeonbranch:December 22, 2007 3:33:34 PM GMT-05:00`. `-rtimeonbranch:timestamp@branchnumber` uses the most recent revision on the specified branch at the specified timestamp. For example, `-rtimeonbranch:December 22, 2007 3:33:34 PM GMT-05:00@1.5.1`. Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-`hours:minutes`. For timestamp examples, see the :time:`timestamp` option.

Note

Updating a revision by timestamp makes the most recent revision at the specified timestamp the member revision.

- :memberbranchtip

  identifies the tip revision on the member revision branch.
- :working

  identifies the working revision.
- :trunktip

  identifies the tip revision on the trunk.
- state:*statename*

  identifies the state, for example, `Beta`. This option is useful when you want to select revisions in a project that are in a specific state.

  For each project member, Integrity searches from the member revision on the development path to the root of the archive to find a revision that corresponds to the specified state. If the member revision is on a branch, Integrity starts from the tip revision and searches to the root of the archive; other branches in the archive are *not* searched. If no revision on the development path matches the specified state, the command fails, stating "Revision does not exist."
- devpath:*devpathname*

  identifies the development path. This keyword only operates on member commands.
- build:*revisionnumber*

  identifies the build revision number, which must be a valid project checkpoint number or project label in which a given member is contained. Must specify a registered project. This keyword only operates on member commands.
- :rule

  identifies a rule defined with the `si setmemberrule` command.
-
- link:p=*project*[::d=*devpath*][::m=*member*][::recurse][::b=*buildrevnumber*]

  allows you to set the member revision to whatever is the member revision for the corresponding member in a specific external project configuration (normal, variant, build). Links the project that the member belongs to (the target project) with the master project where:

  - `project` is the master project
  - `devpath` is the development path for the master project
  - `member` is a member in the target project. If not provided, the project is searched for a member with the same backing archive. If recurse is specified, the search is recursive throughout the subprojects. There must be exactly one backing archive for each member.

  A possible application is to update all members to the same revision, even if they do not have the same backing archive.

- `-S` *sandbox*

  `--sandbox=`*sandbox*

  specifies the location of a Sandbox. In some cases, the commands that take this option do something with the Sandbox contents themselves. In other cases, specifying the Sandbox location is simply a way to locate, or "point to", the corresponding project file. This option is mutually exclusive with `-P project|--project =project`.

Note

Locations that include spaces must be enclosed by quotes.

- `--user=`*name*

  identifies the user to use for connecting to the Integrity Server.
- `--customFieldDefinition=`*value*

  specifies the definition of custom fields.

Note

To define a custom field on an item, a Custom Fields field must be added as a visible field on a type. The data type of a custom field can be *Integer*, *Floating Point*, *Date*, *Pick*, *Logical*, or *Short Text*.

The format to define a custom field is '`name=`*<name>*, `type=`*<integer|float|logical|date|shorttext|pick>*, `description=`*<description>*, `default=`*<default value>*, `values=`*<text=value>*, `newname=`*<new name>*'

`name`

specifies the name of the custom field.

`type`

specifies the data type of the custom field.

Note

After a custom field is defined, the user cannot change the type.

`description`

specifies the description of the custom field.

`default`

specifies the default value.

values

specifies the pick values in the format *text=value:text1=value1*. Text is an alphanumeric string and value is its associated integer value.

newname

specifies the new name for the field. This attribute cannot be specified while defining the custom field.

For example:

- im createissue *--customFieldDefinition="name=field1,type=integer,default=100,description=some sample desc"*

  creates a custom field: field1 of integer data type with default value set to 100
- im createissue*--customFieldDefinition="name=field2,type=pick,default=a,values=a=1:b=2:c=3,description=some sample desc"*

  creates a custom field: field2 of pick data type with pick values as a, b, c and having default value set to a.

  To specify more than one custom field definition, specify this option multiple times for each custom field that you want to add or modify on the issue.

  When there are multiple custom field definitions using the same name, the most recent one is displayed.

  When the name, description or any of the attributes are used multiple times in the same option, the most recent one is displayed. For example, *--customFieldDefinition=name=a10,type=shorttext --customFieldDefinition=name=a10,type=logical,*

  the field a10 with logical data type is displayed.

- --customFieldValue=*value*

specifies value to the Custom Fields field.

Note

To set the custom field values on an item for a custom field defined on a project, the field Custom Field Values must be added as a visible field on a type, and Project field must be set to a valid project.

The format to set a custom field value is described below:

'name=<name>,value=<value>'

name

specifies the name of the custom field defined on a project item.

value

specifies the value for the custom field in a specific format. For example:

im createissue *--customFieldValue=name=field1,value=200*

sets the value of field1 to "200".

im createissue *--customFieldValue=name=field2,value=c*

sets the value of field2 to "c".

To set multiple custom field values, specify this option multiple times for each Custom Field Value that you want to set on the issue.

- --field=*"Custom Fields=value"*

specifies creating multiple Custom Field values in a single command using the --field option.

'value' is 'name=*<name>*, type=*<integer|float|logical|date|shorttext|pick>*, description=*<description>*, newName=*<new name>*, default=*<default value>*, values=*<text=value>*:<text1=value1>'.

When a user specifies --field=*"Custom Fields"="<semi colon delimited custom field definitions>"*, the custom fields are added, or removed depending on following scenario:

When a user specifies all the custom field definitions on the item along with the new definitions that conflict with the existing custom fields, the existing custom field definitions are removed and the new custom field definitions are added.

For example, there are fields such as field1 of integer type, field2 of pick type, and field3 of date type existing on a server. Specify the option along with field4 of logical data type as, --field=*"Custom Fields"="name=field1,type=float,default=0.125;name=field2,type=shorttext,default=sampletext, description=test;name=field3,default=03/12/2016 10:12:11 PM;name=field4,type=logical,default=false"*

This creates new definitions field1, field2, field3 and field4 as specified. Since the fields are created, the type can be changed.

Note

When a user does not specify any custom field definitions, all the existing custom field definitions are removed.

- --[no|confirm]removeCustomFieldDefinitions

controls whether to allow removal of custom field definitions. For example:

im editissue --field=*"CustomFields=name=c10,type=integer" --removeCustomFieldDefinitions*

This removes all the custom fields other than c10 without asking for any confirmation.

im editissue --removeCustomFieldDefinition=*c10* --removeCustomFieldDefinition=*c11* --removeCustomFieldDefinitions

This removes the custom fields c10 and c11 without asking for any confirmation.

- --[no|confirm]removeCustomFieldPickValues

controls whether to allow removal of pick values for the custom fields.

For example, im editissue --field=*"Custom Fields=name=c10,type=pick,values="* --removeCustomFieldPickValues

This removes all the pick values of the custom field c10 without asking for any confirmation.

im editissue --customFieldDefinition=*name=c10*,values= --removeCustomFieldPickValues

This removes the pick values from custom field c10 without asking for any confirmation.

- --field="Custom Field Values=*value*"

where value is 'value' is 'name=*<name>*, value=*<field value>*

specifies creating multiple custom field values in a single command using the --field option.

Note

User can perform this operation by specifying the --customFieldValue option multiple times in a single command.

When a user specifies --field=*"Custom Field Values=<semi colon delimited custom field values>"*, the existing custom field values under the field Custom Field Values are added, or removed depending on following scenario:

User specifies all the custom field values along with the new field values, then the new values are set and the other values are cleared. For example, there are fields such as field1 of integer type, field2 of pick type, and field3 of date type existing on the server, and there is field4 of logical type which is not set

on server. Specify the option as, `--field="Custom Field Values"="name=field4,value=true"`

This sets the value for field4 to "true".

Note

User does not specify any of the existing Custom Field Values, then all the values are cleared.

- `--removeCustomFieldDefinition=name`

  removes the custom fields defined on a project backing item.

Note

To remove a custom field from an item field, the Custom Field must be visible on the type.

- `--[no|confirm]removeCustomFieldValues`

  specifies removal of Custom Field Values.

  For example, `im editissue --field="Custom Field Values=name=c10,value=123" --removeCustomFieldValues`

  This confirms the removal of all the field values of the Custom Fields fields other than c10.

Note

You must enclose the value for custom field in double-quotes so that it is parsed properly.

- `--[no|confirm]projectChangeRemoveCustomField`

  specifies confirmation to remove the custom fields or the custom field values when the project is changed.

Universal Options

The following universal options apply to all CLI commands.

- `--[no]batch`controls batch mode. Batch mode forces the application to process commands without prompting for responses.
- `--cwd=`*`directory`*

  acts as if the command is executed in the specified directory. In particular, any files and members in the selection are treated as being relative to that directory.

  Suppose you are working in the `c:\sandbox` directory and you want to issue the check out command so that the implicit Sandbox selection will work in a subdirectory, rather than having to specify the complete path for subdirectory Sandbox names. You could use the `--cwd` option to do this, for example:

  ```
  si co --cwd=./demoapp/controls demoappctrl.c
  ```

  makes Integrity work in the `c:\sandbox\demoapp\controls` directory and follows implicit Sandbox selection rules from there to find the appropriate Sandbox, then checks out the `demoappctrl.c` file.
- `-F` *`file`*

  `--selectionFile=`*`file`*

  provides an alternative way to specify the selection. The specified `file` is a text file containing a list of file names, members, projects, or sandboxes, one per line. The command operates on all the listed files.

Note

The `--selectionFile` option is only relevant for commands that have selections and can only be used to specify the command selection (not command options). Be careful to avoid duplications. In some cases if a file, member, project or Sandbox is listed twice in the `file`, the command may report an error.

- `--forceConfirm=`*`[yes|no]`*

  `-N`

  `--no`

  `-Y`

  `--yes`

  controls the responses of either "yes" or "no" to all prompts. Specifying "yes" or "no" can be an easy way to accomplish the same thing as specifying other command options with `[no|confirm]` prefixes, for example the `--[no|confirm]overwriteChanged` option in the `si co`, `si resync`, and `si revert` commands. Specifying `--yes` or `--no` accomplishes the same thing for `--overwriteChanged` and `--nooverwriteChanged`, but further responds "yes" or "no" to all other questions asked.

Note

Be careful to use specific options if you want variations in your responses to prompts. The

`--yes` and `--no` options in particular are wide-ranging types of responses and should be used only in rare circumstances.

- `-g`

  `--gui`

  allows user interaction to happen through the GUI (graphical user interface).
- `--width`

  controls the width in pixels of the graphical user interface.
- `--height`

  controls the height in pixels of the graphical user interface.
- `-x`

  specifies the x location in pixels of the graphical user interface window.
- `-y`

  specifies the y location in pixels of the graphical user interface window.
- `--quiet`

  controls the status display to silence most information messages.
- `--settingsUI=`*`[gui|default]`*

  controls the GUI for command options.
- `--status=`*`[none|gui|default]`*

  controls the status display.
- `-?`

`--usage`

shows usage for the command.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Miscellaneous: ACL, diagnostics, preferences

preferences

preferences applicable to the setprefs and viewprefs commands

Description

The `setprefs` and `viewprefs` commands refer to the same group of commands and preference keys for the Integrity component you are configuring (`si`, `im`, `aa`, `integrity`). This reference page is provided as a guide to the preferences you can configure. To see each component's specific keys, simply append the command to the `viewprefs` command. For example,

```
si viewprefs --command=co
```

displays all of the view preferences for the `si co` command.

Preference Keys

The preference keys you can specify for the above commands are often similar, and some are global preferences such that when you change it for one command, it changes for all. The following are some common preference keys:

- `allowImplicitIdentification`

  controls whether to allow the implicit identification and selection of Sandboxes. This means that you can use commands without explicitly specifying a `-S` Sandbox option, and Integrity determines the Sandbox to operate on based on the directory you're working in. Valid values are `true` or `false`.
- `cwd`

  identifies a working directory for the command. In particular, any relative files and members in the selection are treated as being in that directory.
- `command.batchMode`

  a global key that controls batch mode. Batch mode forces the application to process commands without prompting for responses. Valid values are `false`, `true`.
-
- `developmentPath`

  specifies a particular development path to use with the command all the time.
-
- `filter`

  specifies a filter to use with the command all the time.
-
- `forceConfirm`

  specifies whether to force the application to request confirmation for the particular command.
-
- `includeFormers`

  specifies whether to include members that have been dropped from the project.
-
- `projectName`

  specifies a particular Integrity project to work with all the time.
- `projectRevision`

  specifies a particular Integrity configuration management project checkpoint to work with all the time.
- `recurse`

  controls whether to recurse into subprojects.
- `sandbox.allowProjectSelection`

  a global key that controls whether to allow Integrity configuration management project selection. Valid values are `false`, `true`.
- `sandbox.allowSandboxSelection`

  a global key that controls whether to allow Sandbox selection. Valid values are `false`, `true`.
- `sandboxName`

  specifies a particular Sandbox to be used all the time.
- `selectionFile`

  specifies a file that lists member files to work with.
- `server.credential`

  a global key that identifies the credential, or password, for logging into the Integrity Server.
- `server.hostname`

  a global key that identifies the hostname for the Integrity Server.
- `server.port`

  a global key that identifies the port for the Integrity Server.
- `server.user`

  a global key that identifies the user name for logging into the Integrity Server.
- `socksProxyHost`

  a global key that identifies the SOCKS proxy host.
- `socksProxyPort`

  a global key that identifies the SOCKS proxy port.
- `status.popupDelay`

  a global key that controls the duration, in milliseconds (ms), that the commands status appears in the graphical user interface or the command line interface.
- `swing.lookAndFeel`

  a global key that controls the "look and feel" appearance of the graphical user interface. Valid values are `System`, `Windows`, `Motif`, `Metal`.

See Also

- Miscellaneous: diagnostics, options

si intro

introduction to reference pages

Description

A description of an individual topic (for example, a command) is loosely called the reference page for that topic, even if it is actually several pages long.

There are three alternatives for accessing the reference pages to each Integrity configuration management command through the CLI man command.

First, you may simply type the `si` prefix and the command together as one word. Second, you may type the `si` prefix and the command with an underscore between them. Third, you may quote the `si` prefix and the command, with a space in the middle. For example:

```
man siabout
```

```
man si_about
```

```
man "si about" (Windows client only)
```

See the reference page for the man command itself, by typing `man man`, to find out more details.

This reference page describes the parts of a reference page with examples taken from real Integrity reference pages.

The following sections discuss the various elements of a reference page.

Name

The Name section provides the name of the command and a brief functional description.

Synopsis

In the reference page for a command, the Synopsis section provides a quick summary of the command's format. For example, here is the synopsis of the si add command.

```
si add [--archive=filename] [--author=name] [(--binaryFormat|--textFormat)] [(--cpid=ID|--changePackageId=ID)] [--[no]defer] [--
[no|confirm]closeCP] [--issueId=ID] [--description=desc] [--descriptionFile=file] [--issueId=ID] [(-l|--lock)] [--onExistingArchive=
[confirm|sharearchive|newarchive|cancel]] [--[no]createSubprojects] [--[no]retainWorkingFile] [--[no]saveTimestamp] [--[no]unexpand]
[(-r rev|--revision=rev)] [(-R |--[no|confirm]recurse)] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--
password=password] [--user=name] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-g|--gui)] [(-N|--
no)] [--[no]batch] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)] [-R|--
[no|confirm]recurse] [--[no|confirm]includeFormers] [--exclude=file:pattern,dir:pattern...] [--include=file:pattern,dir:pattern...]
nonmember...
```

The synopsis takes the form of a command line as you might type it into the system; it shows what you can type in and the order in which you should do it. The parts that are enclosed in square brackets are optional; you may omit them if you choose. Parts that are not enclosed in square brackets must be present for the command to be correct.

The synopsis begins with the name of the command itself. The Integrity configuration management commands all include the *si* prefix. In Integrity documentation, command names are always written in bold Courier font.

After the command name comes a list of options. A typical Integrity command option consists of either a single dash (-) followed by a single character, usually an uppercase or lowercase letter, or it may consist of a double dash (--) followed by a multi-character option name. Often there are single-character and multi-character options that do the same thing. The multi-character strings are not case sensitive, but are shown in mixed case to facilitate readability. For example, you might have `-L` or `--label`.

The synopsis line shows options in bold Courier font. Note that the case of single-character options is important; for example, in the synopsis of si add, `-r` and `-R` are different options, with different effects.

Furthermore, in the synopsis all options are shown in one long string. Other common option forms are:

`-r rev`

`--revision=rev`

where *rev* or, in some cases, *value* provides extra information for using that option. For example, the si lock command locks Sandbox members; here's the command's synopsis:

```
si lock [--[no|confirm]branch] [(--cpid=ID|--changePackageId=ID)] [--issueId=ID] [(-r rev|--revision=rev)] [(-R|--
[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--
hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-
Y|--yes)] [--[no]batch] [--[no]recordAsWorkInProgress] [--[no|confirm]revisionMismatchIsError] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

In this example, there is the option

`-r rev`

This option tells the si lock command which revision to lock. In a command synopsis, anything appearing in italics is a placeholder for information that you are expected to supply. Sometime after the synopsis, the reference page explains what kind of information is expected in place of the placeholder.

The end of the si lock synopsis is
member...

Since there are no square brackets around the list, in this example, it is mandatory.

This means one or more member names; the ellipsis (`...`) stands for repetitions of whatever immediately precedes it. Most Integrity commands allow you to specify lists of multiple items using spaces between them.

See the options reference page for more details on selecting members, Sandboxes and projects.

The order of items on the command line is important. When you type in a command line, you should specify the parts of the command line in the order they appear in the command synopsis. The exceptions to this are options marked with `a -` or `a --`; they do not have to be given in the exact order shown in the synopsis. However, all the - or -- options must appear in the correct area of the command line. For example, you can specify

```
si lock -r 1.2 --gui member1
si lock --gui -r 1.2 member1
```

but you will not get correct results if you specify

```
si lock member1 -r 1.2 --gui
si lock -r 1.2 member1 --gui
```

and so on.

Description

The Description section simply describes what the command does and how each option works.

Inside the Description section, the names of files and directories are written in normal Courier font. The names of environment variables are written in italic Courier font.

Diagnostics

The Diagnostics division contains information about the exit status returned by the command. You can test this status to determine the result of the operation the command was asked to perform.

See Also

The See Also section refers to other reference pages that may contain information relevant to the reference page you have just read.

si about

displays product information

Synopsis

`si about [--[no]batch] [--cwd=`*`directory`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [(-?|--usage)]`

Description

`si about` displays version information for the Integrity release, build, service pack (if installed), API, and HotFixes (if installed).

Options

`si about` takes a subset of the universal options available to `si` commands. For example,

```
si about --gui
```

displays the product information in a GUI window. See the options reference page for descriptions.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Miscellaneous: diagnostics, options, preferences

si acceptcp

accepts change package

Synopsis

`si acceptcp [--comments=value] [--commentsFile=value] [--reviewer=[user|group:<Group-Name>|super|all]] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue|issue:change package id...`

Description

`si acceptcp` casts an accept vote on a change package under review and in the `submitted state`.

After a change package is submitted, each individual reviewer and one member of each reviewer group (if specified) in the reviewer list must accept the change package before it can be committed to the repository and then closed. For example:

```
si acceptcp --reviewer=user 3433:1
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the `options` reference page for descriptions.

- `--comments=value`

  specifies comments recorded in the review log with the vote.
- `--commentsFile=value`

  specifies a text file that contains the comments to be recorded in the review log with the vote.
- `--reviewer=[user|group:<Group-Name>|super|all]`

  specifies the capacity in which you are casting an accept vote.
- *user*

  casts a vote as an individual reviewer in the reviewer list.
- *group:<Group-Name>*

  casts a vote on behalf of the entire group (only one user from a group is necessary to vote on behalf of the entire group).
- *super*

  an overriding accept vote that is sufficient for accepting the change package even if there are additional reviewers. A super reviewer is not required to be a listed reviewer for the change package.
- *all*

  casts a vote as a specific user and any group to which the reviewer belongs.
- *issue...*

  issue:change package id...

  `issue` identifies a specific issue that contains all submitted change packages that you want to cast accept votes for; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to cast an accept vote for; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si cpissues , si createcp, si drop, si lock, si viewcps, si rejectcp, si opencp, si discardcp
- Miscellaneous: ACL, diagnostics, options, preferences

si activatedevpath

activates an inactive development path

Synopsis

si activatedevpath [--devpath=*value*] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [--[no]failOnAmbiguousProject] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*]

Description

si activatedevpath enables you to activate a development path that has been marked as inactive using the si deactivatedevpath command. This command requires that you specify either the mainline project and a development path, or a variant project or variant sandbox.

The following is an example for activating a development path:

```
si activatedevpath --project=c:/Aurora_Program/bin/Libra/project.pj --devpath=DevStream
```

If the development path was restricted when it was deactivated, you must remove the restriction using the the si unrestrictproject command to allow users to make changes normally.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --devpath=*value*

  the name of the development path to activate.
- *--Pproject*
- --project=*project*
- the path and name of the target project. You must specify either this option or --sandbox; the two options are mutually exclusive.
- *--Ssandbox*
- --sandbox=*sandbox*

  the location of a sandbox. This option can be used to point to a project file. You must specify either this option or --project; the two options are mutually exclusive.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si createsandbox, si deactivatedevpath, si dropdevpath, si memberinfo, si projectinfo, si revisioninfo, si rlog
- Miscellaneous: ACL, diagnostics, options, preferences

si acv

Displays running commands

Synopsis

```
si acv [--[no]batch] [--cwd=directory] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [--height=value] [--width=value] [-x=value] [-y=value] [--forceConfirm=[yes|no]] [(-Y|--yes)] [(-N|--no)] [(-Ffile|--selectionFile=file)] [--[no]persist]
```

Description

`si acv` displays the currently running commands.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]persist` controls the persistance of CLI views.

See Also

- Miscellaneous: diagnostics, options

si add

adds a new member to a project

Synopsis

```
si add [--archive=filename] [--author=name] [(--binaryFormat|--textFormat)] [(--cpid=ID|--changePackageId=ID)] [--[no]defer] [--
[no|confirm]closeCP] [--issueId=ID] [(-ddesc|--description=desc)] [--descriptionFile=file] [(-l|--lock)] [--[no]createSubprojects] [-
-[no]createSubprojectsForEmptyDirectories] [--onExistingArchive=[confirm|cancel|sharearchive|newarchive]][--
onExistingOutOfTreeArchive=[confirm|cancel|sharearchive|newarchive]] [--[no]retainWorkingFile] [--[no]failOnAmbiguousProject] [--
cancelOnInconsistentLineTerminators] [--normalizeInconsistentLineTerminators] [--[no]saveTimestamp] [--[no]unexpand] [(-r rev|--
revision=rev)] [(-R|--[no|confirm]recurse)] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--
password=password] [--user=name] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-g|--gui)] [(-N|--
no)] [--[no]batch] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)] [-R|--
[no|confirm]recurse] [--[no|confirm]includeFormers] [--exclude=file:pattern,dir:pattern...] [--include=file:pattern,dir:pattern...]
nonmember...
```

Description

`si add` adds one or more nonmember files located in a Sandbox directory to a project.

For example, you can add a member from the `c:\apps\prototype\` directory by specifying:

```
si add --description="Prototype application header"
 -S prototype.pj header.c
```

adds the `header.c` file to the project as a new member with the description "Prototype application header". Since the member is added to the project on the Integrity Server, all other users who have Sandboxes pointing to the project can see the same new member. Integrity creates a history on the server for each newly added member that does not already have a history.

If you intend to add a dropped member (see si drop) from an archive located on the server, or to share the history of another member in a different project, then use the si addmemberfromarchive command.

If you are re-adding a dropped member (see si drop), an archive already exists for the member, and you must specify whether you want Integrity to associate the member with the existing archive or generate a new one.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--archive=`*`filename`*

  sets the archive file name, `filename`, containing the server-side path and name by which the newly added member will be archived. This must be an existing archive file - this option does not create a new archive. This is an advanced option that effectively specifies an archive path to override the default mapping, allowing a basic form of archive sharing between projects; its use is not recommended except by advanced Integrity administrators.
- `-r=`*`value`*
- `--revision=`*`value`*

  specifies the revision number for the new member. If you specify an existing archive using the `--archive` option, Integrity creates a branch, unless you specify `:head` or a revision that does not yet exist. If you specify `:head` and the current head revision is exclusively locked by another user, Integrity creates a branch. If you have an exclusive lock on the revision being branched, your lock is removed.
- `--author=`*`name`*

  specifies an author name.
- `--binaryFormat`
- `--textFormat`

  forces the storage of the member file to occur in binary format or text format. This option overrides the preferences settings that control default behavior in the application for storage format. When configured, this option may be overriden by a server-side policy setting controlling default storage format behavior. See the si setprefs and si viewprefs commands for more details on preferences.
- `--[no]defer`

  controls whether to delay the add operation in the project until the deferred operation is submitted. The operation in the Sandbox still takes place immediately.

  If the change package reviews are mandatory, specify the `--deferred` option to create a pending entry for this operation at the time of change package submission. If the `--deferred` option is not enabled, Integrity creates the pending entry at the completion of this procedure. When a deferred add member operation is submitted as part of a review, a pending member is created For more information, see the *PTC Integrity User Guide*.
- `--[no|confirm]closeCP`

  controls whether to close the associated change package.
  - `--nocloseCP` means do not close the change package.
  - `--confirmcloseCP` means ask before closing the change package.
  - `--closeCP` always closes the change package.
- `-l`
- `--lock`

  controls whether to lock the newly created revision. The lock type used is based on your locks policy. For information on your locks policy, contact your administrator.
- `--onExistingArchive=[confirm|cancel|sharearchive|newarchive]`

  controls whether to allow sharing of this member's history, or to create a new archive if there is already an existing in project archive for the member.
  - `--onExistingArchive=confirm` means always ask whether to share the archive, create a new archive, or cancel the operation.
  - `--onExistingArchive=cancel` means cancel the operation.
  - `--onExistingArchive=sharearchive` means share the archive.
  - `--onExistingArchive=newarchive` means create a new archive.
- `--onExistingOutOfTreeArchive=[confirm|cancel|sharearchive|newarchive]`

  controls whether to allow sharing of this member's history between projects, or to create a new archive if there is already an existing out of tree archive for the member.
  - `--onExistingArchive=confirm` means always ask whether to share the archive, create a new archive, or cancel the operation.
  - `--onExistingArchive=cancel` means cancel the operation.
  - `--onExistingArchive=sharearchive` means share the archive.
  - `--onExistingArchive=newarchive` means create a new archive.
- `--description=`*`desc`*

  specifies a description for the new archive; this setting and the `--descriptionFile` option are mutually exclusive. If specifying a *`nonmember`* that has an existing history, this description does not overwrite an existing description and is ignored.

Note

Descriptions that include spaces must be enclosed by quotes.

- `-d` *`desc`*
- `--descriptionFile=`*`file`*

  specifies a file for obtaining a description to apply to the new archive; this setting and the `--description` option are mutually exclusive. If specifying a

nonmember that has an existing history, this description does not overwrite an existing description and is ignored.

- `--[no]createSubprojects`

  controls whether to create subprojects for each subdirectory encountered when adding members. This option is commonly used where you anticipate working with a large directory structure, because multiple subprojects are easier to manage than many subdirectories within one project. For example, specifying:

  ```
  si add --createSubprojects --description="Button Application" \buttons\activebutton.c
  ```

  creates a subproject for the `\buttons` directory and adds the file `activebutton.c`. Without the `--createSubprojects` option, the file and its path simply would be added to the project in the current directory.

- `--[no]createSubprojectsForEmptyDirectories`

  controls whether to create subprojects for each empty subdirectory encountered when adding members (and specifying to recursion into directories); preserving the file system directory structure in a project context.

- `--[no]retainWorkingFile`

  controls whether to retain a working file in the Sandbox after adding the new member.

- `--[no]saveTimestamp`

  controls whether to set the revision timestamp to the modification date and time of the working file rather than the current date and time.

- `--[no|confirm]includeFormers`

  controls whether to include former members. Former members are members that have been dropped but whose working files still reside in the Sandbox directory.

- `--[no]unexpand`controls whether to unexpand keywords in the member file before checking it into the history. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

  ```
  $Author: Warner, Carrie (cwarner) $
  $CompanyInfo$
  $Date: 2015/11/30 11:49:36EST $ $Header: si_add.dita 1.2 2015/11/30 11:49:36EST Warner, Carrie (cwarner) Exp  $
  $Id: si_add.dita 1.2 2015/11/30 11:49:36EST Warner, Carrie (cwarner) Exp  $ $Locker:  $
  $Log: si_add.dita  $ $Revision: 1.2 $
  Revision 1.2 2015/11/30 11:49:36EST Warner, Carrie (cwarner)
  XML tagging fixes
  Revision 1.1 2015/10/29 10:24:45EDT Flett, David (dflett)
  Initial revision
  Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
  $Name:  $
  $ProjectLabel:  $
  $ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
  $ProjectSetting  $
  $ProjectRevision: Last Checkpoint: 1.1.1.6 $
  $RCSfile: si_add.dita $
  $Revision: 1.2 $
  $SandboxSetting  $
  $Setting  $
  $Source: si_add.dita $
  $State: Exp $
  ```

- `-R`
- `--[no|confirm]recurse`

  controls whether to select non-members recursively. Non-members are files that exist in the Sandbox directory but have not previously been added to the server repository.

- `--exclude=`*`file:pattern,dir:pattern...`*

  specifies a file that contains a glob pattern for excluding members.

- `--include=`*`file:pattern,dir:pattern...`*

  specifies a file that contains a glob pattern for including members.

- `--cancelOnInconsistentLineTerminators`

  specifies to cancel the operation if inconsistent line terminators are encountered.

- `--normalizeInconsistentLineTerminators`

  if inconsistent line terminators are encountered, specifies to normalize the line terminators to native for the system. The working file is normalized and then the operation is deferred. Examine the file for correctness before submitting the deferred operation to be committed to the repository. If there are multiple operations that are impacted, each affected operation is deferred and other operations are committed if Integrity is configured to do so.

- *`nonmember...`*
- identifies a specific file to add to your Sandbox; use spaces to specify more than one. When adding members, the available file types are restricted by values specified in the Non-Member Files Filter preferences. Your administrator may configure the preference for you, as well as lock it from being modified. If you are unable to add members with file types that are needed, contact your administrator for more information.

Note

All files must be added "in tree", that is, the nonmember must exist in the Sandbox directory or a subdirectory. Shared "out of tree" histories are supported by using the `--archive` option to point to the "out of tree" history that you want to associate with the member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addmemberfromarchive, si ci, si closecp, si co, si cpissues, si createcp, si drop, si projects, si sandboxes, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si addlabel

assigns a label to project members

Synopsis

si addlabel [--[no|confirm]moveLabel] [(-r *rev*|--revision=*rev*)] [(-L *label*|--label=*label*)] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *member...*

Description

siaddlabel assigns a label to a revision of one or more members of an Integrity configuration management project. For example:

```
si addlabel -L Prototype innactivebutton.c
```

If you do not specify any members, the si addlabel command applies to all project members.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no|confirm]moveLabel

  controls whether the application should move a label from one revision to another.

  - --nomoveLabel disables the moving of a label.
  - --confirmmoveLabel displays a confirmation message.
  - --moveLabel moves the label.
- -L*label*
- --label=*label*

  identifies a label. To add the label "October 15th Prototype" to a member file activebutton.c, you would enter

  si addlabel -L "October 15th Prototype" activebutton.c

  Note the following about using labels:
- Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
- Labels cannot have the same format as a valid revision number.
- Labels that include spaces must be enclosed by quotes.
- PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
- Labels cannot contain numbers without any spaces, for example, 14325. Numbers that contain spaces are acceptable, for example, 2432 1234.
-
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addprojectlabel, si ci, si deletelabel, si deleteprojectlabel, si rlog, si viewlabels
- Miscellaneous: ACL, diagnostics, options, preferences

si addmemberattr

adds attributes to a member

Synopsis

```
si addmemberattr [(--attr=key[=value]|--attribute=key[=value])] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si addmemberattr` adds attributes to a `member`, replacing any previous attribute of that same name specified with the member. If no members are specified, the command applies to all members of the project.

Attributes are key-value pairs of strings that are associated with a member. They are used for automated operations. For example, you could have an attribute indicating the target operating system type:

```
si addmemberattr --attr OS=WIN32 activebutton.c
```

and then you could perform other operations, such as si resync, on just those files through the `--filter` universal option:

```
si resync --filter=attribute:OS=WIN32
```

Note

Attribute keys cannot contain a hyphen (-) as the first character. The first character must either be a underscore (_) or an alpha character.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=`*`key`*`[=`*`value`*`]`
- `--attribute=`*`key`*`[=`*`value`*`]`

  sets the attribute key and, optionally, the value. Note: you cannot have commas (,) in the `key` or `value`, nor an underscore (_) at the beginning of an attribute.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addprojectattr, si dropmemberattr, si dropprojectattr
- Miscellaneous: ACL, diagnostics, options, preferences

si addmemberfromarchive

adds a member from a server archive

Synopsis

```
si addmemberfromarchive [--archive=value] [--[no]createSubprojects] [--[no]defer] [--[no|confirm]includeFormers] [--include=file:pattern,dir:pattern...] [--exclude=file:pattern,dir:pattern...] [--cpid=ID|--changePackageId=ID] [--[no|confirm]closeCP] [--issueId=value] [--[no]failOnAmbiguousProject] [--devpath=value] [-P|--project=value] [-r|--revision=value] [-S|--sandbox=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] nonmember...
```

Description

`si addmemberfromarchive` adds members from server archives.

The `si addmemberfromarchive` command can be used for the following:

- Add previously dropped members without requiring a Sandbox working file, while preserving the member history.

  From the Sandbox directory, for example, and assuming the three specified members have just been mistakenly dropped, the following command will re-add them:

  ```
  si addmemberfromarchive label.c button.c panel.c
  ```

- Add a member that shares the history of another existing member, where the source member can be located in another project from the added member.

  From the Sandbox directory, for example, the following command will add a member named largeLabel.c that shares the history with all members already linked to the specified archive location.

  ```
  si addmemberfromarchive --archive=/anotherproject/rcs/label.c largeLabel.c
  ```

Note

Note the following about the command:

- can be deferred when performed from a Sandbox
- may be part of change package review (displayed as pending members in the project)
- supports multiple selection of archives using the wizard

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--archive=`*value*

  a non-default archive location (the specified archive must exist on the server). To add several members with non-default archive locations in one operation, use the GUI wizard (`si addmemberfromarchive -g`).

- `-r=`*value*
- `--revision=`*value*

  specifies the revision at which the member is to be added. This must be an existing revision in the archive. By default, the member is added at the latest revision on the main branch.

- `--[no]createSubprojects`

  controls whether to create subprojects for each subdirectory encountered when adding members. This option is commonly used where you anticipate working with a large directory structure, because multiple subprojects are easier to manage than many subdirectories within one project. For example, specifying:

  si addmemberfromarchive --createSubprojects --description="Button Application" \buttons\activebutton.c

  creates a subproject for the `\buttons` directory and adds the file `activebutton.c`. Without the `--createSubprojects` option, the file and its path simply would be added to the project in the current directory.

- `--[no]defer`

  controls whether to delay the add member from archive operation in the project until the deferred operation is submitted. The operation in the Sandbox still takes place immediately.

  If the change package reviews are mandatory, specify the `--defer` option to create a pending entry for this operation at the time of change package submission. If the `--defer` option is not specified, Integrity creates the pending entry at the completion of this operation. When a deferred import member operation is submitted as part of a review, a pending member is created For more information, see the *PTC Integrity User Guide*.

- `--[no|confirm]includeFormers`

  controls whether to include former members. Former members are members that have been dropped but whose working files still reside in the Sandbox directory.

- `--exclude=`*file:pattern,dir:pattern...*
- specifies a file that contains a glob pattern for excluding members.
- `--include=`*file:pattern,dir:pattern...*

  specifies a file that contains a glob pattern for including members.

- `--[no|confirm]closeCP`

  closes the change package after command completion.

- *nonmember...*

  identifies non-members to add; use spaces to specify more than one non-member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si cpissues, si createcp, si drop, si lock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si addproject

adds an existing project to the list of top-level projects

Synopsis

```
si addproject [--[no]openView] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]add] project location...
```

Description

`si addproject` adds an existing project to the list of top-level projects. For example,

```
si addproject C:/phonespec/aurora.pj
```

The si addproject command can be used for the following:

- Expose an existing subproject as a top-level project.
- Expose a project imported with the --noadd flag.
- Re-expose a dropped top-level project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]openView`

  controls whether to open the project view after this project is added. If `--noopenView` is used, the project view does not open.
- *project location...*

  identifies a location on the Integrity Server for the existing project to be added. Regardless of your operating system, all paths are specified here using forward slashes (/) and should include the name of the project file (typically `project.pj`). Use spaces to specify more than one project location.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si createproject, si createsandbox, si createsubproject, si dropproject, si projects, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si addprojectattr

adds attributes to a project

Synopsis

```
si addprojectattr [(--attr=key[=value]|--attribute=key[=value])] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si addprojectattr` adds attributes to a project, replacing any previous attribute of the same name specified with the project.

Attributes are key-value pairs of strings that are associated with the project. They are used for automated operations. For example, you could have an attribute indicating the target operating system type:

```
si addprojectattr --project=C:/phonespec/aurora.pj --attr OS=WIN32
```

Note
Attribute keys cannot contain a hyphen (-) as the first character. The first character must either be an underscore (_) or an alpha character.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=key[=value]`
- `--attribute=key[=value]`

  sets the attribute key and, optionally, the value. Note: you cannot have commas (,) in the `key` or `value`, nor an underscore (_) at the beginning of an attribute.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addmemberattr, si dropmemberattr, si dropprojectattr, si dropsandboxattr, si addsandboxattr
- Miscellaneous: ACL, diagnostics, options, preferences

si addprojectlabel

assigns a label to a project

Synopsis

```
si addprojectlabel [--[no|confirm]moveLabel] [(-L label|--label=label)] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]includeUnchangedSubprojects] [-R|--[no|confirm]recurse]
```

Description

`si addprojectlabel` assigns a label to a checkpointed revision of a project. For example,

```
si addprojectlabel --project=c:/Aurora_Program/bin/Libra/project.pj -R --label="Release Candidate 002"
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]moveLabel`

  controls whether the application should move a label from one revision to another.

  `--nomoveLabel` disables the moving of a label.

  `--confirmmoveLabel` displays a confirmation message.

  `--moveLabel` moves the label.

- `-L label--label=label`

  identifies a label. Note the following about using labels:

  - Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
  - Labels cannot have the same format as a valid revision number.
  - Labels that include spaces must be enclosed by quotes.
  - PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
  - Labels cannot contain numbers without any spaces, for example, 14325. Numbers that contain spaces are acceptable, for example, 2432 1234.

- `--[no]includeUnchangedSubprojects`

  specifies if to label unchanged subprojects in the selection. If this option is not specified, the selection does not include subprojects that were not checkpointed because they were unchanged.

- `-R`

  `--[no|confirm]recurse`

  controls whether to assign the label to subprojects recursively. Note the following:

  - If a subproject is not in the most recent checkpoint for the project selection due to it being unchanged, that subproject is not labeled. To label such subprojects, use the `--[no]includeUnchangedSubprojects`option with this command.
  - If you do not have permission to access a subproject, an error is displayed and the subproject and all of its child subprojects are not labeled.
  - If you do not have permission to move a label, an error is displayed for the affected subproject.
  - If a subproject is configured as build, the subproject is not labeled. Consequently, build subprojects on extendable development paths are also not labeled.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addlabel, si ci, si deletelabel, si deleteprojectlabel, si rlog, si viewlabels
- Miscellaneous: diagnostics, options, preferences

si addprojectmetric

adds metrics to a project checkpoint

Synopsis

```
si addprojectmetric [--metric=value] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject]
[--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--
selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si addprojectmetric` adds the value for one or more metrics to the specific project checkpoint. For example,

```
si addprojectmetric --project=c:/Aurora_Program/bin/Libra/project.pj --projectRevision=1.2 --metric=functions=50,10
```

You can use a third party tool to calculate metrics for C or C++ in the context of a sandbox. You add the calculated value of the metric to a project checkpoint using this command.

The `--projectRevision` option is mandatory. Metrics can only be added to a build project.

Note
Metrics are only supported for database type repositories.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--metric=value`

  specifies the metric value to add. The value is in the following format: `metricname=value[,count]`. For example: `--metric=functions=50,10`

  This example sets the count of the number of functions to 50, as computed over 10 files (this allows the metric to be displayed as an average of 5 functions per file).

  The *metricname* must be a known metric. Metrics are created using the `si createmetricinfo` command.

  This option can be repeated to add multiple metrics at the same time.

-

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si viewprojectmetrics, si createmetricinfo, si viewmetricsinfo, si calculateprojectmetrics
- Miscellaneous: diagnostics, options, preferences

si addsandboxattr

adds Sandbox attributes

Synopsis

```
si addsandboxattr [--attr|attribute=key=value] [-S|--sandbox=value] [--[no]failOnAmbiguousProject] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]]
```

Description

`si addsandboxattr` adds Sandbox attributes. For example,

```
si addsandboxattr --sandbox=C:/Aurora_Program/Libra/project.pj --attribute=OS=WIN32
```

Note
Attribute keys cannot contain a hyphen (-) as the first character. The first character must either be a underscore (_) or an alpha character.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=key=value`
- `--attribute=key=value`specifies the attribute to set.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si dropsandboxattr, si configuresandbox, si addprojectattr, si dropprojectattr
- Miscellaneous: ACL, diagnostics, options, preferences

si addsubproject

adds an existing subproject to a project

Synopsis

`si addsubproject [-r subproject revision|--subprojectRevision=subproject revision] [--subprojectDevelopmentPath=subproject development path name|--variant=subproject development path name] [--type=[normal|variant|build|default]] [(-Pproject|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--cpid=ID|--changePackageId=ID] [--[no|confirm]closeCP] [--issueId=value] subproject location...`

Description

`si addsubproject` allows you to re-add a dropped subproject. For example,

`si addsubproject c:/Aurora_Program/bin/Libra/project.pj`

Note

If you want to add a normal subproject to a variant project, the normal subproject does not require a development path.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-r=subproject revision`
- `--subprojectRevision=subproject revision`

  specifies the checkpoint number (for build subprojects). For example, `1.5`. This option is used with `--type=build`.
- *--subprojectDevelopmentPath=subproject development path name*
- *--variant=subproject development path name*

  specifies the development path name (for variant subprojects). For example, `Service_Pack3`. This option is used with `--type=variant`
- `--type=[normal|variant|build|default]`

  specifies the new configuration type for the subproject.

  `--type=normal` configures the subproject based upon the current state of the subproject.

  `--type=variant` configures the subproject based upon a specific development path. The option is used with --variant=*subproject development path name* or --subprojectDevelopmentPath=*subproject development path name*.

  `--type=build` configures the subproject as a static subproject based upon a specific checkpoint of the master project that is used for building or testing the project, but not for further development. This option is used with -r *subproject revision* or `--subprojectRevision=subproject revision`.

  `--type=default` configures the subproject based on the type that is consistent with the parent project that you are adding the subproject to. For example, if you add a subproject to a normal project, the subproject is added as a normal type. For information on what the default type is, see your administrator.
- `--cpid=ID`
- `--changePackageId=ID`

  identifies a change package that is notified of this action, for example, `1452:1`. Note the following about using this option:
- This option can only be specified if change packages are enabled.
- If the integration is enabled, but it is not mandatory to specify a change package, or if no change package is applicable, you can specify `--changePackageId=:none`.
- The next time you are prompted for a change package ID, the last used change package ID is displayed by default, if :none was specified or at least one change package was successfully updated from the last operation.
- `--[no|confirm]closeCP`

  closes the change package after command completion.
- `--issueId=value`

  specifies the issue ID that corresponds to the change package that records the changes.
- *subproject location...*

  specifies the path and name of the subproject you want to add, for example, c:/Aurora_Program/bin/Libra/project.pj

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si configuresubproject, si createproject, si createsubproject, si addsubproject, si movesubproject, si createsandbox, si drop, si dropproject, si projectinfo, si projects, si sharesubproject, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si admingui

displays configuration management policies and permissions in the Administration Client

Synopsis

```
si admingui [--[no]restoreDesktop] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]
```

Description

`si admingui` launches the Integrity Administration Client GUI. The client interface provides a single, centralized access point for the most common administration tasks. More specifically, you can manage Access Control Lists (ACLs), manage and distribute ViewSets, set up workflows and documents, configure configuration management policies, and send alert messages.

Note

The Integrity Administration Client GUI interface essentially acts as a container for several administration applications (workflow and document management; and configuration management). From the client interface, you can specify one or more servers to administer, allowing you to have multiple Administration windows open. In addition, you can specify application preferences. Similar CLI commands that offer full administrative functionality include: `aa admingui`, `im admingui`, and `integrity admingui`. These commands and `si admingui` should not be confused with the `integrity admin` command, which launches an Administration window instance for a specific server. While you can administer all of the administrative applications for that server, you cannot connect to other servers or specify application preferences.

For information on using the Integrity Administration Client, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]restoreDesktop`

  controls whether to restore any windows that were active when the graphical user interface was closed.
- `--height=`*`value`*

  specifies the height in pixels of the window.
- `--width=`*`value`*

  specifies the width in pixels of the window.
- `-x` *`value`*

  used with the `-g` or `--gui` options, specifies the x location in pixels of the window.
- `-y` *`value`*

  used with the `-g` or `--gui` options, specifies the y location in pixels of the window.

See Also

- Commands: integrity about, integrity disconnect, integrity exit, integrity admingui, integrity loadrc, integrity servers, integrity updateclient
- Miscellaneous: options

si annotate

displays an annotated revision.

Synopsis

```
si annotate [--[no]defaultFormat] [--[no|un]expand] [--fields=field1[:width1],field2[:width2]...] [--guiCharacterEncoding=value] [--height=value] [--width= value] [-x=value] [-y=value] [--[no]failOnAmbiguousProject] [-r|--revision=value] [--devpath=value] [-P|--project=value] [-S|--sandbox=value] [--projectRevision=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member
```

Description

`si annotate` displays an annotated revision. For example,

```
si annotate --project=c:/Aurora_Program/bin/Libra/project.pj cell.c
```

Use the command when you want to know the reason and circumstances a line was introduced or changed. Rather than viewing the content of each revision in the history one revision at a time, you can see the line by line history that includes information on a per revision basis.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]defaultFormat`

  specifies if the default format is used to display the output.
- `--fields=field1[:width1],field2[:width2]...`

  The fields available for printing can be one or more of the following:
- `author`

  displays the author of the revision.
- `cpid`

  displays the line's associated change package ID.
- `date`

  displays the date each line in the history was created.
- `labels`

  displays revision labels.
- `linenum`

  displays the line number for each line of text in the revision..
- `revision`

  displays the line's revision number.
- `text`

  displays the text contained in the line.
- `--[no|un]expand`

  specifies if the keywords are expanded in the output.
- `--guiCharacterEncoding=value`

  specifies the character encoding to use for the revision contents in the GUI, and can only be specified with the -g option. Integrity automatically decodes UTF-8 revision contents based on the presence of a byte order mark (BOM) in the file. You can set character encoding preferences for each view and command through your client preferences. The preference setting is used if the character set cannot be determined through the file's BOM, or if no character set is specified in the command line. By default, the character set in the preferences matches the default character set provided by the client's operating system locale. The option does not apply to pure CLI output, or third party merge and/or differencing tools. Possible values are:

  ```
  UTF-8
  US-ASCII
  windows-1252
  ISO-8859 -1
  ISO-8859 -15
  IBM437
  IBM850
  IBM863
   EUC-JP
  Shift_JIS
  x-euc-jp-linux
  x-eucJP-Open
  x-windows-iso2022jp
  IBM862 ISO-8859-8
  ISO-8859-9
  ```

- *member*

  specifies the name of the member to view.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewprojecthistory, si viewsandbox
- Miscellaneous:ACL, diagnostics, options, preferences

si appendcheckpointdesc

appends additional text to a checkpoint description

Synopsis

```
si appendcheckpointdesc [(-d desc|--description=desc)] [--descriptionFile=file] [(-P project|--project=project)] [--
[no]failOnAmbiguousProject] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--
usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] project...
```

Description

`si appendcheckpointdesc` appends additional text to a checkpoint description. For example,

```
si appendcheckpointdesc --projectRevision=1.23 --description="Release Canadidate" c:/Aurora_Program/bin/Libra/project.pj
```

A checkpoint description is created when you checkpoint a project. When you review the description for a specific checkpoint, through the si viewprojecthistory or si projectinfo command, you see the appended information in a manner similar to the following:

```
--- Added comments --- sholmes [Oct 23, 2009 2:39:31 PM EDT]
Appended Description
```

Note

Appending a checkpoint description recursively appends the checkpoint description for all subprojects.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions. You must specify the `--project` and `--projectRevision` options.

- `-d` *desc*
- `--description=`*desc*

  specifies the new description text to be appended to the existing checkpoint description. These options and the `--descriptionFile` option are mutually exclusive.

  Note

  Descriptions that include spaces must be enclosed by quotes.

- `--descriptionFile=`*file*

  specifies a file name, `file`, containing the new description text to be appended to the existing checkpoint description. This option and the `--description` options are mutually exclusive.
- *project...*

  identifies a specific project; use spaces to specify more than one project.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addprojectlabel, si projectinfo, si checkpoint
- Miscellaneous: ACL, diagnostics, options, preferences

si appendrevdesc

appends additional text to a revision description

Synopsis

```
si appendrevdesc [(-d desc|--description=desc)] [--descriptionFile=file] [(-r rev|--revision=rev)] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si appendrevdesc` appends additional text to a revision description. For example,

```
si appendrevdesc --revision=1.38 --description="Release Canadidate" c:/Aurora_Program/bin/Libra/cell.c
```

A revision description is created when you check in a new revision. When you review the description for a specific revision, through the si viewhistory command, you see the appended information in a manner similar to the following:

```
--- Added comments --- sholmes [2002/07/20 20:23:55Z]
Appended Description
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-d` *desc*
- `--description=`*desc*

  specifies the new description text to be appended to the existing revision description. These options and the `--descriptionFile` option are mutually exclusive.

  Note

  Descriptions that include spaces must be enclosed by quotes.

- `--descriptionFile=`*file*

  specifies a file name, `file`, containing the new description text to be appended to the existing revision description. This option and the `-d` or `--description` options are mutually exclusive.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addlabel, si archiveinfo, si ci
- Miscellaneous: ACL, diagnostics, options, preferences

si applycp

applies one or more change packages to an Integrity configuration management project

Synopsis

```
si applycp [--[no]alreadyInProjectIsError] [--backFill=[cp|revision|error|skip|ask]] [--[no]useMaster] [--[no]failOnAmbiguousProject]
[--[no|confirm]closeCP] [--[no]confirm] [--[no|confirm]createVariants] [--notify=[never|onCompletion|onError]] [--[no]ignoreBranches]
[--[no]ignoreServer] [--[no]otherProjectIsError ] [--[no]spanProjects] [--[no]verbose] [(-P project|--project=project)] [(-S
sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname =server] [--port=number] [--password=password] [--user=name] [(-?|--usage)]
[(-F file|--selectionFile=file)] [(-N |--no)] [(-Y|--yes)] [--[no]batch] [--cwd =directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [-
-quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--cpid=ID|--changePackageID=ID] [--[no]ignoreUpdateRevision] [--
issueID=value] [--subprojectPropagation=[explicit|implicit]] issue|issue:change package id...
```

Description

`si applycp` propagates changes recorded in change packages from one project or development path to another. This enables you to propagate only the changes that you want to. The process updates member revisions and can also involve adding, dropping, renaming, and moving files. Additionally, it can involve creating, adding, dropping. or moving subprojects. The `si applycp` command is most useful for building software.

Note

If you are using the `si applycp` command to propagate changes recorded in a change package to an extendable development path, the project configuration is updated automatically. *Extenddevpath* permission is required when operating against an extendable development path.

For more detailed information and examples for `si applycp`, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--cpid=`*ID*
- `--changePackageId=`*ID*

  identifies a propagation change package to record the changes made as a result of the `si applycp` operation. Only open change packages can be used. This is not the change package that you want to apply to the project; it is the change package that you want to use to record the results of the operation.
- `--[no]alreadyInProjectIsError`

  Causes Integrity to terminate the operation if the change being applied has already been applied to the project. If this setting is negated `--noalreadyInProjectIsError`, then the information is displayed as a warning.
- `--backFill=[cp|revision|error|skip|ask]`

  Specifies the way Integrity treats historic revisions required by the specified change package.
  - `--backFill=cp`

    recursively chooses all historic revisions required by the specified change packages and applies them by updating member revisions, adding files, or dropping files.
  - `--backFill=revision`

    processes only the specified change package(s) and chooses only directly associated revisions. It does not process any change packages that are associated with intermediate revisions.
  - `--backFill=error`

    terminates the operation if other change packages are required but are not specified.
  - `--backFill=ask`

    allows you to specify the change packages you want to include.
- `--[no]confirm`

  specifies whether to confirm the actions before starting them.
- `--[no|confirm]createVariants`

  specifies whether to create new variant subprojects within the target variant project as required. This allows the subproject structure of the target variant to mirror the structure of the source project.
- `--notify=[never|onCompletion|onError]`

  specifies whether to display a report when the command is complete. The report details the operations that were performed and any errors that were encountered.
  - `--notify=never`

    never displays the report.
  - `--notify=onCompletion`

    always displays the report.
  - `--notify=onError`

    displays the report if any errors were encountered.
- `--[no]ignoreBranches`

  specifies whether to use the most recent revision when Integrity encounters two revisions of the same member on different branches in the change package being applied.
- `--[no]useMaster`

  operates on the top-level project if the target is a subproject. Changes are applied throughout the top-level project's hierarchy.
- `--[no]ignoreServer`

  specifies whether to perform the Apply CP operation even if the change package members reside on different servers.

  Note

  The `--ignoreServer` option is useful when a server is the same but is now identified differently, for example, the server name has changed. This option is required because projects are defined by their server and path. Use this option with caution.

- `--[no]ignoreUpdateRevision`

  ignores update revision change package entries. Use this option when the change package contains backward revision updates, for example, from 1.4 to 1.2. The default is to include update revision entries for backwards compatibility with older version of Integrity (formerly MKS Source).
- `--[no]otherProjectIsError`

  specifies whether to terminate the command if the change is for a project other than the one you initially targeted. If this setting is negated `--nootherProjectIsError`, then the information is displayed as a warning.
- `--[no]spanProjects`

  specifies whether to apply all changes in the change package, even if this involves a different project than the one you initially targeted. If you are applying a change package that contains move member or move subproject entries between two project, the `--[no]spanProjects` option must be used.

  Caution

This is the only operation that has the potential to affect other projects.

- `--[no]verbose`

  specifies whether to include additional information to track the current state of the command.
- `--[no|confirm]closeCP`

  closes the change package after command completion.
- `--subprojectPropagation=[explicit|implicit]`

  specifies how to apply subproject changes required by the specified change packages.
  - `--subprojectPropagation=explicit`

    creates, adds, drops, or moves a subproject only if there is a explicit command to do so in the change package.
  - `--subprojectPropagation=implicit`

    creates, adds, drops or moves a subproject if the operation is implicitly required based on the change package entries. For example, if you are adding a member that is part of a subproject that does not exist in the target project being updated, the subproject is added.
- `--issueID=value`

  specifies the issue ID that corresponds to the change package that records the changes.
- *issue...*

  issue identifies a specific issue that contains all change packages that you want to include; use spaces to specify more than one issue.
- *issue:change package id...*

  issue:change package id identifies a specific change package to include; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si resync, si resynccp
- Miscellaneous: ACL, diagnostics, options, preferences

si archiveinfo

displays information about an archive

Synopsis

`si archiveinfo [--[no]labels] [--[no]locks] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--lockRecordFormat=value] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...`

Description

`si archiveinfo` displays global information about an archive. For example,

```
si archiveinfo --locks --lockrecordformat={locker} c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

The command does not contain any per-revision information. For example:

```
Member Name: c:\Documentation\Man_Pages\xml_man\si_add.1.xml
Sandbox Name: c:\Documentation\Man_Pages\project.pj
Archive Name: \rd\doc\Man_Pages\xml_man\rcs\si_add.1.xml
Archive Type: Text
Shared
Exclusive Lock Mandatory
Archive Description:
Labels:
 TechReview1 1.7
 PeerReview_1 1.6
```

Note

Shared displays only if other members share the archive and you are using the database repository.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]labels`

  controls whether to show labels and associated revision IDs.
- `--[no]locks`

  controls whether to show locks and associated revision IDs.
- `--lockRecordFormat=value`

  defines the format for displaying lock information for the archive. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:
  - `{revision}`

    displays the revision that is locked.
  - `{locker}`

    displays the user who locked the revision.
  - `{locktype}`

    displays the type of lock on the revision (exclusive or non-exclusive).
  - `{locktimestamp}`

    displays the time when the revision was locked.
  - `{lockcpid}`

    displays the change package associated with the lock on the revision.
  - `{project}`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `{devpath}`

    displays the name of the development path where the lock on the revision was made from.
  - `{sandbox}`

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
  - `{hostname}`

    displays the hostname of the computer that locked the the revision.
  - `{hascpid}`

    displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
  - `{hassandbox}`

    displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
  - `{hasdevpath}`

    displays 1 if the lock was made from a development path, 0 if it wasn't.
  - `{member}`

    displays the name of the locked revision.
- *`member...`*

  identifies a specific member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si memberinfo, si mods, si revisioninfo, si rlog, si viewhistory, si viewlabels
- Miscellaneous:

  ACL, diagnostics, options, preferences

si calculateprojectmetrics

calculates the metrics for the specified project checkpoint

Synopsis

```
si calculateprojectmetrics [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--
[no]recomputeall] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F
file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet]
[--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si calculateprojectmetrics` calculates the metrics for the specified project checkpoint. For example,

```
si calculateprojectmetrics --project=c:/Aurora_Program/bin/Libra/project.pj --projectRevision=1.34 --recomputeall
```

If you decide to track metrics for a project that has already been checkpointed, you can calculate metrics for existing checkpoints using this command. All project level statistics are recomputed, but each member's statistics are only computed if none currently exist. You can use the `--recomputeall` option to recompute all member metrics.

The `--projectRevision` option is mandatory. Metrics can only be calculated for a build project.

There may be a performance impact the first time you calculate project metrics. The first time you calculate metrics for a project with a large number of members that existed before implementing metrics, the metric calculation may take a significant amount of time. The increase in time is due to the need for the metrics to be initially computed for the pre-existing members. Subsequent metric calculations are aggregated for members of the project.

Note

Metrics are only supported for database type repositories.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]recomputeall`

  specifies whether you want to recompute metrics for all members. This is useful when new metrics have been added.

  This recomputes metrics for all members recursively, summing up all their individual metrics, and putting the rolled up values on the specified project.

  If a given member has no metrics, then it is assumed to not have had metrics computed for it. This command computes metrics for them by invoking the metrics trigger. If a member has even one metric, it is assumed to have had metrics computed for it, and this command doesn't compute them. So if you changed the trigger to generate new metrics, they will not automatically be recomputed unless you use the `--recomputeall` option, which forces the command to ignore the existing metrics and rerun metrics on each member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si viewprojectmetrics, si createmetricinfo, si createmetricinfo, si viewmetricsinfo
- Miscellaneous: diagnostics, options, preferences
-

si checkpoint

checkpoint archives in a project

Synopsis

```
si checkpoint [--author=name] [(-d desc|--description=desc)] [--descriptionFile=file] [--[no]notify] [(-L label| --label=label)] [--
[no]failOnAmbiguousProject] [--asof=date] [(-s state|--state =state)] [--[no]stateMembers] [(-P project|--project=project)] [(-S
sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)]
[(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]checkpointUnchangedSubprojects]
```

Description

You can track the evolution of an entire project by preserving the changes made to it from one revision to another. In Integrity, this process is called `checkpointing`. For example,

```
si checkpoint --project=c:/Aurora_Program/bin/Libra/project.pj --label="Release Candidate 002" --description="Ready for Review" --notify
```

Checkpointing a project creates a new revision of the project and adds it to the project history. When you checkpoint a project, you save all the information needed to recreate the project completely at any time in the future. The saved information includes the project structure and the list of members with their revision numbers.

You can either perform a checkpoint on the current configuration of the project, or you can specify a date in the past (using the `--asof` option) to perform a retroactive checkpoint at that specific date.

Checkpointing as of a date can be useful to reduce the number of total checkpoints on a project by only retroactively checkpointing the exact project configuration that you intend to use as a baseline, instead of creating checkpoints of current project configurations that might never be used. When checkpointing as of a date, the best practice is to specify a label to facilitate later identification of that checkpoint.

Date-based checkpoints (a project configuration as of a specific date) have all of the same capabilities as regular checkpoints, but they are identified differently when they are used or appear in the project history. A regular checkpoint is identified using an Integrity revision ID; for example, checkpointing a project at 1.1 results in revision 1.2. However, a date-based checkpoint for a project whose nearest regular checkpoint (not date-based) prior to that date is at revision 1.1 results in revision of 1.1.0.0.*date*.*identifier*, where date is the *date* of the project configuration (presented in milliseconds since the epoch) and *identifier* is an integer (usually 0, but higher if there were simultaneous operations). For example, checkpointing a project configuration that was January 5, 2015, 19:51:29 GMT (and no simultaneous operations) results in a revision ID of 1.1.0.0.1420487490.0. For supported date formats, see the description of the `--asof` option .

All checkpoints are *transactional*. This means that a checkpoint records the structure and contents of the project at the checkpoint date. It does not include anything added to the project or its subprojects after the checkpoint date, such as check ins or submitted change packages. To ensure that only complete change packages are included in your project checkpoints, your administrator must enable change package reviews. For more information on change package reviews, see the *PTC Integrity User Guide*.

While the checkpoint is in progress, you can perform member and subproject operations on the project, but you cannot perform any operations that affect project checkpoints, development paths, or labels. You cannot perform the following operations:

- Checkpoint the project on another development path
- Restore the project to a previous checkpoint
- Delete the project from the database
- Add or delete project labels
- Create or remove development paths

Note

Checkpointing a project checkpoints all its subprojects. If you do not want to create subproject checkpoints, configure them as build subprojects first. See the si configuresubproject command for more details.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--author=`*`name`*

  specifies an author name to identify the author of the new revision.
- *-d desc*
- `--description=`*`desc`*

  specifies a description for the checkpointed revisions. These options and the `--descriptionFile` option are mutually exclusive.

Note

Descriptions that include spaces must be enclosed by quotes.

- `--descriptionFile=`*`file`*

  specifies a file name, `file`, containing the description text for the checkpointed revisions. This option and the `-d` or `--description` options are mutually exclusive.
- `--[no]notify`

  controls whether to notify when the checkpoint is complete.
- `-L`*`label`*
- `--label=`*`label`*

  identifies a label. The label is applied to the checkpointed revision of the project. Note the following about using labels:
- Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
- Labels cannot have the same format as a valid revision number.
- Labels that include spaces must be enclosed by quotes.
- PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
- *`-s state`*
- `--state=`*`state`*

  specifies a state, `state`, for the checkpointed revisions. The state is applied to the checkpointed revision of the project, and to all members if `--stateMembers` is specified.
- `--[no]stateMembers`

  controls whether to set the state for members at member revision, in addition to the project.
- `--asof=`*`date`*

  specifies the project configuration as of a specific date. For example, `--asof="April 28, 2014 3:33:45 AM GMT-05:00"`.

  Integrity recognizes all current timezones whatever your locale (country), for example, CEST, CET, EDT, PST, or GMT+/-*hours*:*minutes*. The following information illustrates North American timestamps recognized by Integrity:

```
EEEE, MMMM d, yyyy h:mm:ss a z  |  Monday, April 28, 2014 3:33:45 AM GMT-05:00
EEEE, MMMM d, yyyy h:mm:ss a    |  Monday, April 28, 2014 3:33:45 AM
EEEE, MMMM d, yyyy h:mm a       |  Monday, April 28, 2014 3:33 AM
MMMM d, yyyy h:mm:ss a z        |  April 28, 2014 3:33:45 AM GMT-05:00
```

```
MMMM d, yyyy h:mm:ss a          |  April 28, 2014 3:33:45 AM
MMMM d, yyyy h:mm a             |  April 28, 2014 3:33 AM
MMM d, yyyy h:mm:ss a z         |  Apr 28, 2014 3:33:45 AM GMT-05:00
MMM d, yyyy h:mm:ss a           |  Apr 28, 2014 3:33:45 AM
MMM d, yyyy h:mm a              |  Apr 28, 2014 3:33 AM
M/d/yy h:mm:ss a z              |  4/28/14 3:33:45 AM GMT-05:00
M/d/yy h:mm:ss a                |  4/28/14 3:33:45 AM
M/d/yy h:mm a                   |  4/28/14 3:33 AM
h:mm:ss a z                     |  3:33:45 AM GMT-05:00
h:mm:ss a                       |  3:33:45 AM
h:mm a                          |  3:33 AM
EEEE, MMMM d, yyyy              |  Monday, April 28, 2014
MMMM d, yyyy                    |  April 28, 2014
MMM d, yyyy                     |  Apr 28, 2014
M/d/yy                          |  4/28/14
MMM d, yyyy - h:mm:ss a         |  Apr 28, 2014 - 3:33:45 AM
MMM d, yyyy - h:mm a            |  Apr 28, 2014 - 3:33 AM
```

- `--[no]checkpointUnchangedSubprojects`

  specifies if to checkpoint unchanged subprojects. If this option is not specified, subprojects that have not changed are not checkpointed; instead the existing revision for the subproject is used in the parent project checkpoint revision.

  If this option is not specified, the following is true:

  - The checkpoint description is not added or appended to the project revision for the unchanged subprojects.
  - The checkpoint label is not added to the revision for the unchanged subprojects.
  - If the subproject revision was created prior to Integrity 10.7, the unchanged subproject is checkpointed and its revision incremented. Only subproject revisions created by Integrity 10.7 or later can be detected as unchanged.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addprojectattr, si addprojectlabel, si ci, si createdevpath, si createproject, si createsubproject, si memberinfo, si projectinfo, si projects, si promoteproject, si setprojectdescription, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si ci

checks in members of a Sandbox

Synopsis

```
si ci [--[no|confirm]ignoreUnresolvedMerge] [(-L label|--label=label)] [--author=name] [--[no]branch] [--[no|confirm]branchVariant] [--[no|confirm]branchUpdate] [--[no|confirm]updateIfNotCurrent] [(--cpid=ID|--changePackageId=ID)] [--[no]failOnAmbiguousProject] [--issueId=ID] [--[no|confirm]checkinUnchanged] [--[no|confirm]closeCP] [--[no|confirm]convertArchiveOnGovernor] [(-d desc|--description=desc)] [--[no]defer] [--descriptionFile=file] [(-l|--[no]lock)] [--[no|confirm]moveLabel] [(-r rev|--revision=rev)] [--[no]retainWorkingFile] [--[no|confirm]revertUnchanged] [--revision=value] [--[no]saveTimestamp] [--onNewerRevision=[confirm|cancel|resync|resyncByCp]] [--[no|confirm]skipUpdateOnLockConflict] [--cancelOnInconsistentLineTerminators] [--normalizeInconsistentLineTerminators] [(-u|--unlock)] [--[no]unexpand] [--[no]update] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [-F value] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--selectionFile=value] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] sandbox member...
```

Description

`si ci` checks in and saves changes to Sandbox members. For example,

```
si ci --label="Pre-Release Candidate" --description="Ready for Review" c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If you do not specify any members, `si ci` applies to all members of an associated Sandbox.

Check in is an operation that adds a new revision of a file to an archive. When a file is checked in to a revision other than the head revision or a branch tip revision, a new branch is created.

`Assigning Revision Numbers`

By default, when you check in a member, Integrity automatically assigns a unique revision number to the new revision. It does this by incrementing the current revision number by one. For example, if the previous revision is 1.3, the new revision is assigned number 1.4.

You can choose the revision number of the changes you are checking in, so long as your revision number:

- is greater than the last revision number (you cannot use previously "skipped" revision numbers)
- has no leading zeros (zeros as complete revision numbers are acceptable)
- starts a new branch based on an existing revision

If you check in a revision using an already existing revision number, Integrity attempts to add one to the revision number and check it in as that revision. If that revision already exists, Integrity then chooses the next available branch number and creates a new branch.

For example, if you are checking in a new revision to an archive where the head revision is 1.7, the following numbers are valid:

- 1.8 (greater than head revision)--if you check in a revision as 1.7, which already exists, Integrity assigns it 1.8
- 1.10 (greater than head revision)
- 1.72 (none of the numbers between 7 and 72 may be used afterwards)
- 2.0
- 1.7.1.1 (if it starts a new branch)
- 1.7.0.1 (leading zero as the branch number)

The following numbers are invalid:

- 1.3 even if there was no revision 1.3 previously
- 1.08 (leading 0 in last portion)
- 02.1 is considered the same as 2.1 (leading zero in branch number)

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]ignoreUnresolvedMerge`
- controls whether to ignore a previously unresolved merge. For more information on merging revisions, see si merge and si mergebranch.
- *-L label*
- `--label=`*`label`*
- identifies a label that is applied to the new revision. Note the following about using labels:
- Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
- Labels cannot have the same format as a valid revision number.
- Labels that include spaces must be enclosed by quotes.
- PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
- `--author=`*`name`*
- specifies an author name.
- `--[no]branch`
- controls whether to force the creation of a branch revision.

Note

If required, a branch revision is created even if this option is set to "no", for example, if you are checking in a member to a revision other than the head revision.

A *branch* is a revision path that diverges from the main line of development (also known as the *trunk*) in a member or project history. A branch is typically created by checking in a file to a revision other than the head revision. The most recent revision of a branch is called the *tip revision*.

Integrity usually places new revisions at the top of the trunk, assigning them two-part revision numbers, such as 1.15. There are times, however, when you do not want your work to be checked into the trunk. You may be pursuing a line of development that will not be included in the finished product, for instance, or you may be doing post-release maintenance while development for the next release continues on the trunk.

Divergent lines of development in the same archive are managed through the use of branches. A branch is an independent revision line that uses an existing revision as its starting point. Members of a branch revision are identified by their revision numbers. Whereas revisions on the trunk are characterized by two-part revision numbers (for example, 1.2 or 3.5), branch revision numbers are prefixed with the number of the revision they start from. For example, if a branch revision is started from revision number 1.2, the members of that branch are numbered

```
  1.2.1.1
 1.2.1.2
 1.2.1.3
```

and so on. The first two digits of the number identify the revision where the branch diverges from the trunk, and the last two represent a position on the branch.

- `--[no|confirm]updateIfNotCurrent`
- controls whether to update the member revision, even if the revision being checked in is not the member revision. This option only applies if `--update` is specified.

  `--noupdateIfNotCurrent` means do not update the member revision.

  `--confirmupdateIfNotCurrent` means ask before updating the member revision.

  `--updateIfNotCurrent` always updates the member revision.
- `--[no|confirm]checkinUnchanged`

controls whether to force the checkin so that the new revision is checked in even if it is not different from the preceding one.

`--nocheckinUnchanged` means do not force the checkin.

`--confirmcheckinUnchanged` means ask before forcing the checkin.

`--checkinUnchanged` always forces the checkin.

- `--[no|confirm]closeCP`
- controls whether to close the associated change package.

  `--nocloseCP` means do not close the change package.

  `--confirmcloseCP` means ask before closing the change package.

  `--closeCP` always closes the change package.
- `--[no|confirm]convertArchiveOnGovernor`
- controls whether to automatically convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size. If you are using the database repository, your administrator may set the policy that determines the maximum size of text working files that can be stored in a store-by-delta archive. The policy is useful in preventing out of memory issues on the Integrity Server when Integrity attempts to difference large text files. If the file exceeds the maximum size during a check in, you can cancel the operation or convert the archive to store-by-reference.

  `--noconvertArchiveOnGovernor` means do not convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size. This cancels the check in operation.

  `--confirmconvertArchiveOnGovernor` means ask before converting a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size.

  `--convertArchiveOnGovernor` always convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size.
- *-d desc*
- `--description=`*desc*

  specifies a description for the new revision. This option and the `--descriptionFile` option are mutually exclusive.

Note

Descriptions that include spaces must be enclosed by quotes.

- `--[no]defer`

  controls whether to delay the checkin operation in the project. The operation in the Sandbox still takes place immediately.

  If reviews are mandatory, specify this option to submit the changes for review as a change package. If this option is not specified, a pending revision is created for the operation, that corresponds to a pending entry in the change package.
- `--descriptionFile=`*file*

  specifies a file name, `file`, containing the description text for the new revision. This option and the `-d` or `--description` options are mutually exclusive.
- `-l`
- `--[no]lock`

  controls whether Integrity locks the new revision. Locking the new revision allows you to update the archive while retaining control of the revision. The type of lock used is the same as the lock type used when the file was checked out.
- `--[no|confirm]moveLabel`

  controls whether the application should move a label from one revision to another.
  - `--nomoveLabel` disables the moving of a label.
  - `--confirmmoveLabel` displays a confirmation message.
  - `--moveLabel` moves the label.
- `--[no]retainWorkingFile`

  controls whether to keep the working file in the Sandbox after checkin. By default, a sparse Sandbox does not retain the working file after checkin.
- `--[no|confirm]revertUnchanged`

  controls whether to revert an unchanged member, releasing the lock.

Note

If specified, the `--[no|confirm]checkinUnchanged` option takes precedence over this option.

- `--[no]saveTimestamp`

  controls whether to set the revision timestamp to the modification date and time of the working file rather than the current date and time.
- `-u`
- `--unlock`

  unlocks the newly checked-in revision. This is equivalent to specifying `--nolock`.
- `--[no]unexpand`

  controls whether to unexpand or ignore keywords in the member file prior to the check in. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

```
$Author: Warner, Carrie (cwarner) $
$CompanyInfo$
$Date: 2015/11/30 13:20:32EST $
$Header: si_ci.dita 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner) Exp  $
$Id: si_ci.dita 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner) Exp  $
$Locker:  $
$Log: si_ci.dita  $
Revision 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner)
XML tagging fixes
Revision 1.1 2015/10/29 10:24:47EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectLabel:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$Revision: 1.2 $
$SandboxSetting  $
$Setting  $
$Source: si_ci.dita $
$State: Exp $
```

- `--[no]update`

  controls whether to set the checked in revision as the project's member revision.

- `--onNewerRevision=[confirm|cancel|resync|resyncByCp|]`

  controls what happens when the revision being checked in is not the member revision in the development path.

  - `--onNewerRevision=confirm` means ask for confirmation of the action to be taken.
  - `--onNewerRevision=cancel` means cancel the operation.
  - `--onNewerRevision=resync`means resynchronize the member revision into the working file and move the lock (if any) to the member revision. The check in operation is not completed. You should perform additional testing on the merged file before checking it in.
  - `--onNewerRevision=resyncbycp` means resynchronize the member revision into the working file by change package, and move the lock (if any) to the member revision. The check in operation is not completed. You should perform additional testing on the merged file before checking it in.

Note

If you specify a revision using the`--revision` option or force the creation of a branch using the `--branch` option, no action is required when checking in a member that is not the member revision.

- `--[no|confirm]branchVariant`

  controls whether or not to create a branch when in a variant context and branching is optional. Branching is optional if the revision being checked in is the branch tip. If the revision is a previous revision on the branch, then a new revision branch must be created regardless of the value of this option. Note that this check is done once per member on the development path, and is not done again on future check in operations for that member. This option is useful for keeping changes in the variant separate from changes made in the mainline project.
- `--[no|confirm]skipUpdateOnLockConflict`

  instructs Integrity not to update the member revision if the member revision is exclusively locked by another user. This option is development path specific. If this option is not specified, the default is to confirm. This option is not available in the GUI, but can be set for the GUI from the `si setprefs` command
- `--cancelOnInconsistentLineTerminators`

  specifies to cancel the operation if inconsistent line terminators are encountered.
- `--normalizeInconsistentLineTerminators`

  if inconsistent line terminators are encountered, specifies to normalize the line terminators to native for the system. The working file is normalized and then the operation is deferred. Examine the file for correctness before submitting the deferred operation to be committed to the repository. If there are multiple operations that are impacted, each affected operation is deferred and other operations are committed if Integrity is configured to do so.
- *sandbox member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si closecp, si co, si cpissues, si createcp, si diff, si edit, si lock, si mergebranch, si rlog, si unlock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si closecp

closes a change package.

Synopsis

`si closecp [--[no|confirm]allowOrphanedDeferred] [--[no]confirm] [--[no|confirm]releaseLocks] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`issue|issue:change package id...`*

Description

`si closecp` closes a change package. For example,

```
si closecp --confirmallowOrphanedDeferred --confirm --confirmreleaseLocks 3434:2
```

Note
If reviews are mandatory, you cannot close the change package using this command. The change package can only be closed by Integrity once it has successfully completed the review process. For more information, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]allowOrphanedDeferred`

  controls if orphaned deferred operations are permitted when the change package is closed.
- `--[no]confirm`

  controls whether to confirm closing individual change packages.
- `--[no|confirm]releaseLocks`

  controls whether or not outstanding locks are released when the change package is closed.
- *issue...*

  issue:change package id...

  issue identifies a specific issue that contains all open change packages that you want to close; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to close; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si cpissues, si createcp, si drop, si lock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si co

checks out members into working files in a Sandbox

Synopsis

si co [--mergeType=*[confirm|cancel|automatic|manual]*] [--lockType=*[exclusive|nonexclusive|auto]*] [--[no|confirm]lockOnFreeze] [--onMergeConflict=*[confirm|cancel|mark|launchtool|highlight|error]*] [--[no]failOnAmbiguousProject] [(--cpid=*ID*|--changePackageId=*ID*)] [--issueId=*ID*] [(-l|--[no|auto]lock)] [--[no|confirm]merge] [--[no|confirm]overwriteChanged] [--[no|confirm]downgradeOnLockConflict] [----[no|confirm]moveLock] [(-r *rev*|--revision=*rev*)] [(-u|--unlock)] [--[no]update] [--[no|un]expand] [-f] [--[no|confirm]overwriteDeferred] [--[no]restoreTimestamp] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-S *sandbox*|--sandbox=*sandbox*)] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] [--[no|confirm]retainbinaryfiles] sandbox member...

Description

`si co` checks out members, typically for modification. For example,

```
si co c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If no members are specified, `si co` applies to all Sandbox members.

Check out is an operation that extracts the contents of a revision in an archive and copies it to a working file. You can check out any revision by specifying either its revision number or label. By default, the member revision is used. If an existing revision other than the head revision is specified, a branch from that revision is created if you specify to do so.

Use the si resync command to get a read-only copy of a member in your project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--mergeType=[confirm|cancel|automatic|manual]`

  specifies how you want merge changes from the working file in your Sandbox into the working file that will be created upon check out.

  `--mergeType=confirm` prompts you to confirm a merge type.

  `--mergeType=cancel` cancels the selected merge.

  `--mergeType=automatic` completes the merge process without launching the Visual Merge tool.

  Note

  While Integrity and Visual Merge are capable of performing automatic merging, PTC cannot guarantee that the merged results are \u201ccorrect\u201d. PTC recommends that you examine and test the merged results before checking them into the repository.

  `--mergeType=manual` completes the merge operation through the Visual Merge tool or a third party merge tool, depending on your Difference and Merge Tool preferences. The merge tool launches, displaying the revisions you want to merge. For more information on the Visual Merge tool, refer to the *PTC Integrity User Guide*.
- `--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]`

  specifies what to do when conflicts occur during a merge.

  `--onMergeConflict=confirm` prompts you to confirm a merge conflict option.

  `--onMergeConflict=cancel`cancels the merge.

  `--onMergeConflict=mark` marks the revisions indicating that merging is required without completing all of the merge related tasks. This provides time you may need to investigate conflicts or consult on editing or difference blocks before finishing the merge.

  `--onMergeConflict=launchtool` launches the Visual Merge tool to resolve the conflicts, all non-conflicting blocks will already have been applied.

  `--onMergeConflict=highlight` indicates conflicts in the file with the following characters: "<<<<<<" and ">>>>>>".

  `--onMergeConflict=error` indicates merge conflicts with an error message prompt.

  Note

  `--onMergeConflict=launchtool` does not require the `-g` or `--gui` options.

- `-l`

  `--[no|auto]lock`

  controls whether to create locks on members.

  `--lock` means always lock the member.

  `--nolock` means never lock the member.

  `--autolock` means lock the member based on the locks policy. For information on the locks policy, contact your administrator.
- `--[no|confirm]lockOnFreeze`

  controls whether to lock the specified member even if it is frozen.

  `--lockOnFreeze` means always lock the specified member.

  `--nolockOnFreeze` means never lock the specified member.

  `--confirmlockOnFreeze` means always ask before locking the specifies member.
- `--[no|confirm]merge`

  controls whether to merge changes from the working file in your Sandbox into the working file that will be created upon check out.

  `--merge` means always merge.

  `--nomerge` means never merge.

  `--confirmmerge` means always ask before merging. When asked to confirm the merge, you have the option of specifying `y`, `n` or `d`. Specifying `d` displays the differences between the member's working file and the revision that is being checked out.

  Note

  - Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked
  - By default, Integrity prompts you for the type of merge (manual or automatic) to perform and what action to take if a conflict is encountered.

- `--lockType=[exclusive|nonexclusive|auto]`

  specifies the type of lock obtained on checkout.

  `--lockType=exclusive` obtains an exclusive lock on the member. Exclusive locks enable only one member at a time to lock a specific revision.

`--lockType=nonexclusive` obtains a non-exclusive lock on the member. Non-exclusive locks enable multiple users to check out the same revision for editing.

`--lockType=auto` obtains a lock on the member based on the locks policy. For information on the locks policy, contact your administrator. If the locks policy does noe require a lock, but the `--lock` option is set, a non-exclusive lock is obtained.

- `--[no|confirm]downgradeOnLockConflict`

  controls whether to downgrade your lock to non-exclusive if another user has an exclusive lock on the revision.

  `--downgradeOnLockConflict`means always downgrade.

  `--nodowngradeOnLockConflict` means never downgrade.

  `--confirmdowngradeOnLockConflict` means always ask before downgrading.

- `--[no|confirm]moveLock`

  moves any lock you have on a revision in the same development path to the member revision, including the change package associated with the lock operation. Since you can only have one lock per member per development path, if you already have another revision locked, you need to move that lock to the member revision in order for the check in to succeed. The `downgradeOnLockConflict` option determines what happends if the member revision already has an exclusive lock. You also need to move your lock if it is associated with a different change package than the one you are using for the check out operation.

  `--moveLock` means always move the lock.

  `--nomoveLock` means never move the lock.

  `--confirmmoveLock` means always ask whether to move the lock.

- `-u`
- `--unlock`

  keeps the member unlocked, and is equivalent to `--nolock`.

- `--[no]update`

  controls whether to update the project's member revision to the checked out revision.

Note
In a variant Sandbox, specifying `si co --noupdate --branchVariant` *`member`* updates the member revision.

- `--[no|un]expand`

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

```
$Author: Warner, Carrie (cwarner) $
$CompanyInfo$
$Date: 2015/11/30 13:20:32EST $
$Header: si_co.dita 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner) Exp  $
$Id: si_co.dita 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner) Exp  $
$Locker:  $
$Log: si_co.dita  $
Revision 1.2 2015/11/30 13:20:32EST Warner, Carrie (cwarner)
XML tagging fixes
Revision 1.1 2015/10/29 10:24:48EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectLabel:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$RCSfile: si_co.dita $
$Revision: 1.2 $
$SandboxSetting  $
$Setting  $
$Source: si_co.dita $
$State: Exp $
```

- `-f`

  overwrites the working file if it has changed since it was last checked in. This is equivalent to specifying `--overwriteChanged`.

- `--[no|confirm]overwriteChanged`

  controls whether to overwrite the working file if it has changed since it was last checked in.

  `--overwriteChanged` means always do it

  `--nooverwriteChanged`means never do it.

  `--confirmoverwriteChanged` means always ask before doing it. When asked to confirm overwriting the working file, you have the option of specifying `y`, `n` or `d`. Specifying `d` displays the differences between the member's working file and the revision that is being checked out.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or`n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no|confirm]overwriteDeferred`

  controls whether to overwrite the working file if there is a deferred drop or update revision operation on the member.

  `--overwriteDeferred` means always do it.

  `--nooverwriteDeferred` means never do it.

  `--confirmoverwriteDeferred` means always ask before doing it. When asked to confirm overwriting the working file, you have the option of specifying `y`, `n` or `d`. Specifying `d` displays the differences between the member's working file and the revision that is being checked out.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no]restoreTimestamp`

  controls whether to restore the timestamp of the working file to the timestamp of the revision in the member history. If this is not specified, the timestamp is set to the current date and time.

- `--[no|confirm]retainbinaryfiles`

controls whether a confirmation message appears while checking out a modified binary file.

`--retainbinaryfiles` the modified binary file is checked out, without any confirmation message.

`--[no]retainbinaryfiles` the modified binary file is not checked out, without any confirmation message.

`--[confirm]retainbinaryfiles` a confirmation message appears, asking if you want to retain the changes to the modified binary file.

- *sandbox member...*
  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si closecp, si cpissues, si createcp, si diff, si edit, si lock, si resync, si rlog, si unlock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si configuresandbox

configures Sandbox properties

Synopsis

`si configuresandbox [--lineTerminator=`*`[lf|cr|crlf|native]`*`] [--[no]sparse] [--scope=`*`[subproject:subprojectConfigurationPath|attribute:name[=value]|path:expression|name:expression|type:text|binary|memberrevlabellike:expression|anyrevlabellike:expression|any]`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] sandbox location...`

Description

Integrity provides a way to configure the following Sandbox properties: line terminator, sparse settings, and Sandbox scope. For example,

```
si configuresandbox --lineTerminator=cr --sparse --scope=name:*.java c:/Aurora_Program/bin/Libra/project.pj
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `---lineTerminator=`*`[lf|cr|crlf|native]`*

  explicitly identifies the characters to use as the line terminator for this Sandbox. `lf` is a line feed character, `crlf` is the combination of the carriage return and line feed characters, and `native` uses the default line terminator in the client operating system: `crlf` in Windows, `cr` in Mac OS, and `lf` in UNIX.

  Note

  When you create a Sandbox Integrity remembers the setting of the line terminator, and all checkouts (using si co) and resyncs (using si resync) in that Sandbox use the same line terminator setting. If you want to override the line terminator on an individual member, you must use some utility such as the MKS Toolkit `flip` command.

- `--[no]sparse`

  controls whether or not to make the Sandbox sparse.

  Making the Sandbox sparse with `--sparse` affects the way the Sandbox behaves with the si ci check in command and the si co check out command, as well as si resync. The intent of a sparse Sandbox is that it contains only those working files that are actively being worked on by you at the time. Using si ci to check in a member to a sparse Sandbox removes the working file from the Sandbox (unless you did a check in locked, using the `-l` option). Similarly, using si resync on the Sandbox will by default remove the working files for those members which are not locked by you, although you can override this with si resync `--populate` in a sparse Sandbox, which in effect causes the resync to behave like it does in a "normal", non-sparse Sandbox.

- `--scope=`*`[subproject:subprojectConfigurationPath|attribute:name[=value]|path:expression|name:expression|type:text|binary|memberrevlabellike:expression|anyrevlabellike:expression|any]`*

  defines what subprojects, members, or both subprojects and members are included in the Sandbox, and transfers specific members from the Integrity Server to the Sandbox directory. For a more detailed overview of scoped Sandboxes, see the si createsandbox command.

  You can include or invert (`!`) one or more of the following options:

  - `--scope=subproject:`*`subprojectConfigurationPath`* specifies one or more subprojects, where the subproject name is relative to its parent. Subproject scope cannot be inverted.

    To avoid potential ambiguities, the subproject name must specify the subproject using a Sandbox-relative source configuration path. PTC recommends that you use explicit non-compact keyword string syntax to match the path definition defined in the GUI instead of using compact hash syntax.

    To specify a logical OR operator, separate each option with a comma. For example, for project/project.pj, include subprojects with `--scope=subproject:s#=sub1/project.pj`, `subproject:s#=sub2/project.pj` to define the scope as including sub1 or sub2. This option is case-sensitive. Joining subproject scope clauses with a logical AND operator is not supported.

  - `--scope=attribute:`*`name`*`[=`*`value`*`]` specifies project members with an attribute or an attribute set to a value. For example, include members with `--scope=attribute:Beta` or do not include members with `--scope=!attribute:OS=Windows`. This option is case-sensitive.

  - `--scope=path:`*`expression`* specifies project members that reside in a directory, relative to the top-level Sandbox, for example, `watch/lib/*`. The specified path does not differentiate between subdirectories and subproject names. This means that you cannot specify individual co-located subprojects.

    For example, if you create a scoped Sandbox from the following top-level project:
    `/p1/project.pj`

    with the following subprojects and members:

    `/p1/sub1/project.pj`

    `/p1/sub1/aa.txt`

    `/p1/sub1/bb.txt`

    `/p1/sub1/dd.txt`

    `/p1/sub2/project.pj`

    `/p1/sub2/sub1/cc.txt`

    specifying `--scope=path:sub1` matches `/p1/sub1/aa.txt` and `/p1/sub1/dd.txt` or specifying `--scope=path:*sub1` matches `/p1/sub1/aa.txt`, `/p1/sub1/bb.txt`, `/p1/sub2/sub1/cc.txt`, and `/p1/sub1/dd.txt`.

    Note

    If the client OS is a case-sensitive file system and the database repository on the server is case-sensitive, the `--scope=path:`*`expression`* and `--scope=name:`*`expression`* options are case-sensitive. Otherwise, these options are case-insensitive.

  - `--scope=name:`*`expression`* specifies project members with a name or file extension. A name is only valid for a file name, not a leading directory prefix. For example, include members with `--scope=name:Readme.txt` or do not include members with `--scope=!name:*.java`.
  - `--scope=memberrevlabellike:`*`expression`* specifies project members with a label at member revision. For example, include members with `--scope=memberrevlabellike:TEST` or do not include members with `--scope=!memberrevlabellike:TEST`. This option is case-sensitive and accepts wildcards (* and ?).
  - `--scope=anyrevlabellike:`*`expression`* specifies project members with a label at any revision. For example, include members with `--scope=anyrevlabellike:PROD` or do not include members with `--scope=!anyrevlabellike:PROD`. This option is case-sensitive and accepts wildcards (* and ?).
  - `--scope=type:binary|text` specifies project members that are a binary or text archive type. For example, include members with `--scope=type:binary` or do not include members with `--scope=!type:text`.
  - `--scope=any` specifies all project members. This is the default setting.

  Using Logical Operators to Define Sandbox Member Scope

  You can also combine Sandbox member scope selections using logical AND or OR operators, and invert Sandbox scope selections by preceding each member selection with a ! character. An AND operator is specified as a `--scope` option. For example, to transfer project members with the member attributes `Alpha` AND `Beta`, specify `--scope=attribute:Alpha --scope=attribute:Beta`. An OR operator is specified as a comma between values in a `--scope` option. For example, to transfer project members with member attributes `Alpha` OR `Beta` OR `GA`, specify `--scope=attribute:Alpha,attribute:Beta,attribute:GA`. You can combine logical operators to create more complex Sandbox scope selctions. For example, to transfer project members with member attributes `Alpha` OR `Beta` OR `GA` AND name `*.java`, specify `--scope=attribute:Alpha,attribute:Beta,attribute:GA --scope=name:*.java`.

  Tip

  Using the `si createsandbox` and `si configuresandbox` commands, you can create and edit more complex Sandbox member scope definitions than the GUI using a combination of logical AND or OR operators; however, these definitions may not always be editable from the GUI. If you attempt to edit a complex member scope definition from the GUI, Integrity truncates the definition to what the GUI is capable of displaying. If you attempt to edit a complex scope definition using the `si configuresandbox -g/gui` command, Integrity displays a warning message that choosing to edit the member scope definition removes the options the GUI is not capable of displaying.

- *`sandbox location...`*

  specifies the path of the Sandbox directory including the project file. For example, `C:\sourcecode\framework\scripts\project.pj`.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createsandbox, si importsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si configuresubproject

configures a subproject's type

Synopsis

si configuresubproject [-r *subproject revision*|--subprojectRevision=*subproject revision*] [--subprojectDevelopmentPath=*subproject development path name*|--variant=*subproject development path name*] [--[no]failOnAmbiguousProject] [--type=[normal|variant|build|default]] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] [--cpid=*ID*|--changePackageId=*ID*] [--[no|confirm]closeCP] [--issueId=*value*] *subproject* or *subsandbox*

Description

Once you have created or added a subproject, you can modify the type to suit your needs. For example, you can change a normal subproject to a build subproject to suspend development, or you can change a variant subproject to a normal subproject to continue development on the main trunk.

⚠ Caution

Any changes you make when reconfiguring a subproject affects the project as a whole and any shared subprojects. More specifically, changing the type, revision, or development path of a subproject can radically change the list of members of that subproject. After configuring a subproject, existing sandboxes whose contents were in sync with the subproject before it was configured will display deltas when you use si viewproject and si viewsandbox, reflecting the following possible differences:

- Members that exist in both the old and new version of the subproject display a delta if the working revision in the Sandbox is different from the new member revision.
- Members that are in the new version of the subproject but were not in the old version of the subproject display a "no working file" delta, indicating that the Sandbox does not have a copy of the new member yet.
- Members that were in the old version of the subproject but are not in the new version display as former members.
- Subprojects that were in the old version of the subproject but are not in the new version will display as former subprojects.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-r=`*`subproject revision`*
- `--subprojectRevision=`*`subproject revision`*

  specifies the checkpoint number (for build subprojects). For example, `1.5`. This option is used with `--type=`*`build`*.
- `--subprojectDevelopmentPath=`*`subproject development path name`*
- `--variant=`*`subproject development path name`*

  specifies the development path name (for variant subprojects). For example, `Service_Pack3`. This option is used with `--type=`*`variant`*.
- `--type=[normal|variant|build|default]`

  specifies the new configuration type for the subproject.
  - `--type=normal` configures the subproject to the master (non-variant) version of the subproject.
  - `--type=variant` configures the subproject based upon a specific development path. The option is used with `--variant=`*`subproject development path name`* or `--subprojectDevelopmentPath=`*`subproject development path name`*.
  - `--type=build` configures the subproject as a static subproject based upon a specific checkpoint of the master project that is used for building or testing the project, but not for further development. This option is used with `-r `*`subproject revision`* or `--subprojectRevision=`*`subproject revision`*.
  - `--type=default` configures the subproject based on the type that is consistent with the parent project that you are adding the subproject to. For example, if you add a subproject to a normal project, the subproject is added as a normal type. For information on what the default type is, see your administrator.
- `--cpid=`*`ID`*
- `--changePackageId=`*`ID`*

  identifies a change package that is notified of this action, for example, `1452:1`. Note the following about using this option:
  - This option can only be specified if change packages are enabled.
  - If the integration is enabled, but it is not mandatory to specify a change package, or if no change package is applicable, you can specify `--changePackageId=`*`:none`*.
  - The next time you are prompted for a change package ID, the last used change package ID is displayed by default, if *`:none`* was specified or at least one change package was successfully updated from the last operation.
- `--[no|confirm]closeCP`

  closes the change package after command completion.
- `--issueId=`*`value`*

  specifies the issue ID that corresponds to the change package that records the changes.
- *`subproject`*
- *`subsandbox`*

  specifies the subproject or sub Sandbox name to which the new configuration applies, for example, `tools.pj`

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addsubproject, si checkpoint, si createproject, si createsandbox, si createsubproject, si drop, si dropproject, si projectinfo, si projects, si sharesubproject, si movesubproject, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si connect

establishes a connection to an Integrity Server

Synopsis

```
si connect [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)]
[(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si connect` establishes a connection to an Integrity Server host. Most commands implicitly connect to the host, this does so explicitly. For example

```
si connect --hostname=abcFinancial --port=7001 --user=jriley --password=jriley
```

connects to the *abcFinancial* server, logged in as *jriley*.

In fact, all the other commands call `si connect` to establish the connection. The client supports multiple connections to multiple servers.

Note

You can use si disconnect to disconnect from an Integrity Server host.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=`*`server`*

  identifies the name of the host server where the Integrity Server is located.
- `--password=`*`password`*

  identifies the password to use for connecting to the Integrity Server.
- `--port=`*`number`*

  identifies the port on the host server where the Integrity Server is located.
- `--user=`*`name`*

  identifies the user to use for connecting to the Integrity Server. This typically defaults to the name you have used to log into your client machine.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si disconnect, si exit, si servers
- Miscellaneous: diagnostics, options, preferences

si copypolicysection

copies the policy settings from an existing policy section (global or project) to a new project policy section

Synopsis

`si copypolicysection [--source=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] policy section name`

Description

`si copypolicysection` copies the policy settings from an existing policy section (global or project) to a new project policy section. You specify the global or project policies you want to copy and then specify the registered project you want to copy those policies to.

Note

- You can copy policy sections from the global policy section to a new project policy section, or from an existing project policy section to a new project policy section. You cannot copy existing project policy sections to global policies.
- The `ViewPolicy` and `EditPolicy` permissions are required.
- Global policies contains some policies that only apply at the global level, for example, `IntegrityManagerEnabled`. If `:global` is specified as the source, these policies are not copied.

For example, to create a new policy section for `/newProject/project.pj` and copy any policies that are not global-specific from the global policy, type the following:

```
si copypolicysection --source=:global /newProject/project.pj
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--source=`*`value`*

  specifies the policy section to copy.

- *`policy section name`*

  specifies the new project policy section to copy the policy section to, where `policy section name` is of the form:

  - `:global` for a global policy.
  - *`project`* for a project, for example, `/demo/project.pj`.
  - *`project ; development path`* for a development path on a project, for example, `/demo/project.pj;Variant1`.
  - *`development path`* for all projects that are configured on the development path at the top level, for example, `;Variant1`.

  Note

  You may need to escape the ; in your command line environment, for example, enclose it in "" or escape the individual character with a /.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si deletepolicysection, si setpolicysection si viewpolicysection, si viewpolicysections
- Miscellaneous: diagnostics, options, preferences

si cpissues

displays all Integrity issues that are in a valid state to accept new change packages

Synopsis

`si cpissues [--fields=field1[:width1],field2[:width2]...] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]persist] [--quiet]`

Description

`si cpissues` displays all Integrity issues that are in a valid state to accept new change packages. For example,

```
si cpissues --fields=id,summary --persist
```

Only issues assigned to the user are displayed. Executing the command without the `--fields` option displays the valid fields configured by the global Integrity Item Fields policy on the Integrity Server.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format *field1[:width1],field2[:width2]....* Specifying the column `[:width]` (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a `space`.

  The fields available for printing can be one or more of the valid fields configured by the global Integrity Item Fields policy on the Integrity Server.
- `--[no]persist`

  controls the persistence of CLI views.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si createcp, si createcp, si drop, si lock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si createcp

creates a new change package

Synopsis

`si createcp [--description=value] [--issueId=value] [--summary=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si createcp` creates a new change package. For example,

```
si createcp --description="Second attempt at patch fix." --issueId=34332 --summary="Patch Fix For Customer Ajax"
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--description=value`

  specifies a description of the change package being created.
- `--issueId=value`

  specifies the ID of the issue you are creating a change package for. This option can only be specified if Integrity configuration management is enabled with workflows and documents.

  Caution

  Your ability to create a change package is based on the change package creation policy for the type that you want to create a change package for.
- `--summary=value`

  specifies a brief summary of the change package being created.

  Note

  A change package summary is mandatory.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si cpissues, si drop, si lock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si createdevpath

creates a development path

Synopsis

```
si createdevpath [--projectRevision=rev] [--devpath=path] [(-P project|--project=project)] [--onExistingDevpath=
[confirm|sharedevpath|cancel]] [--creationMethod=[full|extendable|build]] [--onLiveConfiguration=[retainLive|createVariant] [--
[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name]
[(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]]
[(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si createdevpath` creates a development path from a specific project checkpoint.

A development path is an identifier given to a new branch of software development. Changes made through the new development path are kept separate from the main development trunk unless you choose to merge them later.

Once you have created a development path, you can open and work in a project on the development path by opening a variant Sandbox using the si createsandbox command.

Integrity allows multiple developers to point to the same development path, each using their own variant Sandbox. In the variant Sandbox, you can see the current state of the project along the development path and the changes made by other developers using it.

By default, when a development path is created for a project, it is also created for all subprojects, reserving the assigned name as a unique identifier and ensuring no two paths can share the same name. The project is also automatically checkpointed when a development path is created

Branching projects is useful for:

- extracting and building from previous versions of a project
- building customized versions of a product
- performing branch development work
- performing post-release maintenance
- fixing defects in previous versions of the product
- testing new features outside of the main development path
- experimenting with research that does not affect regular development

There are main models to follow when branching projects:

- release based branching
- project based branching

For more information on these models, see the *PTC Integrity User Guide*.

Note

You can branch members instead of projects by specifying `--branch` when checking files in. You still may have to merge changes, but the development path remains the same. For details on branching members, see the si ci reference page.

The following is an example for creating a development path:

```
si createdevpath --project=c:/Aurora_Program/bin/Libra/project.pj --projectRevision=1.3 --devpath=DevStream2
```

Managing an Existing Development Path

When creating a development path with the same name as an existing development path on a subproject in the target project, the default behavior is to use the existing development path in the subproject project and silently add it into the new top-level development path. To let you choose the appropriate course of action, specify the `--onExistingDevpath=[confirm|sharedevpath|cancel]` option. For example, if you incorrectly created a development path on a subproject and not the intended top-level project, you can choose to use the existing development path on the top-level project.

Note

- While you can configure the `si createdevpath --onExistingDevpath=[confirm|sharedevpath|cancel]` preferences from the Integrity Client, an administrator can override and enforce a specific preference on the Integrity Server.
- If the existing development path cannot be used with the target project, an error message appears, displaying the configuration path that is incompatible with the target project.
- If specifying `confirm`, you cannot specify to create an extendable development path (an error is returned).

Configuration Options For Live (Non-Build) Subprojects

Depending on your software development needs, a project and its subprojects may be configured similarly or differently. For example, a project and its subprojects may use the same main development path for a major release. In another example, a project and several of its subprojects may use a development path branched off the main development path for a service pack release; however, one subproject may be configured as a build subproject because it contains a set of shared libraries at a specific version that you do not want to change.

When creating a development path for a project, the default behavior is to create a new development path for the project and any subprojects that are configured the same as the parent project. For example, if a parent project and its subprojects are on the mainline development path or they are on a variant development path, a new development path is created for the parent project and its subprojects. This is corresponds to setting `--creationMethod=full` and `--onLiveConfiguration=retainLive`.

If a project contains subprojects that are configured differently than the parent project, the default behavior is to retain the current configuration for those subprojects. For example, if a parent project contains one subproject that uses the same development path as the parent, but another subproject uses a different development path than the parent, a new development path is created for the parent and the subproject on the same development path as the parent. The subproject that uses a different development path than the parent remains untouched, retaining its existing development path.

During the course of development, you may decide that any subprojects that are subject to change (subprojects configured as normal or variant, also known as live subprojects) need to be on the same new development path as the parent project or be configured as build subprojects so that they remain at a specific version, based on the configuration as of the checkpoint that the new development path is created from. For example, if you have a project that is on an existing development path and contains one subproject using the same development path and another subproject using a different development path, you can choose to create a new development path for the parent project and both subprojects. In another example, if a parent project is on an existing development path and contains one subproject that is on a different development path and another subproject that is a build subproject, you can choose to create a new development path for the parent project and configure the existing variant subproject as a build subproject; the existing build subproject remains untouched.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--devpath=`*path*

  identifies the name of the new development path.

  Note

  The following characters are not permitted in a development path name:

  - \n
  - \r

- \t
- :
- [', ']
- #
- ISO control characters
- Ignorable characters in a Java or Unicode identifier
- Any other characters that your administrator has identified using the `mksis.si.restrictedCharsForDevpathName` property. For more information, see "Source Configuration Properties in the Database" in the Integrity Help Center.

- `--projectRevision=`*rev*

  identifies the checkpointed revision number to create the variant against.
- `--onExistingDevpath=[confirm|sharedevpath|cancel]`

  specifies the appropriate course of action if the development path has the same name as an existing development path on a subproject in the target project.
  - `confirm` prompts you to confirm using the existing development path for the target project. Integrity displays the total number of subprojects with the same development path name and the names of the first 10 subprojects.
  - `sharedevpath` uses the existing development path for the target project.
  - `cancel` means cancel the operation and choose a new development path name.
- `--creationMethod=[full|extendable|build]`

  specifies the way to create the development path.
  - `full` creates a full development path in a single locking transaction.
  - `extendable` creates an extendable development path. For information on extendable development paths, see the documentation for si extenddevpath.
  - `build` creates a development path at the root of a project hierarchy. All subprojects are explicitly configured as build and marked as non-extendable.
- `--onLiveConfiguration=[retainLive|createVariant`

  specifies how to treat live subprojects. Live configurations are subprojects that are configured to either normal or to a variant other than the variant you are creating.
  - `retainLive` specifies that the project tree retain its existing live configuration. All live configurations remain live in the new development path.
  - `createVariant` creates the development path on all previously live subprojects.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si createsandbox, si dropdevpath, si memberinfo, si projectinfo, si revisioninfo, si rlog, si extenddevpath
- Miscellaneous: ACL, diagnostics, options, preferences

si createmetricinfo

creates a new metric to be measured

Synopsis

```
si createmetricinfo [--description=value] [--name=value][--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si createmetricinfo` creates a new metric to be measured.

You can use a third party tool to track metrics in addition to the standard metrics tracked by Integrity. Metrics tracked by a third party tool must be defined in Integrity and recorded for a specific project checkpoint. You define a metric by specifying a name and description for it using this command. Both `--name` and `--description` are mandatory.

You can run this command against an existing metric name to replace the description.

To define a metric you need the `Metrics` permission.

Note
Metrics are only supported for database type repositories.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies a name for the metric being created
- `--description=value`

  specifies a description for the metric being created

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si viewmetricsinfo, si calculateprojectmetrics, si viewprojectmetrics, si addprojectmetric
- Miscellaneous:diagnostics, options, preferences

si createproject

creates a new empty project

Synopsis

```
si createproject [--[no]openView] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] project location
```

Description

`si createproject` creates a new, empty project on the Integrity Server in a specified directory. For example,

```
si createproject c:/Aurora_Program/bin/Libra/project.pj
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]openView`

  controls whether to open the project view after this new project is registered on the server. If `--noopenView` is used, the project view does not open.

- *project location*

  identifies a location on the Integrity Server for the new project. Regardless of your operating system, all paths are specified here using forward slashes (/), and are server absolute paths. You must also specify the name of the project file as part of the location; typically this is `project.pj`.

Note

By convention, you should name your new project file with a suffix of `.pj`. The project file is a virtual file, and it does not appear in the file system for Sandboxes.

Intermediate directories on the Integrity Server are created for you. Remember, however, that server-side restrictions may exist controlling where you are permitted to locate any projects. Contact your system administrator for permitted locations.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si createsandbox, si createsubproject, si dropproject, si projects, si viewproject
- Miscellaneous:ACL, diagnostics, options, preferences

si createsandbox

creates a new Sandbox on your local machine

Synopsis

```
si createsandbox [(-R|--[no|confirm]recurse)] [--
scope=[subproject:subprojectConfigurationPath|attribute:name[=value]|path:expression|name:expression|type:text|binary|memberrevlabellike:expression|anyrevlabellike:expression|any]]
[--lineTerminator=[lf|cr|crlf|native]] [--[no]failOnAmbiguousProject] [--[no]populate] [--[no]sparse] [--[no]openView] [(-Pproject|--project=project)] [--devpath=path] [--
projectRevision=rev] [--[no]awaitServer] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--
no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] directory
```

Description

`si createsandbox` creates a new Sandbox on the client machine in a specified directory. When you want to work with a project, you create a Sandbox. A Sandbox resides on your local drive and is a mirror image of the project. Working in a Sandbox allows you to operate in your own workspace without interfering with the work of others. Once a Sandbox is created, you can add files or members to that Sandbox. The project is then updated to reflect the addition of new project members.

Different types of sandboxes are available for different types of development. *Normal* sandboxes are useful for the sequential development of a project over the long or short term. *Variant* sandboxes are useful for branching off the main development path. *Build* sandboxes are useful for testing a specific checkpoint of the project.

Using the Current Configuration

If you want to create a new Sandbox based on a subproject, then by default, the `si createsandbox` command uses the current configuration of that project, whether that is normal (the mainline), variant, or build.

Using Normal (Mainline)

You can force the new Sandbox to reference the mainline by specifying the configuration for the project and appending the special token `#n=` to the configuration path. This effectively forces a jump to the mainline and is the best method for ensuring a well-formed development path.

For example, assume the following project structure:

- Top level project `/Top` on the mainline
- Subproject `/Top/Sub` on the devpath

To create a subproject Sandbox on the development path:

```
si createsandbox --project="#/Top/Sub" sandbox
```

To create a subproject Sandbox on the mainline:

```
si createsandbox --project="#/Top/Sub#n=" sandbox
```

Note

If the existing project configuration path is sufficiently complex (for example, if it contains explicit jumps between development paths), then appending `#n=` may not work, and you may need to provide a different, less complex configuration path to the mainline.

Using Variant Sandboxes

A variant Sandbox is based on a specific checkpoint of your project that has been branched off the main development path. When you create a variant Sandbox, you choose the development path to use. For example:

```
si createsandbox --devpath=MyVariant
```

In the variant Sandbox, you can see the current state of the project along that development path and the changes made by other developers using it. When a variant Sandbox is created for the first time, it is also created for all subprojects, reserving the assigned name as a unique identifier and ensuring no two paths can share the same name. For more information on creating development paths, see si createdevpath

Using Build Sandboxes

After major milestones, such as product releases, you might want to recreate a static version of an entire project as it existed at some point in the past. You create a build Sandbox, to build or test the project, not to begin further work along a new development path.

A build Sandbox is associated with a particular project checkpoint, and has no development path (since it is static and not meant for further development). For example:

```
si createsandbox --projectRevision=1.34
```

creates a *build* Sandbox based on the project checkpoint 1.34, and it cannot be modified.

With a build Sandbox, you cannot:

- check out, lock, or check in members
- remove or add members
- freeze or thaw members
- checkpoint the master project
- modify project or member attributes
- revert members
- set the member revision

However, with a build Sandbox, you can:

- change labels and states (see si addlabel)
- resynchronize your Sandbox (see si resync)
- compare a member revision in the build Sandbox to another revision (see si diff)
- merge a member revision (si merge) in the build Sandbox with another revision (of course, you cannot check that merged file back into the build Sandbox)
- check for differences between project checkpoints, (see si mods)

Specifying the Sandbox Scope

For large projects with mixed content, such as source code, simulation files, calibration files, and documentation, creating complete Sandboxes from these projects can impact Integrity Client performance and consume large amounts of network bandwidth. While you may need all components in the projects (subprojects), you may only need to work with certain subprojects, files, or file types.

Specifying a Sandbox scope allows you to define what subprojects, project members, or both subprojects and members are included in a Sandbox. A Sandbox scope transfers specific subprojects and/or members from the Integrity Server to the Sandbox directory when the Sandbox is created, and controls what initially displays in the Sandbox view. Specifying Sandbox scope can greatly improve the performance of the Create Sandbox operation, reduce network traffic, and make it easier to locate the relevant subprojects and/or members that you need to work with.

Changes to the scope definition are automatically reflected in the Sandbox view. Out of scope subprojects are automatically hidden from the view unless the corresponding folders exist on disk. Members with working files in the Sandbox that do not match the scope definition display deltas to indicate that they are out of scope. Performing a `si resync` or `si revert` removes the members and the out of scope subprojects from the Sandbox.

Note the following:

- When you define both subproject scope and member scope for a Sandbox, the member scope definition is combined with the subproject scope definition using a logical AND operation.
- The scope of a Sandbox is saved with the Sandbox and persists when the Sandbox or Integrity Client are closed and restarted. It is possible to have two Sandboxes of the same project with each one having a different Sandbox scope.
- For future Create Sandbox operations, you can use the si setprefs command to set the Sandbox scope for the `si createsandbox` command. The syntax for setting scope preferences is `si setprefs --command=createsandbox scope="`*scopepreference*"`.To specify a logical OR operator, separate the preferences with a comma. To specify a logical AND operator, type each scope preference on a separate line with no spaces before them.
- Defining Sandbox scope is different from filtering members using the view filters using the si viewsandbox command. Sandbox scope determines what subprojects and members are initially included in the Sandbox and are transferred to the file system when you resynchronize. View filters allow you to further filter which members are displayed in the Sandbox view. For example, if you create a scoped Sandbox that includes members with attribute name/value `OS=Windows`, your Sandbox only includes members that match that scope. You can then apply the **Locked Members** filter to your Sandbox to display only members that match the Sandbox scope and have a lock. Members of the project that have a lock but do not match your scope are not visible in the view.
- When you add new members that do not match the current Sandbox scope, the new member is added and the new member displays as an out of scope member in the Sandbox. However, when you add new subprojects using the Integrity client that do not match the Sandbox scope, the new subproject is not visible in the Sandbox view because the folder is not created on disk. A new folder is created on disk and visible in the Sandbox view when you create a new subproject using the CLI.
- Other operations can result in subprojects or members in the Sandbox that do not match the current Sandbox scope. For example, you can resynchronize change packages using `--resyncIfOutOfScope` to resynchronize both in scope and out of scope subprojects and members. You can also modify a member so that it no longer matches the Sandbox scope, or update the Sandbox scope definition.
- For commands that include the `--filter` option, you can specify subprojects or members that do not match the current Sandbox scope definition. For example, you can display the subprojects and/or members in your Sandbox that do not match the current scope (`si viewsandbox --filter=outofscope`), or resynchronize subprojects and/or members in your Sandbox that do not match the current scope (`si resync --filter=outofscope`).
- The `--scope` and `--[no]sparse` options are independent of one another. For example, you can create a sparse scoped Sandbox.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--scope=`
  [*subproject:subprojectConfigurationPath|attribute:name[=value]|path:expression|name:expression|type:text|binary|memberrevlabellike:expression|anyrevlabellike:expression|any*]

  defines what subprojects, members, or both subprojects and members are included in the Sandbox, and transfers specific members from the Integrity Server to the Sandbox directory.

  You can include or invert (`!`) one or more of the following options:

  - `--scope=subproject:`*subprojectConfigurationPath*

    specifies one or more subprojects, where the subproject name is relative to its parent. Subproject scope cannot be inverted.

    To avoid potential ambiguities, the subproject name must specify the subproject using a Sandbox-relative source configuration path. PTC recommends that you use explicit non-compact keyword string syntax to match the path definition defined in the GUI instead of using compact hash syntax.

    For example, the following source configuration path:

    `#s=sub/project.pj#s=sub2/project.pj#s=sub3/project.pj`

    is the same as the following compact source configuration path:

    `#sub/sub2/sub3 or #sub#sub2#sub3`

however, the non-compact keyword string syntax is recommended to account for cases where project paths are not necessarily well-formed.

When specifying a Sandbox-relative source configuration path using keyword-based strings, the following keywords are not supported: `#b/#build`, `#d/#devpath`, or `#n/#normal`. See the options reference page for details on specifying source configuration management projects and for a description of all keywords.

To specify a logical OR operator, separate each option with a comma. For example, for project/project.pj, include subprojects with `--scope=subproject:s#=sub1/project.pj`, `subproject:s#=sub2/project.pj` to define the scope as including sub1 or sub2. This option is case-sensitive. Joining subproject scope clauses with a logical AND operator is not supported.

- `--scope=attribute:`*`name`*`[=`*`value`*`]` specifies project members with an attribute or an attribute set to a value. For example, include members with `--scope=attribute:Beta` or do not include members with `--scope=!attribute:OS=Windows`. This option is case-sensitive.
- `--scope=path:`*`expression`* specifies project members that reside in a directory, relative to the top-level Sandbox, for example, `watch/lib/*`. The specified path does not differentiate between subdirectories and subproject names. This means that you cannot specify individual co-located subprojects.

  For example, if you create a scoped Sandbox from the following top-level project:

  `/p1/project.pj`

  with the following subprojects and members:

  `/p1/sub1/project.pj`

  `/p1/sub1/aa.txt`

  `/p1/sub1/bb.txt`

  `/p1/sub1/dd.txt`

  `/p1/sub2/project.pj`

  `/p1/sub2/sub1/cc.txt`

  specifying `--scope=path:sub1` matches `/p1/sub1/aa.txt` and `/p1/sub1/dd.txt` or specifying `--scope=path:*sub1` matches `/p1/sub1/aa.txt`, `/p1/sub1/bb.txt`, `/p1/sub2/sub1/cc.txt`, and `/p1/sub1/dd.txt`.

Note

If the client OS is a case-sensitive file system and the database repository on the server is case-sensitive, the `--scope=path:`*`expression`* and `--scope=name:`*`expression`* options are case-sensitive. Otherwise, these options are case-insensitive.

- `--scope=name:`*`expression`* specifies project members with a name or file extension. A name is only valid for a file name, not a leading directory prefix. For example, include members with `--scope=name:Readme.txt` or do not include members with `--scope=!name:*.java`.
- `--scope=memberrevlabellike:`*`expression`* specifies project members with a label at member revision. For example, include members with `--scope=memberrevlabellike:TEST` or do not include members with `--scope=!memberrevlabellike:TEST`. This option is case-sensitive and accepts wildcards (* and ?).
- `--scope=anyrevlabellike:`*`expression`* specifies project members with a label at any revision. For example, include members with `--scope=anyrevlabellike:PROD` or do not include members with `--scope=!anyrevlabellike:PROD`. This option is case-sensitive and accepts wildcards (* and ?)
- `--scope=type:binary|text` specifies project members that are a binary or text archive type. For example, include members with `--scope=type:binary` or do not include members with `--scope=!type:text`.
- `--scope=any` specifies all project members. This is the default setting.

Using Logical Operators to Define Sandbox Member Scope

You can also combine Sandbox member scope selections using logical AND or OR operators, and invert Sandbox member scope selections by preceding each selection with a ! character. For example, to transfer project members with the member attributes `Alpha` AND `Beta`, specify `--scope=attribute:Alpha --scope=attribute:Beta`. An OR operator is specified as a comma between values in a `--scope` option. For example, to transfer project members with member attributes `Alpha` OR `Beta` OR `GA`, specify `--scope=attribute:Alpha,attribute:Beta,attribute:GA`. You can combine logical operators to create more complex Sandbox scope selections. For example, to transfer project members with member attributes `Alpha` OR `Beta` OR `GA` AND name `*.java`, specify `--scope=attribute:Alpha,attribute:Beta,attribute:GA --scope=name:*.java`.

Tip

Using the `si createsandbox` and `si configuresandbox` commands, you can create and edit more complex Sandbox member scope definitions than the GUI using a combination of logical AND or OR operators; however, these definitions may not always be editable from the GUI. If you attempt to edit a complex member scope definition from the GUI, Integrity truncates the definition to what the GUI is capable of displaying. If you attempt to edit a complex member scope definition using the `si configuresandbox -g/gui` command, Integrity displays a warning message that choosing to edit the scope definition removes the options the GUI is not capable of displaying.

- `--lineTerminator=`*`[lf|cr|crlf|native]`*

  explicitly identifies the characters to use as the line terminator for this Sandbox. *`lf`* is a line feed character, *`crlf`* is the combination of the carriage return and line feed characters, and *`native`* uses the default line terminator in the client operating system: *`crlf`* in Windows, *`cr`* in Mac OS, and *`lf`* in UNIX.

Note

When you create a Sandbox, Integrity remembers the setting of the line terminator, and all checkouts (using si co) and resyncs (using si resync) in that Sandbox use the same line terminator setting. If you want to override the line terminator on an individual member, you must use some utility such as the MKS Toolkit `flip` command.

- `--[no]populate`

  controls whether to populate the Sandbox with read-only working files for all members.
- `--[no]sparse`

  controls whether to create a sparse Sandbox.

  Creating a sparse Sandbox with `--sparse` affects the way the Sandbox behaves with the si ci check in command and the si co check out command, as well as si resync. This option creates a Sandbox with no working files and defers the creation of Sandbox directories and subsandboxes until you need to work with them. A sparse Sandbox does not retain working files when a member is checked in and continues to function this way throughout its use; however, once created, Sandbox directories and subsandboxes remain in the Sandbox. Using si ci to check in a member to a sparse Sandbox removes the working file from the Sandbox (unless you did a check in locked, using the `-l` option). Similarly, using si resync on the Sandbox by default removes the working files for those members that are not locked by you, although you can override this with si resync `--populate` in a sparse Sandbox, which in effect causes the resync to behave like it does in a normal, non-sparse Sandbox.
- `--[no]awaitServer`

  instructs the client to repeatedly attempt the operation if the Integrity Server does not respond.
- `--[no]openView`

  controls whether to open the Sandbox view after this new Sandbox is registered on the client. If `--noopenView` is used, the Sandbox view does not open.
- *`directory`*

  identifies a location on the Integrity client system for the new Sandbox. Use spaces to identify more than one directory.

Note

Do NOT append the name of the project file (typically `project.pj`) to the directory. The name will be calculated automatically.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si checkpoint, si co, si createdevpath, si createproject, si dropsandbox, si importsandbox, si merge, si resync, si sandboxes, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si createsubproject

creates a subproject or adds an existing project as a subproject

Synopsis

```
si createsubproject [ --[no|confirm]reuseDroppedSubproject] [ --[no]createSubprojects] [(-P project| --project=project)] [--
[no]failOnAmbiguousProject] [(-S sandbox| --sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port =number] [--
password=password] [ --user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes )] [--[no]batch] [ --
cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [ --quiet] [--settingsUI=[gui|default]] [ --status=[none|gui|default]] [--
cpid=ID|--changePackageId=ID] [--[no|confirm]closeCP] [--issueId=value] subproject location...
```

Description

`si createsubproject` creates a subproject on the server in a specified directory, which must be under an existing project specified by the `-P project` or `--project=project` options. For example,

```
si createsubproject --project=c:/Aurora_Program/project.pj c:/Aurora_Program/bin/Libra/project.pj
```

This command works with Regular and Variant sandboxes, but not with Build sandboxes.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]reuseDroppedSubproject`

  controls whether to add an existing subproject to the named project. This can be useful if the existing subproject had been dropped from a project.

  Note

  While si addsubproject is the recommended command for adding an existing subproject to a project, this option is available for backwards compatibility and any scripts you may use.

- `--[no]createSubprojects`

  controls whether to create one subproject for each directory encountered when creating subprojects.

- `--cpid=ID`
- `--changePackageId=ID`

  identifies a change package that is notified of this action, for example, `1452:1`.

  Note

  - This option can only be specified if change packages are enabled.
  - If the integration is enabled, but it is not mandatory to specify a change package, or if no change package is applicable, you can specify `--changePackageId=:none`.
  - The next time you are prompted for a change package ID, the last used change package ID is displayed by default, if `:none` was specified or at least one change package was successfully updated from the last operation.

- `--[no|confirm]closeCP`

  closes the change package after command completion.

- `--issueId=value`

  specifies the issue ID that corresponds to the change package that records the changes.

- *`subproject location...`*

  identifies a location on the Integrity Server for the new subproject. Regardless of your operating system, all paths are specified here using forward slashes (/), and may be server relative or absolute paths. You must also specify the name of the project file as part of the location; typically this is `project.pj`.

  Note

  Subprojects must be added "in tree", that is, in the project directory or a subdirectory.

  As with the si createproject command, remember that server-side restrictions may exist controlling where you are permitted to locate any subprojects on the Integrity Server. Contact your system administrator for permitted locations.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si createproject, si createsandbox, si drop, si dropproject, si projectinfo, si projects, si viewproject, si configuresubproject, si addsubproject, si movesubproject, si sharesubproject
- Miscellaneous: ACL, diagnostics, options, preferences

si deactivatedevpath

deactivates a development path

Synopsis

```
si deactivatedevpath [--devpath=value] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [--[no]failOnAmbiguousProject] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

When a development path is no longer active, but you do not want to permanently delete it, you can deactivate the development path. `si deactivatedevpath` enables you to mark a development path as inactive.

Tip

If you want to permanently remove a development path, you can delete it instead. For more information, see "Removing a Development Path" in the Integrity Help Center.

Inactive development paths are filtered out of most views and commands. This command requires that you specify either the mainline project and a development path, or a variant project or variant sandbox.

Once a development project has been deactivated, it cannot be modified. The only actions that you can take are dropping or activating the development path. Change packages for a deactivated development path can be discarded.

Later, you can reactivate the development path if necessary using the si activatedevpath command.

The following is an example for deactivating a development path:

```
si deactivatedevpath --project=c:/Aurora_Program/bin/Libra/project.pj --devpath=DevStream
```

Tip

While you cannot rename a development path directly, you can use the Deactivate action to achieve the same result by performing the following steps:

1. Checkpoint the development path that you plan to deactivate.
2. From the checkpoint that you created, create a new development path using the new name.
3. Copy permissions from the deactivated development path (with the old name) to the new development path (with the new name). For more information, see "Copy Project Permissions" in the Integrity Help Center.
4. Deactivate the development path that you want to rename.

Optionally, if you want to reuse the old development path name on the same project, remove the deactivated development path. For more information, see "Removing a Development Path" in the Integrity Help Center.

Before Deactivating a Development Path

Before you deactivate a development path, you must ensure that there is no work in progress and restrict any further work on the variant project:

1. Identify and remove any change packages, locks, and pending operations:
   - Ensure that there are no locks that are not under change packages using the si projectlocks command.
   - If you use change packages, do the following:
     - Confirm that there are no open change packages for the variant project using the si viewcps command:

       ```
       si viewcps --filter=variant:devpathname
        [--filter=project:project].
       ```

     - Verify that there are no pending operations using the si viewproject command:

       ```
       si viewproject --filter=pending --filtersubs --project=project
       [--devpath=devpathname]
       ```

2. Restrict the variant project to ensure that no more changes are made using the si restrictproject command.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--devpath=`*value*

  the name of the development path to deactivate.
- `--P`*project*
- `--project=`*project*

  the name of the target project. You must specify either this option or `--sandbox`; the two options are mutually exclusive.
- `--S`*sandbox*
- `--sandbox=`*sandbox*

  the name of the sandbox, which can be used to redirect to a project. You must specify either this option or`--project`; the two options are mutually exclusive.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si activatedevpath, si archiveinfo, si createsandbox, si dropdevpath, si memberinfo, si projectinfo, si revisioninfo, si rlog
- Miscellaneous: ACL, diagnostics, options, preferences

si deletearchive

deletes archives from the database repository

Synopsis

```
si deletearchive [--commit] [--[no]deleteIfInUse] [--[no]links] [--mark] [--removeUnreferencedDirectories] [--rollback] [--status] [--[no]targets] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]confirm] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-g|--gui)] archive1, archive2...
```

Description

`si deletearchive` deletes archives from the database.

⚠ Caution

The `--dump` option cannot be used with the `si deletearchive` command. Using this command deletes archives without creating backups for them. Archive backup (and restore) is not availble for individual achives; only configuration management projects that include archives. If you require archive backups, use the `si deleteproject` command instead.

The command works through a multi-phase process called a delete session. The delete session is as follows:

1. `Mark` starts the delete session. Prepares to delete the specified archives by marking them as new candidates.

   ⚠ Caution

   At any phase before Commit, you can enter the Rollback phase to discard the target information, and end the delete session with the database unchanged.

   ---

2. `Commit` deletes all of the delete targets identified in the Mark stage, breaks links to deleted objects, and then ends the delete session.

For more detailed information on performing a delete session, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--commit`

  subcommand: Commits the deletion by permanently deleting the marked objects.
- `--[no]deleteIfInUse`

  used with the `--mark` subcommand, and specifies if to delete targets that are still in use. One case when a project or archive is still in use is when an out-of-tree item is referring to it. However, it is beyond the scope of this documentation to provide all possible situations when an archive is still in use.
- `--[no]links`

  used with the `--status` subcommand, and displays a list of all links that are broken upon commit.
- `--mark`

  subcommand: Starts a delete session by marking objects and their dependents as new candidates. The subcommand determines what objects are to be delete targets (archives to be deleted) and what objects are to be kept candidates.

  ⚠ Caution

  The Integrity Server only supports one delete session at a time and additional targets cannot be added with a second invocation of the `--mark` subcommand.

  Kept candidates are members that are not deleted for one of the following reasons:

  - You do not have `DeleteArchive` permission for the members.
  - You used the `--nodeleteIfInUse` option, and the members are referenced by an outside link. The link can be to any revision referenced in another project. The link can also be that there are shared member archives in other projects that are not specified for the purge session.
- `--removeUnreferencedDirectories`

  used with the `commit` subcommand. Allows the administrator to select or skip the automatic removal of unreferenced directories. If the option is not set, no cleanup step occurs. If the option is set, the cleanup step occurs automatically after running `si deletearchive`. Depending on the number of unreferenced directories that are found, this cleanup step can take an extended amount of time. By default, `--removeUnreferencedDirectories` is unset and no cleanup is done for unreferenced directories as part of the `si deletearchive`\u00a0command. This option does not apply for other subcommands, such as `--dump` or `--mark`.
- `--rollback`

  subcommand: Cancels a delete session, leaving the database unchanged. This subcommand can only be used in phases prior to Commit.
- `--status`

  subcommand: displays the status of the current delete session.
- `--[no]targets`

  used with the `--status` subcommand, to display all delete targets and kept reasons.
- `--[no]confirm`

  specifies if to confirm the delete operation.
- *`archive1, archive2...`*

  specifies archives for the current delete session.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si deleteproject
- Miscellaneous: ACL, diagnostics, options, preferences

si deletelabel

removes a label from a member

Synopsis

```
si deletelabel [(-L label|--label=label)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject]
[(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password]
[--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [-R|--[no|confirm]recurse] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si deletelabel` removes a label from one or more members. For example,

```
si deletelabel --label="GA Release" c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If you do not specify any members, the `si deletelabel` command applies to all project members with associated archives, deleting the label from whatever revision the label happens to exist on in the archive. Please note the `--projectRevision` option is not used for identifying the revision to delete labels from.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-L` *label*
- `--label=`*label*

  identifies a label. Labels cannot contain colons(:), square brackets ([ ]), or leading spaces. Additionally, they cannot have the same format as a valid revision number.

Note

- Labels that include spaces must be enclosed by quotes
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addlabel, si addprojectlabel, si deleteprojectlabel, si viewlabels
- Miscellaneous: ACL, diagnostics, options, preferences

si deletepolicysection

deletes a project policy section

Synopsis

```
si deletepolicysection [--[no]confirm] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F
file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet]
[--settingsUI=[gui|default]] [--status=[none|gui|default]] policy section name...
```

Description

If you created a project policy section that you no longer need, you can delete it using the `si deletepolicysection` command.

Note

- The `ViewPolicy` and `EditPolicy` permissions are required.
- Deleted project policies cannot be restored, you can only recreate them.
- While you can always delete a created project policy section, you cannot delete the default global policies section. Global policies can only be edited, viewed, or copied.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies whether or not to confirm the deletion of the policy section. If you do not specify this option, a confirmation message displays by default.
- *`policy section name...`*

  specifies the policy section to delete, where *`policy section name`* is of the form *`project,devpath`*. For example:
  - `:global` for a global policy.
  - *`project`* for a project, for example, `/demo/project.pj`.
  - *`project;development path`* for a development path on a project, for example, `/demo/project.pj;Variant1`.
  - *`development path`* for all projects that are configured on the development path at the top level, for example, `;Variant1`.

  Note

  You may need to escape the ; in your command line environment, for example, enclose it in "" or escape the individual character with a /.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si copypolicysection, si setpolicysection si viewpolicysection, si viewpolicysections
- Miscellaneous: diagnostics, options, preferences

si deleteproject

deletes projects from the database repository

Synopsis

```
si deleteproject [--commit] [--[no]deleteIfInUse] [--dump] [--[no]links] [--mark] [--[no]recurse] [--removeUnreferencedDirectories]
[--rollback] [--status] [--[no]targets] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F
file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]confirm] [--
quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-g|--gui)] project1, project2...
```

Description

`si deleteproject` deletes configuration management projects and their members from the database. All projects and archives are exported to the file system as a backup if the `--dump` subcommand is used.

The command works through a multi-phase process called a delete session. The delete session is as follows:

1. `Mark` starts the delete session. Prepares to delete the specified projects and their members by marking them as new candidates.
2. `Dump` records the list of items marked for purging from the database repository in the current delete session. If projects are marked, the command creates backups of the projects (and all the members, archives, subprojects, and variants that are referenced by them) into backup tables in the database.

Caution

At any phase before Commit, you can enter the Rollback phase to discard the target information, and end the delete session with the database unchanged.

---

3. `Commit` deletes all of the delete targets identified in the Mark stage, breaks links to deleted objects, and then ends the delete session.

For more detailed information on performing a delete session, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--commit`

  subcommand: Commits the deletion by permanently deleting the marked objects.
- `--[no]deleteIfInUse`

  used with the `--mark` subcommand, and specifies if to delete targets that are still in use. One case when a project or archive is still in use is when an out-of-tree item is referring to it. However, it is beyond the scope of this documentation to provide all possible situations when an archive is still in use.
- `--dump`

  subcommand: Dumps the objects marked for deletion in the current delete session. If projects are marked, the command recursively creates backups of the projects in backup tables in the database (\u201cdumping\u201d them). However, the subcommand does not export all of the members, subprojects, and variants that are referenced by those projects; only the in-tree objects.

  Note

  Before using the `--dump` subcommand, you must create backup tables in the database using the `isutil -c cmdbbackupcreate` command.

  To create, remove, or migrate backup tables, use the following commands:
  - `isutil -c cmdbbackupcreate` creates an empty set of backup `cm` tables in the database (this first performs an implicit `runcmdbbackupdestroy` operation).
  - `isutil -c cmdbbackupdestroy` removes the backup tables from the database without creating new tables.
  - `isutil -c cmdbbackupmigrate` migrates the backup tables in the database to the new schema version.

  After creating the backup tables, verify that they were created. For example, search for the `B_CMPROJECT` table in the database.
- `--[no]links`

  used with the `--status` subcommand, and displays a list of all links that are broken upon commit.
- `--mark`

  subcommand: Starts a delete session by marking objects and their dependents as new candidates. The subcommand determines what objects are to be delete targets (projects or archives to be deleted) and what objects are to be kept candidates.

  Caution

  The Integrity Server only supports one delete session at a time and additional targets cannot be added with a second invocation of the `--mark` subcommand.

  Kept candidates are subprojects or members that are not deleted for one of the following reasons:
  - The subprojects or members are not located in the directory tree of any of the projects specified out of tree.
  - You do not have `DeleteProject` permission for the specified project or its development paths.
  - You do not have `DeleteArchive` permission for the members.
  - You used the `--nodeleteIfInUse` option, and the projects are referenced by an outside link. The link can be that the projects are shared subprojects in other projects that are not specified for the delete session. Links also include every past or present subproject and member archive of every branch of the project specified (mainline, variant, or dropped variant).
  - You used the `--nodeleteIfInUse` option, and the members are referenced by an outside link. The link can be to any revision referenced in another project.

  Note

  When later restoring deleted projects, the kept candidates do not maintain their linkages with restored projects. For information on restoring deleted projects, see the *PTC Integrity Server Administration Guide*.
- `--[no]recurse`

  used with the `--mark` subcommand, to specify if to recurse into subprojects. The default is to recurse.
- `--removeUnreferencedDirectories`

  used with the `commit` subcommand. Allows the administrator to select or skip the automatic removal of unreferenced directories. If the option is not set, no cleanup step occurs. If the option is set, the cleanup step occurs automatically after running `si deleteproject`. Depending on the number of unreferenced directories that are found, this cleanup step can take an extended amount of time. By default, `--removeUnreferencedDirectories` is unset and no cleanup is done for unreferenced directories as part of the `si deleteproject` command. This option does not apply for other subcommands, such as `--dump` or `--mark`.
- `--rollback`

  subcommand: Cancels a delete session, leaving the database unchanged. This subcommand can only be used in phases prior to Commit.
- `--status`

  subcommand: displays the status of the current delete session.
- `--[no]targets`

used with the `--status` subcommand, to display all delete targets and kept reasons.

- `--[no]confirm`

  specifies if to confirm the delete operation.

- *`project1, project2...`*

  specifies projects for the current delete session.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si deletearchive
- Miscellaneous: ACL, diagnostics, options, preferences

si deleteprojectlabel

removes a label from a project

Synopsis

```
si deleteprojectlabel [(-L label|--label=label)] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--[no]failOnAmbiguousProject] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [-R|--[no|confirm]recurse]
```

Description

`si deleteprojectlabel` removes a label from an Integrity configuration management project, deleting the label from whatever project checkpoint the label happens to exist on. For example,

```
si deleteprojectlabel --label="GA Release" --projectc:/Aurora_Program/bin/Libra/project.pj
```

Note

The `--projectRevision` option is not used for identifying the project checkpoint to delete labels from; it is the generic option available on all project-based commands that allows you to specify the checkpoint of the build project you want to work with.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-L` *label*
- `--label=`*label*

  identifies a label. Labels cannot contain colons(:), square brackets ([ ]), or leading spaces. Additionally, they cannot have the same format as a valid revision number.

  Note

  Labels that include spaces must be enclosed by quotes.

- `-R`
- `--[no|confirm]recurse`

  specifies if to recursively delete the label from a project and its subprojects.

  Caution

  - The project that the command uses to recurse is the checkpoint where the label is found on the specified project. When performing the recursion, the label is deleted from whichever revision of the subproject the label exists on. Only subprojects that are in this hierarchy are included in the command recursion.
    - When performing a recursion through the subproject hierarchy, if the label cannot be found, the command continues the recursion.
    - If the label cannot be deleted, the command stops recursion for that project; consequently the label is not deleted for that project's child subprojects.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addlabel, si addprojectlabel, si deletelabel, si viewlabels
- Miscellaneous: ACL, diagnostics, options, preferences

si deleterevision

permanently deletes a revision from a member

Synopsis

si deleterevision [--[no]confirm] [--[no]confirminuse] [-f] [--range=*value*] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-P *project*|--project=*project*)] [--[no]failOnAmbiguousProject] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *member...*

Description

si deleterevision deletes an archived revision of a project member. For example,

```
si deleterevision --range=1.2-1.5 --confirm --confirminuse c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

deletes revisions 1.2, 1.3, 1.4, and 1.5 from the archive xml_man without prompting to confirm to the user if the revisions are in use in other projects.

Note

It is recommended that you never delete a revision. Since historical versions of projects may still point at the deleted revision, it may become impossible to recreate old projects.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no]confirm

  controls the display of confirmation messages.
- --[no]confirminuse

  controls whether Integrity warns you about deleting the revision if it is used in other projects. If the revision is used in other projects, Integrity warns you that deleting the selected revision will break the listed items. This option is valid only with the database repository.
- -f

  forces the revision to be deleted, without any confirmation message.
- --range=*value*

  specifies a range of revision numbers. The format of value for a single revision is simply the revision number itself. Separate multiple non sequential revisions with a comma, and use a dash to indicate a sequential range, for example, 1.2-1.5.

  You may also prefix or append a dash to a single revision number, meaning you want to delete from 1.1 to the specified revision, or from the specified revision to the tip. For example, you would specify -1.6 to delete revisions 1.1 to 1.6. Or you could specify 1.2- to delete revisions 1.2 to the tip revision, whatever it may be.

  The revisions are deleted in order from the highest revision to the lowest.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si diff, si merge, si revisioninfo, si rlog
- Miscellaneous: ACL, diagnostics, options, preferences

si demote

demotes the state of a member

Synopsis

`si demote [(-r rev|--revision=rev)] [(-s state|--state=state)] [(-R|--[no|confirm]recurse)] [--devpath=path] [--filter=filteroptions] [(-P project|--project=project)] [--projectRevision=rev] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--hostname=server] [--port=number] [--password=password] [--user=name] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-g|--gui)] [(-N|--no)] [--[no]batch] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)] member...`

Description

`si demote` demotes a project member to a lower state of development. For example,

```
si demote --state=Development c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

A state is a textual description defined by the administrator to classify the condition of a revision in a history. If no state is assigned to a revision at check-in, a default value of `Exp` (for Experimental) is used. See the *PTC Integrity Server Administration Guide* for details on setting up states.

Project members can also be demoted to a predefined lower state of development using this command.

Note

Demoting members is for historical purposes only. To more effectively control project workflow, PTC recommends using the Integrity integration.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-s state`
- `--state=state`

  specifies the target state to be set for the demoted member. If a state is not specified, the member is demoted to the next state.
- *member...*
- identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si demoteproject, si promote, si promoteproject
- Miscellaneous: ACL, diagnostics, options, preferences

si demoteproject

demotes the state of a project

Synopsis

```
si demoteproject [(-s state|--state=state)] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--
[no]failOnAmbiguousProject] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--
user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si demoteproject` demotes a project to a lower state of development. For example,

```
si demoteproject --state=Development --project=c:/Aurora_Program/bin/Libra/project.pj
```

A state is a textual description, defined by the administrator, used to classify the condition of a revision in a history. If no state is assigned to a revision at checkpoint, a default value of `Exp` (for Experimental) is used. See the *PTC Integrity Server Administration Guide* for details on setting up states.

Just as you can with a project member, you can demote the project itself (change its state, in other words). Only the project is demoted, not the members of the project.

Note

Demoting projects is for historical purposes only. To more effectively control project workflow, PTC recommends using Integrity.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-s`*state*
- `--state=`*state*

  specifies the target state to be set for the demoted project. If no state is specified, the project is demoted to the next state.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si demote, si promote, si promoteproject
- Miscellaneous: ACL, diagnostics, options, preferences

si diff

displays differences between two revisions of a member

Synopsis

```
si diff [--context=value] [--[no]ignoreBlanks] [--[no]ignoreCase] [--[no]ignoreWhitespace] [-r rev1[-r rev2]] [--
[no]failOnAmbiguousProject] [(-R|--[no|confirm]recurse)] [--guiCharacterEncoding=value] [--filter=filteroptions] [(-P project|--
project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--
password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si diff` displays the differences between the following types of files:

- any two text-based revisions in a history
- any two symbolic link file revisions in a history
- a text-based revision and its associated working file in a Sandbox
- a symbolic link file revision and its associated working file in a Sandbox

For example,

```
si diff -r 1.3 -r 1.4 --project=c:/Documentation/Man_Pages/xml_man/project.pj si_add.1.xml
```

displays the differences between revision 1.3 and revision 1.4 of `si_add.1.xml`.

If no members are specified, `si diff` operates on all project members.

By default, in a Sandbox, this command compares the member revision against the working file.

If you use the `-g` or `--gui` option, this command invokes the Visual Difference tool. Otherwise, the `diffb` utility is used to evaluate differences in binary files.

Note

A symbolic link is a special type of file that contains a reference that points to another file or directory. When comparing symbolic link file members using the `-g` or `--gui` option, the Visual Difference tool does not display. Instead, a dialog box displays the original target file path and the newer target file path.

If used with `-g` or`--gui`, only one member can be differenced at a time, and the `--context` option is ignored.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--context=`*value*

  shows `value` number of lines of context, before and after each difference between two files. `si diff` marks lines removed from the "old" revision with `-`, marks lines added to the "new" revision with `+`, and marks lines changed in both files with `!`

- `--[no]ignoreBlanks`

  controls whether to ignore blanks when comparing textual revisions. When `--ignoreBlanks` is used, all differences involving blanks are ignored except blanks at the start of a line. These are treated differently than blanks between other characters. The following pair would be considered different, for example, because of the leading blank:

  ```
  a b c def
   a b c def
  ```

- `--[no]ignoreCase`

  controls whether to ignore case when comparing revisions.

- `--[no]ignoreWhitespace`

  controls whether to ignore white space at the end of each line (except the newline) and treat all consecutive strings of white space elsewhere in a line as equivalent (effectively, reducing all strings of white space to a single space for the purpose of comparing lines).

- *-rrev1[-r rev2]*

- uses a specified revision. Specify this option twice to identify the two revisions to be compared; the first instance is the "old" revision, the second is the "new". If the second revision is not specified, the working file is assumed. Both revisions must be specified if operating against a project. `rev` can be a valid revision number or a label. If neither is specified, it defaults to member revision.

- `--guiCharacterEncoding=`*value*

  specifies the character encoding to use for the revision contents in the GUI, and can only be specified with the -g option. Integrity automatically decodes UTF-8 revision contents based on the presence of a byte order mark (BOM) in the file. You can set character encoding preferences for each view and command through your client preferences. The preference setting is used if the character set cannot be determined through the file's BOM, or if no character set is specified in the command line. By default, the character set in the preferences matches the default character set provided by the client's operating system locale. The option does not apply to pure CLI output, or third party merge and/or differencing tools. Possible values are:

  ```
  UTF-8
  US-ASCII
  windows-1252
  ISO-8859 -1
  ISO-8859 -15
  IBM437
  IBM850
  IBM863
  EUC-JP
  Shift_JIS
  x-euc-jp-linux
  x-eucJP-Open
  x-windows-iso2022jp
  IBM862
  ISO-8859-8
  ISO-8859-9
  ```

- *member...*

  identifies a specific project or Sandbox member, depending on whether the `-S sandbox` option or the `-P project` option is specified. To specify more than one member, use spaces between them.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si merge, si mods, si revisioninfo, si rlog, si updaterevision, si viewrevision
- Miscellaneous: ACL, diagnostics, options, preferences

si difffiles

compares the differences between two text files

Synopsis

`si difffiles [--context=value] [--[no]ignoreBlanks] [--[no]ignoreCase] [--[no]ignoreWhitespace] [--guiCharacterEncoding=value] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] file1 file2`

Description

`si difffiles` compares the differences between two text files. This is useful when you want to compare the differences between two arbitrary text files (not members or revisions).

In graphical mode where the `-g` or `--gui` option is used, this invokes a dialog box that allows you to browse for each file. The results appear in Visual Difference.

Note

The dialog box keeps track of the last 10 files in both fields.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--context=value`

  shows *value* number of lines of context, before and after each difference between two files. `si difffiles` marks lines removed from the first specified file with `-`, marks lines added to the second specified file with `+`, and marks lines changed in both files with `!`

- `--[no]ignoreBlanks`

  controls whether to ignore blanks when comparing text files. When `--ignoreBlanks` is used, all differences involving blanks are ignored except blanks at the start of a line. These are treated differently than blanks between other characters. The following pair would be considered different, for example, because of the leading blank:

```
  a b c def
a b c def
```

- `--[no]ignoreCase`

  controls whether to ignore case when comparing files.

- `--[no]ignoreWhitespace`

  controls whether to ignore white space at the end of each line (except the newline) and treat all consecutive strings of white space elsewhere in a line as equivalent (effectively, reducing all strings of white space to a single space for the purpose of comparing lines).

- `--guiCharacterEncoding=value`

  specifies the character encoding to use for the revision contents in the GUI, and can only be specified with the -g option. Integrity automatically decodes UTF-8 revision contents based on the presence of a byte order mark (BOM) in the file. You can set character encoding preferences for each view and command through your client preferences. The preference setting is used if the character set cannot be determined through the file's BOM, or if no character set is specified in the command line. By default, the character set in the preferences matches the default character set provided by the client's operating system locale. The option does not apply to pure CLI output, or third party merge and/or differencing tools. Possible values are:

```
UTF-8
US-ASCII
windows-1252
ISO-8859 -1
ISO-8859 -15
IBM437
IBM850
IBM863
EUC-JP
Shift_JIS
x-euc-jp-linux
x-eucJP-Open
x-windows-iso2022jp
IBM862
ISO-8859-8
ISO-8859-9
```

- *file1 file2*

  identifies the two files you want to compare.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si diff, si ci, si merge, si mods, si revisioninfo, si rlog, si updaterevision, si viewrevision
- Miscellaneous: ACL, diagnostics, options, preferences

si discardcp

discards a change package

Synopsis

`si discardcp [--[no]confirm] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`issue|issue:change package id...`*

Description

`si discardcp` discards a change package. For example,

```
si discardcp --confirm 34343:2
```

Integrity provides a way to remove change packages from active use when they are not needed by discarding them.

Change packages can only be discarded if they are in the Open or Rejected states. In order to discard a change package, that change package must be both created by you and not contain any committed entries. However, a change package administrator may discard a change package created by another user.

When a change package is discarded, the following happens where applicable:

- all uncommitted entries are removed from the change package
- all deferred operations corresponding to deferred entries are reverted
- all locks on members corresponding to lock entries are released
- all work in progress entries are reverted
- all pending operations corresponding to pending entries are reverted, and appear as discarded entries in the review log (to preserve the review history)
- all pending revisions that correspond to pending entries are deleted
- any archives created for pending members associated with pending entries in the change package are deleted from the server (pre-existing archives that were shared to create a pending member are not deleted)

Note the following about discarded change packages:

- The change package ID for the discarded change package can never be used for any future change packages that are created.
- Discarded change packages have a state of Discarded, and can still be viewed using the Change Package filter.
- If the change package has undergone a review, the review log persists.
- Discarded change packages can be opened and used again.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies if to confirm discarding the change package.
- *`issue...`*

  `issue` identifies a specific issue that contains all change packages that you want to discard; use spaces to specify more than one issue.
- *`issue:change package id...`*

  *`issue:change package id`*identifies a specific change package to discard; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si co, si cpissues, si createcp, si drop, si lock, si viewcps, si acceptcp, si rejectcp, si opencp
- Miscellaneous: ACL, diagnostics, options, preferences

si disconnect

disconnects from the Integrity Server

Synopsis

```
si disconnect [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--[no]confirm] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si disconnect` disconnects the client connection to the host Integrity Server. For example

```
si disconnect --hostname=abcFinancial --port=7001 --user=jriley
```

Note

When disconnecting a connection that is the current connection, all open client views close. All new views use the connection specified in the Integrity preferences, or an existing connection, as the new current connection.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm` controls whether to implement the Integrity Server disconnection confirmation policy.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si connect, si servers
- Miscellaneous: diagnostics, options, preferences

si drop

drops a member or subproject from a project

Synopsis

si drop [--cpid=*ID*|--changePackageId=*ID*] [--[no|confirm]closeCP] [--issueId=*ID*] [(-f|--[no]confirm)] [--[no]defer] [--[no]delete] [--filter=*filteroptions*] [--[no]failOnAmbiguousProject] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *member... or subproject...*

Description

`si drop` drops specified members and subprojects from a project. For example,

```
si drop --confirm --nodelete c:/Aurora_Program/bin/Libra/project.pj
```

Files dropped in a Sandbox are not actually dropped from the Sandbox until you use the si resync command, or if you specify `--delete` with this command. This allows Sandbox users to be protected from a file deletion until an appropriate point, and therefore be in control.

Note

This command does not operate in a Build Sandbox.

If a member has outlived its usefulness or just does not belong in a project any more, you can remove it at any time. After you remove a member from a project, the member is no longer listed as part of the Sandbox or master project, but the member history remains, in case you need to recreate an earlier version of the project. The build project and variants may still have the file as a member.

Adding a member (see si add) which has previously been dropped will result in the history of the dropped member becoming the history of the newly added member.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]closeCP`

  controls whether to close the associated change package.

  `--nocloseCP` means do not close the change package.

  `--confirmcloseCP` means ask before closing the change package.

  `--closeCP` always closes the change package.

- `-f`

  `--[no]confirm`

  controls the display of confirmation messages. `-f` forces an action without any prompt.

- `--[no]defer`

  controls whether to delay the drop operation in the project until the deferred operation is submitted. The operation in the Sandbox still takes place immediately.

- `--[no]delete`

  control whether the working file should be deleted immediately from your Sandbox, when using `-S` or `--sandbox`.

- *member... or subproject...*

  identifies a specific member or subproject; use spaces to specify more than one.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si closecp, si cpissues, si createcp, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si dropdevpath

drops a variant project from the master project

Synopsis

```
si dropdevpath [--devpath=path] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [--[no]failOnAmbiguousProject] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si dropdevpath` drops a variant project.

Tip

If you do not want to permanently remove a development path, you can deactivate it instead using the si deactivatedevpath command.

For example:

```
si dropdevpath --forceconfirm=yes --project=c:/Aurora_Program/bin/Libra/project.pj --devpath=ReleaseCycle2
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createdevpath, si createproject, si dropproject, si projectinfo, si projects, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si dropmemberattr

drops an attribute from a member

Synopsis

```
si dropmemberattr [(--attr=key[=value]|--attribute=key[=value])] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si dropmemberattr` drops an attribute from a member. For example,

```
si dropmemberattr --attribute=OS=nt c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If no members are specified, the command applies to all members of the project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=`*`key`*`=`*`value`*
- `--attribute=`*`key`*`=`*`value`*

  identifies the attribute key and, optionally, the value to be dropped. If only the key is given, this command deletes any attribute with that key. If a `value` is given for the key, then only a key with that value is dropped.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addmemberattr, si addprojectattr, si dropprojectattr
- Miscellaneous: ACL, diagnostics, options, preferences

si dropproject

drops (unregisters) a project from an Integrity Server

Synopsis

```
si dropproject [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] project...
```

Description

`si dropproject` unregisters a project from the Integrity Server. When a configuration management project has outlived its usefulness or just does not belong on the Integrity Server anymore, you can remove it. For example,

```
si dropproject --forceConfirm=yes c:/Aurora_Program/project.pj
```

Project paths are referenced using the repository location. This allows you to reference checkpoints in dropped configuration management projects. If a checkpoint is currently referenced and the corresponding project is later dropped, the checkpoint remains accessible as read-only. For example, you can continue to open the referenced checkpoint from an SI project field or create a build Sandbox from the checkpoint for auditing purposes.

Dropped projects can also be accessed as read-only from the `si viewcps` and `si viewlocks` commands until the project is purged or reclaimed by your administrator.

Note

- If a project is dropped and any Sandboxes or variants are associated with it, those Sandboxes or variants no longer function.
- Dropping a configuration management project unregisters it from the Integrity Server, but it does not delete the project. Dropped configuration management projects can be added back into Integrity
- This works only on "top level" projects; to drop a subproject from its master project, use si drop

---

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *project...*

  identifies a specific project location with a fully qualified server-relative path; use spaces to specify more than one.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createproject, si createsubproject, si drop, si projects, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si dropprojectattr

drops an attribute from a project

Synopsis

```
si dropprojectattr [(--attr=key[=value]|--attribute =key[=value])] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)]
[--[no]failOnAmbiguousProject] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--
usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--
gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si dropprojectattr` drops an attribute from a project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=`*`key`*`=`*`value`*
- `--attribute=`*`key`*`=`*`value`* identifies the attribute key and, optionally, the value to be dropped. If only the key is given, this command deletes any attribute with that key. If a *`value`* is given for the key, then only a key with that value is dropped.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addmemberattr, si addprojectattr, si dropmemberattr, si dropsandboxattr, si addsandboxattr
- Miscellaneous: ACL, diagnostics, options, preferences

si dropsandbox

drops a Sandbox

Synopsis

```
si dropsandbox [--[no]confirm] [--delete=[none|members|all]] [-f] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] sandbox location...
```

Description

`si dropsandbox` unregisters a Sandbox from the Integrity Client. For example,

```
si dropsandbox --confirm --delete=none c:/Aurora_Program/project.pj
```

Once a Sandbox is dropped, Sandbox commands do not operate on the Sandbox.

Note

Dropping a Sandbox is permanent. Even if you do not delete the Sandbox files, you cannot import the Sandbox again.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-f`

  `--[no]confirm`

  controls the display of confirmation messages. `-f` forces an action without any prompt.
- `--delete=[none|members|all]`

  deletes files from the client Sandbox directory according to the value specified. If `none` is specified, then no deletion occurs after the Sandbox is dropped. If `members` is specified, then the specified Sandbox, sub Sandboxes, and members are deleted. If `all` is specified, then everything in the current directory and subdirectories is deleted. For example:
- `--delete=`*none* sandbox.pj

  causes sandbox.pj to be removed from the registry, but not deleted from the client directory.
- `--delete=`*members* sandbox.pj

  causes sandbox.pj to be removed from the registry and deleted from the client directory along with all its sub Sandboxes and members.
- `--delete=`*all* sandbox.pj

  causes sandbox.pj to be removed from the registry and all files in the client directory and subdirectories to be deleted.

  Note

  Since `--delete=all` is a destructive operation, it always requires confirmation. This is a preventative security measure. For example, if you create a Sandbox in the root of a file system, this operation would delete your entire drive.

- *sandbox location...*

  identifies the location of a specific Sandbox; use spaces to specify more than one Sandbox.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createsandbox, si importsandbox, si sandboxes, si viewsandbox
- Miscellaneous: diagnostics, options, preferences

si dropsandboxattr

drops Sandbox attributes

Synopsis

```
si dropsandboxattr [--attr|attribute=key=value] [-S|--sandbox=value] [--[no]failOnAmbiguousProject] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]]
```

Description

`si dropsandboxattr` drops Sandbox attributes.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--attr=key=value`
- `--attribute=key=value` specifies the attribute to drop.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addsandboxattr, si configuresandbox, add projectattr, dropprojectattr
- Miscellaneous: ACL, diagnostics, options, preferences

si echo

echoes a string to user interface

Synopsis

```
si echo [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...
```

Description

`si echo` echoes a string to the user interface when used in a client side event trigger for configuration management.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *`string...`*

  specifies a string to display in the appropriate user interface.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: im createtrigger, im edittrigger, im viewtrigger, im runtrigger, im deletetrigger, im triggers
- Miscellaneous: ACL, diagnostics, options, preferences

si edit

opens a revision or working file for editing

Synopsis

`si edit [--editor=value] [(-r rev|--revision=rev)] [--[no]systemEditor] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...`

Description

`si edit` opens a current, former, or deferred member revision for editing. For example,

```
si edit c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

In a Sandbox, by default not specifying the `-r` option edits the working file.

If this command is invoked in the CLI mode (that is, no `-g` or `--gui` option is used), then the `EDITOR` environment variable is used and runs with the name of the temporary file containing the requested revision. If the `EDITOR` variable is not set, or if the `-g` or `--gui` option is used, a system-specific method is used. In particular, on WIN32-based systems, the operating system is instructed to open the file and does so based on the extension association. Under UNIX, the `-g` or `--gui` option uses the `EDITOR` environment variable; if it is not set, then Vi is used.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--editor=value`

  identifies the editor to be used, overriding any `EDITOR` environment variables or system setting.

- `--[no]systemEditor`

  controls whether to use the system editor, set through si setprefs. If set, this overrides any editor setting if `-g` or `--gui` is given. This is used to temporarily override the preference system.

- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si co, si ci, si lock, si unlock, si viewrevision
- Miscellaneous: diagnostics, options, preferences

si editcp

edits an existing change package

Synopsis

`si editcp [--description=value] [--descriptionFile=value] [--state=[open|closed|submitted|accepted|rejected|discarded|commitfailed]] [--summary=value] [--summary=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--quiet] [(-g|--gui)] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue|issue:change package id...`

Description

`si editcp` edits an existing change package. For example,

```
si editcp --state=open 3433:3
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--description=`*value*

  specifies a description of the change package being edited.
- `--descriptionFile=`*value*

  specifies the file that contains the description for the change package.
- `--state=[open|closed|submitted|accepted|rejected|discarded|commitfailed]`specifies the state of the change package. The change package progresses through states as part of the change package review workflow. If reviews are not mandatory, some states are not applicable. For more information, see the *PTC Integrity User Guide*.

  `--state=open` specifies that the change package is open.

  `--state=closed` specifies that the change package is closed.

  `--state=submitted` specifies that the change package has been submitted for review.

  `--state=accepted` specifies that the change package has been accepted by all reviewers.

  `--state=rejected` specifies that the change package has been rejected by at least one reviewer.

  `--state=discarded` specifies that the change package has been discarded.

  `--state=commitfailed` specifies that the change package has failed to commit to the repository.
- `--summary=`*value*

  specifies a brief summary of the change package being edited.
- *issue...*
- *issue:change package id...*

  `issue` identifies a specific issue that contains all change packages that you want to edit; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to edit; use spaces to specify more than one change package.

Diagnostics

See the diagnosticsreference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si discardcp, si opencp, si closecp, si createcp, si add, si ci, si co, si cpissues, si createcp, si drop, si lock, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si exit

exits the current Integrity Client session

Synopsis

```
si exit [--[no]abort] [--[no|confirm]shutdown] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch]
[--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si exit` exits the current Integrity Client session. When you run any Integrity command from the CLI, or when you open the Integrity Client GUI, you start a client session. Only one client session is running at a time, regardless of how many GUI windows you have open or how many CLIs you are using. To close the GUI you use the appropriate menu commands, and to close the CLI you use the `si exit` command. In both cases you may have the further option of shutting down the Integrity client altogether, based on your preferences (see si setprefs); if you do not, the client is still running and available for additional interaction. If you do shut down the client completely, then running any Integrity command from the CLI or opening the Integrity Client GUI starts a new client session.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]abort`

  controls whether to shut down any other Integrity commands that may be running. Some commands allow you to specify a `--persist` option which keeps those commands active during a client session. Using `--abort` with `si exit` is recommended for stopping all persistent views that have been specified with another command's `--persist` option.

- `--[no|confirm]shutdown`

  controls the shutting down of the Integrity client without getting a prompt.

Note

Specifying `--noshutdown` with `si exit` does nothing.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si about, si gui, si setprefs
- Miscellaneous:diagnostics, options, preferences

si extenddevpath

extends an extendable development path

Synopsis

```
si extenddevpath [--extendRecursively] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--
[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name]
[(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]]
[(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si extenddevpath` takes selected build subprojects and extends the closest enclosing development path down to these subprojects. This command, which requires `ExtendDevpath` permission, creates the variant on the specified subproject and its ancestors. The configuration of sibling subprojects is left intact. If every subproject in the tree is included in a development path, an extendable development path is equivalent to a full development path.

For more information, see "Extending an Extendable Development Path" in the *PTC Integrity Help Center*.

Options

This command takes the universal options available to all *si* commands, as well as some general options. See the options reference page for descriptions.

- `--extendRecursively` extends the extendable development path with full recursion below the selected subproject in a multi-transactional way that does not lock the project for other users. When this option is used, the specified subproject can be either a variant subproject or a build subproject.

Notes

To specify the subproject to extend, you can use any of the following methods:

`-P` *`configPathToBuildSubInVariantHierarchy`*

where *`configPathToBuildSubInVariantHierarchy`* is the configuration path to the build subproject in the variant hierarchy

`-P` *`legacyPathToParentProject`*`--devpath` *`legacyDevpath subName/project.pj`*

where *`legacyPathToParentProject`* is the legacy path to the parent project and *`legacyDevpath subName/project.pj`* is the legacy development path subproject name followed by a backward slash and the project name

`-P` *`configPathToParentProject subName/project.pj`*

where *`configPathToParentProject subName/project.pj`* is the configuration path to the parent project followed by the path subproject name, a backward slash, and the project name

If you do not specify the `-P` option, you will be prompted and should specify the configuration path to the subproject.

When using the `-g` option, you need to select the build subproject in the variant hierarchy. The `-g` option does not support selecting multiple subprojects to extend from the GUI.

The optional `--devpath` option and subproject name selection are never prompted. You must specify them upfront if you plan on using them.

Examples of Extending Extendable Development Paths

Extendable development paths can be extended using configuration paths, legacy syntax, and sandbox context. Several examples follow:

To extend an extendable development path from "path1" to "sss1" using a configuration path:

```
si extenddevpath --project=#/demo1#d=path1#s1#ss1#sss1
```

To extend an extendable development path from "path2" to "s1" using legacy syntax:

```
si extenddevpath --project=/demo1/project.pj --devpath=path2 s1/project.pj
```

To extend an extendable development path from "path3" to "s1" in a sandbox context:

```
si extenddevpath --sandbox=c:/sandbox/demo1path3/project.pj c:/sandbox/demo1path3/s1/project.pj
```

To extend an extendable development path from from "path4" to "sss1" in a sandbox context under a sandbox directory:

```
C:\sandbox\demo1path4>si extenddevpath s1/ss1/sss1/project.pj
```

To extend an extendable development path from "path5" to multiple subprojects ("s1" and "s2") that have the same parent:

```
si extenddevpath --project=#/demo1#d=path5 s1/project.pj s2/project.pj
```

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

Commands:

si createdevpath

Miscellaneous:

ACL, diagnostics, options, preferences

si freeze

freezes a project member

Synopsis

`si freeze [(-R|--[no|confirm]recurse)] [--filter=`*`filteroptions`*`] [(-P `*`project`*`|--project=`*`project`*`)] [--[no]failOnAmbiguousProject] [(-S `*`sandbox`*`|--sandbox=`*`sandbox`*`)] [--devpath=`*`path`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`member...`*

Description

`si freeze` freezes one or more project members. For example,

```
si freeze c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If no members are specified, `si freeze` assumes all members of the project. Members can be unfrozen with the si thaw command. When a member is frozen, all Integrity operations are run on the frozen member revision. The member revision and attributes of the frozen member cannot be changed until it is thawed. This ensures that a particular revision is always picked up, even if development on it continues.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *member...*
  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si thaw
- Miscellaneous: ACL, diagnostics, options, preferences

si getdbfile

retrieves a configuration management file from the database

Synopsis

```
si getdbfile [--encoding=value] [--output=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...
```

Description

`si getdbfile` retrieves a configuration management file from the database, such as the `si.properties` file. Although PTC recommends editing the `si.properties` file in the GUI, you can retrieve the `si.properties` file from the database for manual editing. Once you are finished editing this file, you can store it in the database using the `si putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=`*`value`*

  specifies the code set to save the file in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--output=`*`value`*

  specifies the name of the file to store the output to on the local file system.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag=listdbfiles`, for example, `config\properties\si.properties`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: si putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

si gui

starts the Integrity Client graphical user interface

Synopsis

`si gui [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si gui` starts the Integrity Client graphical user interface.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Miscellaneous: diagnostics, options, preferences
-

si importsandbox

imports an existing sandbox

Synopsis

```
si importsandbox [--[no]openView] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [--[no]failOnAmbiguousProject] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] file...
```

Description

`si importsandbox` imports an existing Sandbox and registers it as a new Sandbox on the Integrity Client. For example,

```
si importsandbox c:/Aurora_Program/bin/Libra/project.pj
```

You must correlate this Sandbox with a project that already exists on the Integrity Server, and it is assumed that the project you identify is the project the Sandbox was first created from. A Sandbox must first be registered before further commands can operate on the Sandbox and its members.

Note
This command can only import Sandboxes that were created using Integrity 10.7 or earlier. Once a Sandbox has been opened in Integrity 10.8 or later, it cannot be imported using this command.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]openView`

  controls whether to open the Sandbox view after this Sandbox is imported and registered on the client. If `--noopenView` is used, the Sandbox view does not open.
- *file...*

  identifies an existing Sandbox file (`project.pj`) on the Integrity Client system to be imported; use spaces to specify more than one.

  Note
  Unlike the si createsandbox command, for `si importsandbox` you must include the name of the Sandbox file, which is typically `project.pj`

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si createproject, si dropsandbox, si sandboxes, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si integrations

manages integrations.

Synopsis

`si integrations [--action=`*`[list|enable|enableAll|disable|disableAll]`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`string...`*

Description

`si integrations` enables or disables Integrity integrations. You can select from a list of available IDE integrations and enable or disable them as required to work with your preferred IDE. There is no requirement to re-install the Integrity Client, but if prompted, you may need to reboot your computer or restart the IDE for the integration to take effect.

An Integrated Development Environment, or IDE, is a supported development application, such as Sybase PowerBuilder, that allows you to access Integrity functionality by installing the appropriate extension.

For complete information on using third party integrations with Integrity, see the *PTC Integrity Integrations User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--action =[`*`list|enable|enableAll|disable|disableAll`*`]`
- specifies the action to perform on the integration. An action must be specified. There is no default specification for this option, so to list all of the available integrations, use the `list` flag.

Note

Integrity asks you to restart your computer to complete the integration changes. If you are disabling[enabling] a number of integrations, you do not need to restart between each `si integrations` command. When you have disabled[enabled] all the required integrations, you can restart your machine.

- *string...*

  specifies the integration to perform an action upon.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Miscellaneous: ACL, diagnostics, options, preferences
-

si loadrc

loads the Integrity Client preferences file

Synopsis

`si loadrc [--[no]merge] [--rc=value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-F|--selectionFile=value]`

Description

`si loadrc` loads the user's `IntegrityClient.rc` file, which contains your personal preferences for configuring the Integrity Client. The `IntegrityClient.rc` file is loaded when you start the client; however, this command reloads the file without restarting the client. If for some reason your personal `IntegrityClient.rc` file has changed, this command will reload your preferences. Your preferences file shouldn't change, unless you happen to copy someone else's file or if you happen to restore a backup that you made.

Your administrator can lock certain preferences from the Integrity Server, preventing you from configuring them using the `si setprefs` command. Preferences that are locked -- viewable with `si viewprefs` display (locked) at the end of the output line. The following preferences can be locked from the server by editing Integrity policies in the Integrity Administration Client.

Tip

For information on editing Integrity policies, see the PTC Integrity Server Administration Guide.

Preference
branchIfVariant
breakLock
changePackageID
createBranch
onLock
restoreTimestamp
retainWorkingFile
saveTimestamp
sparse
updateMemberRev

Note

Do not edit the `IntegrityClient.rc` file manually, because preferences that appear more than once in the `IntegrityClient.rc` file can cause the Integrity Client to behave unpredictably.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]merge`
- controls whether settings from the loaded file should be merged into existing preferences.
- *--rc=value*
- identifies the file containing settings for running the Integrity Client. The default is the `IntegrityClient.rc` file in your home directory.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:
- si setprefs, si viewprefs
- Miscellaneous:
- diagnostics, options, preferences

si locate

displays where a project, subproject, member, or revision is used

Synopsis

```
si locate [--depth=[current|build|all]] [--devpathscope=[this|others|all]] [--[no]failOnAmbiguousProject] [--distinct=
[project|devpath|registeredproject]] [--[no]exactmatch] [--[no]casesensitive] [--[no]limittoactivepaths] [--
listfields=field1[:width1],field2[:width2]...] [--mode=[distinct|list]] [--numberofresults=value]] [--memberbyname=value]] [--
subprojectbyname=value]] [--projectscope=[this|others|all]] [(-r value|--revision=value)] [--height=value] [--width=value] [-x=value]
[-y=value] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--
hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
member/subproject
```

Description

As projects grow in complexity, it can be difficult to determine the relationships that Integrity configuration management objects--projects, subprojects, and members--have with one another. When working with multiple development paths and shared subprojects or shared member archives, it becomes very important to understand what areas of a project might be affected by parallel development. Using the `si locate` command, you can locate where a specific Integrity object is used in your source code repository and refine and sort the search results.

For example, you might need to:

- Determine what active projects (a project currently under development) a member, subproject, or registered top-level project is shared into.
- Determine what development paths a selected member or subproject is a part of.
- Determine if a selected member revision is part of a checkpoint.
- Confirm all existing checkpoints for a selected revision before you delete it.
- Display the development path that a corresponding member revision belongs to.
- Display which master projects a selected subproject belongs to.

Integrity displays the search results in one of the following modes:

- Distinct Mode provides a high-level view of the information.
- List Mode provides a detailed view of the information.

Key Considerations:

- You require the *OpenProject* permission for the project you are running the `si locate` command from. When the Locate view displays, Integrity informs you if there are additional projects that you do not have permission to view.
- Archive locations are not displayed in the search results.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--depth=[current|build|all]`

  specifies the depth of the search. `--depth=current` searches current project configurations (normal and variant).`--depth=build` searches current project configurations and checkpoints.`--depth=all` searches current project configurations, checkpoints, and in between checkpoints.
- `--devpathscope=[this|others|all]`

  specifies which development paths to include in the search. `--devpathscope=this` searches the current development path. `--devpathscope=others` searches all development paths, except the current development path. `--devpathscope=all` searches all development paths.
- `--distinct=[project|devpath|registeredproject]`

  specifies the information you want displayed in Distinct Mode. This option must be used with `--mode=distinct`. `--distinct=project` lists only distinct project names.`--distinct=devpath` lists only development path names.`--distinct=registeredproject` lists only registered top-level project names.
- `--[no]exactmatch`

  when searching by name, specifies whether to match the name exactly. This option can only be used in conjunction with the `--memberbyname` or `--subprojectbyname` options.
- `--[no]casesensitive`

  when searching by name, specifies whether to match the case of the name. This option can only be used in conjunction with the `--memberbyname` or `--subprojectbyname` options and with case insensitive databases.
- `--[no]limittoactivepaths`

  specifies whether to search projects referenced by current registered top level projects on the server.
- `--listfields=field1[:width1],field2[:width2]...`

  allows you to select fields to be displayed when you use the `--mode=list` option, specified in the format *field1[:width1],field2[:width2]...*. Specifying the column *[: width]* (in pixels) for each field is optional (widths are only available in the GUI). In the CLI, fields are separated with a space. This option must be used with `--mode=list`.

  You can display one or more of the following fields:
  - `checkpoints`

    displays the checkpoints that the object belongs to.
  - `configPath`

    displays the configuration path for the object.
  - `dates`

    displays the date ranges in which the object exists.
  - `devpath`

    displays the development paths that the object belongs to.
  - `flatPath`

    displays the flat path for the object.
  - `name`

    displays the object's path and name.
  - `project`

    displays the projects that the object belongs to.
  - `revisions`

    displays the member's revisions.
  - `registeredproject`

    displays the top-level projects that the object belongs to.
- `--mode=[distinct|list]`

  specifies the level of detail to display in the search results. `--mode=distinct` displays only the projects and/or development paths that contain the object you are searching. This option must be used with `--distinct=[project|devpath|registeredproject]`. `--mode=list` displays all occurrences of the object in the projects and/or development paths that contain the object. If you do not specify a mode, `--mode=distinct` and `--distinct=devpath` are used.
- `--memberbyname=value`

  specifies the name of the member to search for. The name you enter can contain the `*` and `?` wildcard characters. For UNIX, the name you enter can

contain the \ escape character.

- `--subprojectbyname=`*`value`*

  specifies the name of the subproject to search for. The name you enter can contain the * and ? wildcard characters. For UNIX, the name you enter can contain the \ escape character.

- `--numberofresults=`*`value`*

  when searching by name, specifies the maximum number of results to return. This option can only be used in conjunction with the `--memberbyname` or `--subprojectbyname` options.

- `--projectscope=[this|others|all]`

  specifies which projects to search. `--projectscope=this` searches the specified project.`--projectscope=others` searches all projects and subprojects, except the current project.`--projectscope=all` searches all projects and subprojects.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si deleterevision, si viewproject, si viewprojecthistory, si viewhistory
- Miscellaneous: ACL, diagnostics, options, preferences

si lock

locks project members

Synopsis

si lock [--cpid=*ID*|--changePackageId=*ID*] [--lockType=*[exclusive|nonexclusive|auto]*] [--[no|confirm]downgradeOnLockConflict [--issueId=*ID*] [(-r *rev*|--revision=*rev*)] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-P *project*|--project=*project*)] [--[no]failOnAmbiguousProject] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--[no|confirm]revisionMismatchIsError] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *member...*

Description

`si lock` locks one or more project members. For example,

```
si lock --lockType=nonexclusive c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If no members are specified, `si lock` applies to all project members. By default, the member revision is locked and not necessarily the working revision, if in a Sandbox.

This is purely a project operation -- if performed in a Sandbox, no memory of what revision was locked is maintained in the Sandbox. However, you can create a work in progress indicator for the the associated working file in the Sandbox.

Depending on the locking policy configured by your administrator, you may hold a lock on only one revision of a particular member at a time. It is not an "error" to relock an already locked member.

If the target revision is already locked by a user other than yourself, then specifying the `--branch` option creates a branch revision and locks it. Otherwise, you cannot lock the revision at all.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--lockType=[exclusive|nonexclusive|auto]`

  specifies the type of lock obtained on checkout.

  `--lockType=exclusive` obtains an exclusive lock on the member. Exclusive locks enable only one member at a time to lock a specific revision.

  `--lockType=nonexclusive` obtains a non-exclusive lock on the member. Non-exclusive locks enable multiple users to check out the same revision for editing.

  `--lockType=auto` obtains a lock on the member based on the locks policy. For information on the locks policy, contact your administrator. If the locks policy does noe require a lock, but the --lock option is set, a non-exclusive lock is obtained.
- `--[no|confirm]downgradeOnLockConflict`

  controls whether to downgrade your lock to non-exclusive if another user has an exclusive lock on the revision.

  `--downgradeOnLockConflict` means always downgrade.

  `--nodowngradeOnLockConflict` means never downgrade.

  `--confirmdowngradeOnLockConflict` means always ask before downgrading.
- `--[no|confirm]revisionMismatchIsError`

  controls whether to display an error message if the working revision does not match the member revision. This only applies if you are performing the lock operation in a Sandbox.
- *`member...`*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si ci, si co, si edit, si rlog, si unlock
- Miscellaneous:

  ACL, diagnostics, options, preferences

si locks

displays a user's locks.

Synopsis

`si locks [--locker=value] [--fields=field1[:width1],field2[:width2]...] [--height=value] [--width=value] [-x=value] [-y=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)][--quiet][--settingsUI=[gui|default]] [--[no]persist] [--status=[none|gui|default]]`

Description

`si locks` displays all of a user's locks on the target server.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--locker=value`

  specifies the user whose locks are retrieved. If `:anyone` is specified, all locks in the system are displayed.
- `--fields=field1[:width1],field2[:width2]...`

  The fields available for printing can be one or more of the following:
  - `archive`

    displays the name of the locked archive.
  - `cpid`

    displays the change package ID number if it is a lock entry in a change package.
  - `devpath`

    displays the name of the development path where the lock on the revision was made from. This information is relevant when the locking policy is set to devpath, allowing a single user to have a single lock on an archive per development path.
  - `host`

    displays the hostname of the computer which the lock was performed on.
  - `member`

    displays the name of the locked member.
  - `project`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `revision`displays the locked revision number.
  - `locktype`

    displays the type of lock on the revision: exclusive or non-exclusive.
  - `sandbox`

    displays the name of the sandbox where the lock on the revision was made, and is relevant when viewing the information from the machine that locked it.
  - `timestamp`

    displays the time the archive was locked.
  - `user`

    displays the user holding the lock.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewprojecthistory, si viewsandbox
- Miscellaneous:

  ACL, diagnostics, options, preferences

si logging

modifies Integrity logging levels

Synopsis

`si logging [--category=value] [--debug] [--level=value] [--off] [--on] [--target=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si logging` modifies Integrity logging levels.

In addition to modifying the `logger.properties` file, you can use the `si logging` command to increase or decrease logging levels for several different categories (as outlined in the table below). You can run this command from a client or server machine, and use it to set client or server side logging. Any category included in the `logger.properties` file can be used. For example:

```
si logging --target=client --category=CACHE --level=5
```

changes *cache* logging on the client to level *5*.

Note

- The client and server do not need to be restarted after making changes.
- Logging changes last only until the Integrity Server or Integrity Client is restarted.
- You need the `DebugServer` permission in the `mks:si` ACL to run this command.
- `server.log` logs information about this command when it runs.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--category=value`

  where *value* specifies the category name. Possible values are:

  - `DEBUG`

    Logs debug messages.

    Note

    This command overrides the settings in `logger.properties`, but only until the Integrity Server is restarted. Additionally, this option does not explicitly log debug exceptions. To log exceptions, open `logger.properties`, uncomment `mksis.logger.exception.includeCategory.DEBUG`, and set the value to 10.

  - `SQL`

    Logs all SQL commands to `server.log`. For example, if you view an issue, the SELECT statement is logged. When editing an issue, the INSERT, UPDATE, and SELECT statements are logged. `--level=5` logs all SQL commands. `--level=10` adds additional information such as Rollback time.

    If SQL logging is enabled on the Integrity Server, you can set SQL logging levels for specific users. Setting the logging level can assist in isolating specific user commands and improving server performance. For example, if logging levels are high and server performance is impacted, reducing logging levels may improve performance.

    To set SQL logging levels for a user, type:

    `im/si --diag=sqllogging username logginglevel`

    where

    username is the user ID of the user whose commands you want to log.

    logginglevel is one of the following number or text values: `high (0), medium (5), low (10),` or `off (-1)`.

    For example, typing:

    `si diag --diag=sqllogging jriley high`

    logs all configuration management SQL commands for the user `jriley` to the category `SQL-jriley`.

  - `CACHE`

    Logs cache operation information. Levels 0–3 are not verbose. Levels 4–15 are verbose.

  - `SMTP`

    Logs communication between the Integrity Server and SMTP server.

  - `IM-NOTIFICATION`

    Logs e-mail notification.

  - `ACL`

    Logs permission checking to `server.log`. For example, the `add label` command logs the following:

    ```
    2007-08-16 13:45:17,140 INFO [mksis.IntegrityServer] ACL(5): Check user administrator for permission CreateProject against acl mks:si. Resolved ACL: mks:si Decision: GRANTED
    ```

  - `TRANSACTION`

    Logs all transactions performed by a specific user, for example, `si logging --category=TRANSACTION-jdoe --on` logs all transactions performed by jdoe. To log all cache transactions, type TRANSACTION-system

  - `SILIB`

    Similar to TRANSACTION, this option logs more information about a single command performed by a user, for example, `si logging --category=SILIB-jdoe --on`.

  - `SICIAGENT`

    Provides communications between functionality for workflows and documents, and configuration management, and communications between the Integrity Client and Integrity Server.

  - `REALM`

    Logs NT realm information.

  - `API`

    Logs Integrity API information.

  - `HLL`

    Logs Integrity HLL server information.

  - `LDAP`

    Logs LDAP security scheme information.

- `--debug`

  is equivalent to `--category=DEBUG`.

- `--level=value`

where `value` specifies the log level (-1 disables logging, 10 logs all messages)

- `--off`

  is equivalent to `--level=-1` (no reporting in this category).
- `--on`

  is equivalent to `--level=10` (logs all messages in this category).
- `--target=`*`value`*

  specifies the target the debugging is enabled for. Valid values for `--target` are `client`, `server`, and `proxy`. The default value for `--target` is `server`.

See Also

- Commands: si purgeauditlog, si viewauditlog, im viewauditlog
- Miscellaneous: options

si makewritable

makes Sandbox members writable

Synopsis

si makewritable [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *sandbox member...*

Description

`si makewritable` makes Sandbox members writable. For example,

```
si makewritable c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If no members are specified, `si makewritable` applies to all Sandbox members.

This command is useful when you want to edit a file locked by someone else, and you have no intention of checking the files back in.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *sandbox member...*
  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si diff, si edit, si lock, si resync, si rlog, si unlock
- Miscellaneous: ACL, diagnostics, options, preferences

si memberinfo

lists member information

Synopsis

```
si memberinfo [--[no]acl] [--[no]attributes] [--[no]changePackage] [--[no]labels] [--[no]rule] [(-P project|--project=project)] [(-S
sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--lockRecordFormat=value] [--devpath=path] [--projectRevision=rev] [--
hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--
[no]locate] member
```

Description

`si memberinfo` lists current information about a project member. This includes general information about the member, and specific information on the member revision. For example:

```
si memberinfo c:\Documentation\Man_Pages\xml_man\si_add.1.xml
```

displays

```
Member Name: c:\Documentation\Man_Pages\xml_man\si_add.1.xml
Sandbox Name: c:\Documentation\Man_Pages\project.pj
The member archive is shared
Member Revision: 1.7
 Created By: sueq on July 11, 2009 - 11:36 AM
 State: state_a
 Revision Description:
 Updated for Technical Review
 Labels:
 TechReview1
 Change Package:
 ID: 1:1
 Member Revision:
 In project C:\Aurora_Program\Documentation\project.pj, this revision
 is member revision on:
 development path:Service Pack 1
Attributes: none
This member does not have a revision rule.
```

Note

The member archive is shared displays only if the member shares another member's archive and you are using the database repository.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]acl`

  shows ACL information for the member.
- `--[no]attributes`

  controls whether to display member attributes. If no attributes exist, `Attributes:none` is displayed.
- `--[no]changePackage`

  controls whether to display change package information (applies only change packages are enabled).
- `--[no]labels`

  controls whether to display member labels.
- `--[no]rule`

  controls whether to display the member revision rule.
- `--[no]locate`

  controls whether to display the project and development path that the member is a member revision in.

  Note

  If a revision is member revision in a shared subproject, only the original (canonical) project path displays. Any projects where the subproject was added as shared do not display.

  Only projects that you have the `OpenProject` permission for display.
- `--lockRecordFormat=value`

  defines the format for displaying lock information for the member. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:
  - `{revision}`

    displays the revision that is locked.
  - `{locker}`

    displays the user who locked the revision.
  - `{locktype}`

    displays the type of lock on the revision (exclusive or non-exclusive).
  - `{locktimestamp}`

    displays the time when the revision was locked.
  - `{lockcpid}`

    displays the change package associated with the lock on the revision.
  - `{project}`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `{devpath}`

    displays the name of the development path where the lock on the revision was made from.
  - `{sandbox}`

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
  - `{hostname}`

    displays the hostname of the computer that locked the the revision.
  - `{hascpid}`

    displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
  - `{hassandbox}`

displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.

- `{hasdevpath}`

  displays 1 if the lock was made from a development path, 0 if it wasn't.

- `{member}`

  displays the name of the locked revision.

- *member*

  identifies one specific member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si archiveinfo, si mods, si revisioninfo, si rlog, si setmemberrule

- Miscellaneous:

  ACL, diagnostics, options, preferences

si merge

merges data from two revisions of a project member into a working file

Synopsis

```
si merge [--mergeType=[confirm|cancel|automatic|manual]] [--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]] [--
[no]failOnAmbiguousProject] [--outputFile=value] [-r rev] [--resolve] [--guiCharacterEncoding=value] [(-R|--[no|confirm]recurse)] [--
filter=filteroptions] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|-
-usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--
gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] sandbox member...
```

Description

`si merge` merges data from two revisions of a project member against a root revision. By default, the result overwrites the working file. This operates in a Sandbox. You cannot perform a merge on a project-only basis, since the result is left in your Sandbox.

The second `-r rev` specified is merged with the revision in the current working file. The first `-r rev` specified is the *root* revision. Both the working file revision and the other revision should have been derived from this *root* revision. For example, suppose development has branched from revision 1.3, with maintenance occurring on the 1.3.1 branch. Further suppose the tip of that branch is revision 1.3.1.3, the headrev member revision is 1.5, and that the revision in the Sandbox is 1.5.

```
1.5 (headrev)
1.4
 1.3.1.3 (maintenance branch tip)
 1.3.1.2
 1.3.1.1
1.3 (root)
```

The command `si merge -r 1.3 -r 1.3.1.3` merges the changes in 1.3.1.3 that have occurred since 1.3, into revision 1.5 in the working file in the Sandbox.

`si merge` is useful for combining separate changes into a checked out revision. Suppose *root* is the original, and both *headrev* and *branch tip* are modifications of *root*. Then, `si merge` combines both changes.

An overlap occurs if both branches of development (*headrev* and *branch tip*) have changes in a common segment of lines. Depending on the value used in `--onMergeConflict`, `si merge` displays how many overlaps occurred, and includes both alternatives in the result, delimiting them as follows:

```
<<<<<<< file1
lines in file1
=======
lines in file2
>>>>>>> file2
```

If there are overlaps, by default Integrity asks you how you want to resolve the conflict. If you specify `--mergeType=automatic` and `--onMergeConflict=highlight`, it is up to you to determine how to resolve the conflict by editing the result.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--mergeType=[confirm|cancel|automatic|manual]`

  specifies how you want to complete the merge.

  - `--mergeType=confirm` prompts you to confirm a merge type.
  - `--mergeType=cancel` cancels the selected merge.
  - `--mergeType=automatic` completes the merge process without launching the Visual Merge tool.

    Caution

    While Integrity and Visual Merge are capable of performing automatic merging, PTC cannot guarantee that the merged results are "correct". PTC recommends that you examine and test the merged results before checking them into the repository.

  - `--mergeType=manual` completes the merge operation through the Visual Merge tool or a third party merge tool, depending on your Difference and Merge Tool preferences. The merge tool launches, displaying the revisions you want to merge. For more information on the Visual Merge tool, refer to the *PTC Integrity User Guide*.
- `--onMergeConflict =[confirm|cancel|mark|launchtool|highlight|error]`

  specifies what to do when conflicts occur during the merge.

  - `--onMergeConflict=confirm` prompts you to confirm a merge conflict option.
  - `--onMergeConflict=cancel` cancels the merge.
  - `--onMergeConflict=mark` marks the working file in your Sandbox indicating that merging is required without completing all of the merge related tasks. This provides the time to investigate conflicts, edit, or difference blocks before finishing the merge.

    Tip

    To display the affected member revisions, use the `si print --filter=unresolvedmerges` command. After you resolve the merge conflicts, merge the revisions using the `si merge --resolve` command.

  - `--onMergeConflict=launchtool` launches the Visual Merge tool to resolve the conflicts, all non-conflicting blocks will already have been applied.
  - `--onMergeConflict=highlight` indicates conflicts in the working file with the following characters: "<<<<<<" and ">>>>>>". You are responsible for manually resolving conflicts in the working file.

    Note

    `--onMergeConflict=launchtool` does not require the `-g` or `--gui` options.

  - `--onMergeConflict=error` displays an error when a merge conflict is encountered.
- `--outputFile=value`

  identifies the location and name of a file that is to contain the result of merging data from two member revisions. If you do not specify a file, the working file is used.
- `-r rev`

  uses a specified revision. *value* can be a valid revision number or a label.

  Specify this option twice to identify the two revisions to be merged; the first instance is the "old" revision, the second is the "new". The changes needed to make the "old" into the "new" are then incorporated into the working file or into the `--outputFile`. If you specify the `-r rev` option only once, `si merge` uses the latest revision on the default branch as the other revision.
- `--resolve`

  merges a previously unresolved merge.
- `--guiCharacterEncoding=value`

  specifies the character encoding to use for the revision contents in the GUI, and can only be specified with the -g option. Integrity automatically decodes UTF-8 revision contents based on the presence of a byte order mark (BOM) in the file. You can set character encoding preferences for each view and command through your client preferences. The preference setting is used if the character set cannot be determined through the file's BOM, or if no character set is specified in the command line. By default, the character set in the preferences matches the default character set provided by the client's

operating system locale. The option does not apply to pure CLI output, or third party merge and/or differencing tools. Possible values are:

```
UTF-8
US-ASCII
windows-1252
ISO-8859 -1
ISO-8859 -15
IBM437
IBM850
IBM863
EUC-JP
Shift_JIS
x-euc-jp-linux
x-eucJP-Openx-windows-iso2022jp
IBM862
ISO-8859-8
ISO-8859-9
```

- *sandbox member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si diff, si mergebranch, si revisioninfo
- Miscellaneous: diagnostics, options, preferences

si mergebranch

merges changes made on a branch into a single revision

Synopsis

si mergebranch [--mergeType=*[confirm|cancel|automatic|manual]*] [--onMergeConflict=*[confirm|cancel|mark|launchtool|highlight|error]*] [--[no]failOnAmbiguousProject] [--branchRevision=*value*] [(--cpid=*ID*|--changePackageId=*ID*)] [--issueId=*ID*] [(--[no]lock)] [--targetRevision=*value*] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-S *sandbox*|--sandbox=*sandbox*)] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *sandbox member...*

Description

si mergebranch allows you to combine the development occurring on different branches by merging the branched revisions into a single revision.

Integrity recognizes cases where a previous merge has occurred, (whether by merging the branch, automatic merging, or using the Visual Merge or a third party merge tool), and merges only changes since the last merge operation. Thus, the first merge branch operation occurring on a given branch merges all changes throughout the branch. Any subsequent merge branch operations will merge only changes since the previous merge operation.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --mergeType=*[confirm|cancel|automatic|manual]*

  specifies how you want to complete the merge.

  --mergeType=*confirm* prompts you to confirm a merge type.

  --mergeType=*cancel* cancels the selected merge.

  --mergeType=*automatic* completes the merge process without launching the Visual Merge tool.

  Caution

  While Integrity and Visual Merge are capable of performing automatic merging, PTC cannot guarantee that the merged results are \u201ccorrect\u201d. PTC recommends that you examine and test the merged results before checking them into the repository.

  --mergeType=*manual* completes the merge operation through the Visual Merge tool or a third party merge tool, depending on your Difference and Merge Tool preferences. The merge tool launches, displaying the revisions you want to merge. For more information on the Visual Merge tool, refer to the *PTC Integrity User Guide*.
- --onMergeConflict=*[confirm|cancel|mark|launchtool|highlight|error]*

  specifies what to do when conflicts occur during the merge.

  --onMergeConflict=*confirm* prompts you to confirm a merge conflict option.

  --onMergeConflict=*cancel* cancels the merge.

  --onMergeConflict=*mark* marks the working file indicating that merging is required without completing all of the merge related tasks. This provides the time you may need to investigate conflicts, edit, or difference blocks before finishing the merge.

  --onMergeConflict=*launchtool* launches the Visual Merge tool to resolve the conflicts, all non-conflicting blocks will already have been applied.

  --onMergeConflict=*highlight* indicates conflicts in the file with the following characters: "<<<<<<" and ">>>>>>". You are responsible for manually resolving conflicts in the working file.

  Note

  --onMergeConflict=*launchtool* does not require the -g or --gui options.

  --onMergeConflict=*error* displays an error when a merge conflict is encountered.
- --branchRevision=*revision*

  specifies the branch revision that you want to merge into the head revision. If no revision is specified, the working revision is used.
- --[no]lock

  controls whether to lock the target revision, so that the results of the merge can be checked in. The lock type used is based on your locks policy. For information on your locks policy, contact your administrator.
- --targetRevision=*revision|:symbolic*

  specifies the symbolic name (i.e. member, working, branch, etc.), revision number, or label of the revision that you want to merge the branch revision into; typically the head revision. If no revision is specified, the member revision is used.
- *sandbox member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si ci, si diff, si edit, si lock, si merge, si resync, si rlog, si unlock
- Miscellaneous:

  ACL, diagnostics, options, preferences

si mergechilddevpath

merges development path into its parent revision

Synopsis

`si mergechilddevpath[--issueId=`*value*`][--sourcedevpath=`*value*`][--[no]failOnAmbiguousProject] [(-S `*sandbox*`|--sandbox=`*sandbox*`)][--hostname=`*server*`][--port=`*number*`][--password=`*password*`][--user=`*name*`][(-?|--usage)][(-F `*file*`|--selectionFile=`*file*`)][(-N|--no)][(-Y|--yes)][--[no]batch][--cwd=`*directory*`][--forceConfirm=`*[yes|no]*`][--quiet] [(-g|--gui)][--settingsUI=`*[gui|default]*`][--status=`*[none|gui|default]*`]

Description

`si mergechilddevpath` merges all changes from a branch into that branch's immediate parent revision.

Integrity provides you with a streamlined method of merging a development path into its parent development path. The parent development path may be mainline, or it may be a development path itself. The merge destination just needs to be the parent of the child being merged into it. For detailed information on merging a child development path, see the *PTC Integrity User Guide*.

Tip

To run the `si mergechilddevpath` command in batch mode from the CLI or API, set the desired merge preferences on `si resynccp` using the options for `--onMergeConflict` and `--mergeType`.

- `--sourcedevpath=`*value*

  specifies the development path to merge into the parent project revision. Only children of the previously selected project can be specified. This option can only be specified if the integration between configuration management, and workflow and document management is enabled.

- `--[no]failOnAmbiguousProject`

  This option does not currently function for the `si mergechilddevpath` command.

- `--sandbox=`*sandbox* `or -S` *sandbox*

  specifies the location of the target Sandbox. Locations that include spaces must be enclosed by quotes.

- `--issueID=`*value*

  specifies the item to use to store the propagation change package information.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si resynccp, si submitcp, si closecp
- Miscellaneous: ACL, diagnostics, options, preferences

si migrate

is used for backing up projects, members and their associated archives to the backup schema, and restoring them from that schema

Synopsis

`si migrate [--publish] [--resume] [--rollback] [--dumpToBackup=`*`projectList`*`] [--restoreFromBackup=`*`projectNames`*`] [--[no]confirm] [--verbose=`*`projectLocation`*`]`

Description

`si migrate` is used for backing up projects, members and their associated archives to the backup schema, and restoring them from that schema.

Caution

- Multiple migration-specific options may not be used together. For example, `--resume` and `--rollback` may not be used together.
- If there is a temporary error, such as "file in use" or "project in use", the migrator performs an automatic resume operation.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `----[no]confirm`

  displays a confirmation message before performing the command. The `--confirm` option is enabled by default.
- `--dumpToBackup=`*`projectlist`*

  `projectList` copies projects to a backup database schema. The value for projectList specifies a selection of projects. This option is only available in MKS Integrity 2009 or greater. For detailed procedures on copying projects to a backup database, see the *Integrity Server 2009 Administration Guide*.
- `--publish`

  commits newly restored projects and archives to the database repository and ends the migration session. The `--publish` option also generates output on the progress of the migration. Running a publish operation causes a resume operation to run first, allowing the migrator to capture any changes that were made when the repository was fully accessible.
- `--resume`

  continues an restore that was previously interrupted. The `--resume` option also re-migrates projects and archives that have changed since the migration began or since the last interruption occurred. This option can only be used after a previous restore was started and has stopped, before a rollback, or before a publish operation.
- `--rollback`

  deletes all projects and archives in the database repository that have not been published, as well as objects that refer to them.
- `--restoreFromBackup=`*`projectnames`*

  restores previously deleted projects. The value for projectNames specifies the names of projects to restore from the backup tables in the database. This option is only available in MKS Integrity 2009 or greater. For detailed procedures on restoring deleted projects, see the *Integrity Server 2009 Administration Guide*.
- `--verbose=`*`projectlocation`*

  more detailed information on the progress of the migration operation. Using the `--verbose` option increases the size of the `migration.log` file.

  Note

  Using the `--verbose` option can degrade performance during a migration. Use this option only when required.

- *`projectlocation...`*
- specifies the location of projects on the Integrity Server. By default, the migrator only migrates projects that are visible in the project registry. If your project registry contains invisible projects, you must locate the project within the file system, and use the `si addproject` command to add it to project registry.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createproject, si dropproject, si projectinfo, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si mods

lists changes made to a project since a checkpoint

Synopsis

si mods [--fields=*field1[:width1],field2[:width2]...*] [(-r *rev*)] [--[no]showAppliedCPList] [--[no]showAttrChanges] [--[no]showChangePackages] [--[no]failOnAmbiguousProject] [--[no]showMemberChanges] [--[no]showRevDescription] [--devpath=*value*] [--projectRevision=*value*] [(-R|--[no|confirm]recurse)] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [--height=*value*] [--width=*value*] [-x=*value*] [-y=*value*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*]

Description

si mods displays modifications to a project between checkpoints or between a checkpoint and the working project.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --fields=*field1[:width1],field2[:width2]...*

  allows you to select fields to be printed, specified in the format *field1[:width1],field2[:width2]...*. Specifying the column [:width] (in pixels) for each field is optional - widths are only available with the -g or --gui options. Under the CLI the fields are separated with a space.

  The fields available can be one or more of the following:

  - change

    displays details of changes made to project members or subprojects.
  - name

    displays the member or subproject name.
- -r *rev*

  specifies one or more project checkpoints to compare. rev can be a valid checkpoint number or label. To compare two specified revisions of the project, use the -r option twice. If you only use the -r option once, the working project (or last checkpoint on a build project's branch) is compared to the specified project checkpoint. If you do not use this option at all, the working project (or build project checkpoint) is compared to the last checkpoint (or last checkpoint on the build project's branch).

  The -r option when used with this command can also take the "asof:" identifier with a date, to return the project configuration as of a specific date. For example, -r asof:"April 28, 2014 3:33:45 AM GMT-05:00".

  It is possible to specify the branch with the identifier, for example -r asof:"April 28, 2014 3:33:45 AM GMT-05:00"@1.3.1

  It is possible to specify the development path with the identifer using the form asof:*date*:@:*devpath*, for example -r asof:"April 28, 2014 3:33:45 AM GMT-05:00"@:Release2

  The following are additional forms may be displayed for configuration paths, and are represented here for information purposes:

  asof:ts=*timestamp.number*

  asof:ts=*timestamp.number@branch*

  asof:ts=*timestamp.number@:devpath*
- --[no]showAppliedCPList

  controls whether to show change packages applied to a project through si applycp or si resynccp.

  For more information on applying change packages, see si applycp.
- --[no]showAttrChanges

  controls whether to show the member attribute changes.
- --[no]showMemberChanges

  controls whether to show member changes.
- --[no]showRevDescription

  controls whether to show revision descriptions of new revisions that have been added. To provide context to the list of revisions, the initial member revision for the specified checkpoint is also included. For example, if a project member changed from 1.10 to 1.12, this option displays the descriptions for revisions 1.10, 1.11, and 1.12, even though 1.10 is not new since the last checkpoint.
- --[no]showChangePackages

  controls whether to show the change packages that correspond to member and subproject operations. Change package information displayed is drawn from the modifications calculated from the checkpoint comparison.

  The By Change Package panel displays the following information:

  - ID displays the change package ID for change packages that contain entries with member or subproject changes between project checkpoints.
  - Summary displays the change package summary for change packages that contain entries with member or subproject changes between project checkpoints.

  Modifications that are not associated with a change package (unresolved) are displayed in the bottom frame of the By Change Package panel. Where possible, the member or project name is displayed with full path information. The following are reasons why modifications can appear in the bottom frame:

  - The change package entry is missing; either because one was not recorded for the member or subproject operation, or the entry was discarded.
  - The operation that made the modification cannot use a change package, for example, freezing or thawing a member.
  - The project state captures compared are from different development paths.

  Pending change package entries do not appear in the By Change Package panel (GUI).
- -R
- --[no|confirm]recursecontrols whether to recurse into subprojects that are determined to be in common between the two projects.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si appendrevdesc, si archiveinfo, si checkpoint, si ci, si co, si diff, si memberinfo, si merge, si projectinfo, si revisioninfo, si rlog, si sandboxinfo, si updatearchive, si updaterevision, si viewhistory
- Miscellaneous: ACL, diagnostics, options, preferences

si move

moves one or more project members between projects, or between directories in a single project

Synopsis

```
si move [(--cpid=ID|--changePackageId=ID)] [--[no|confirm]closeCP] [--[no]confirm] [--[no]defer] [-f] [--issueId=ID] [--
[no]moveWorkingFile] [--[no]failOnAmbiguousProject] [--[no|confirm]overwrite] [--[no|confirm]branchVariant] [--[no]createSubprojects]
[--targetDevpath=value] [--targetDir=value] [--targetProject=value] [--targetSandbox=value] [--filter=filteroptions] [(-P project|--
project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--
user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

As software designs and project structures change, it may become necessary to move one or more members between projects, variants of the same project, or directories in a project.

Moving a member performs a drop in the source location and an add in the target location, creating a new revision in the existing archive. The member's archive remains in its original location in the repository so that change packages, member histories, and project histories continue to work. If the move is performed with a change package, the member is added as a single *Move* entry in the change package. If you are moving multiple members, any common directory prefix shared by the members is automatically removed during the move.

Key Considerations

- Moving members between projects on different servers is not supported.
- You can perform deferred member moves only if both the source and target locations are sandboxes.
- `si move` does not work recursively on subprojects. To move one or more subprojects, use the `si movesubproject` command.
- Integrity detects whether any ACLs exist in the tree defined by the source project, the target project, and the common root between them (or the global ACL, if there is no common root). If any ACLs are found, Integrity warns you so you can manually make any required adjustments to the ACLs after the move is complete.

The following is an example of the command syntax:

```
si move --confirm --targetProject=c:/Aurora_Program/bin/project.pj c:/Aurora_Program/bin/Libra/code.c
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]closeCP`

  controls whether to close the associated change package.

  `--nocloseCP` means do not close the change package.

  `--confirmcloseCP` means ask before closing the change package.

  `--closeCP` always closes the change package.
- `--[no]confirm`

  controls whether to confirm the move before proceeding.
- `--[no]defer`

  controls whether to delay the move operation in the project until the deferred operation is submitted. The move operation in the sandbox still takes place immediately. This option is available only if the source and target are sandboxes.
- `-f`

  forces the move without confirmation.
- `--[no|confirm]overwrite`

  controls whether to overwrite the working file if it exists. This option is valid only if you are moving one or more members from a source sandbox to a target sandbox.
- `--[no]moveWorkingFile`

  controls whether to move the working file into the sandbox immediately. This option is valid only if you are moving one or more members from a source sandbox to a target sandbox.
- `--[no]createSubprojects`

  controls whether to create subprojects in existing directories that do not contain subprojects.
- `--targetDevpath=value`

  specifies the target development path (for variant projects). This is a label that was associated with a branch of the project by si createdevpath. Paths that include spaces must be enclosed by quotes. The following characters may not be used in a development path: \n, \r, \t, :, ['], #.

Note
This option cannot be used if you specify a target project using a keyword string for the `--targetProject` option.

- `--targetDir=value`

  specifies the subdirectory in the destination project or sandbox's directory that you want to move the member(s) to.
- `--targetSandbox=value`

  specifies the name of the target sandbox (can be used as a project redirector).
- `--targetProject=value`

  specifies the name of the target project. For information on how to specify a project, see the options reference page.
- `--[no|confirm]branchVariant`

  controls whether Integrity create a branch for the moved member in the target location.
- *member...*

  identifies a member to move; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si co, si rename, si movesubproject
- Miscellaneous: ACL, diagnostics, options, preferences

si movesandbox

moves a top level Sandbox to a new location

Synopsis

```
si movesandbox [(-S sandbox|--sandbox=sandbox)] [--[no]confirm] [--[no]overwrite] [--targetDir=value] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|actions|default]]
```

Description

You can move a top level Sandbox, and all of its members, subsandboxes, and associated metadata to another location. Moving a Sandbox performs a Sandbox retarget operation followed by a Sandbox move or rename operation. After a Sandbox move operation:

- Everything in the current directory and subdirectories is moved, including non members and work-in-progress files,
- The target Sandbox retains its configuration type (normal, variant, build), and
- All deferred change package entries will be updated with the new Sandbox name.

Before using `si movesandbox`, note the following:

- Moving a Subsandbox or multiple Sandboxes is not supported.
- When you move a Sandbox, the command creates the target directory if it does not already exist on disk.
- The operation will fail if you have locks on members in the Sandbox. Use the si locks command to find and address locked members, and then run the `si movesandbox` command again.
- When performing a Sandbox move operation, Integrity performs a Sandbox retarget operation before moving the files on disk. If the Sandbox move operation fails due to file system issues (such as directory permissions), you can correct the file system issue and move the Sandbox contents manually. You can also resychnronize the target Sandbox; however, resynchronizing can overwrite work-in-progress files.

The following is an example of the command syntax:

```
si movesandbox --confirm --sandbox=c:/Aurora_Program/CurrentModel/project.pj
--targetDir=c:/Aurora_Program/NewModel
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *-S sandbox*--`sandbox=`*sandbox*

  specifies the location of a Sandbox. This option is required. Locations that include spaces must be enclosed by quotes.

-
- `--[no]confirm`

  controls whether to confirm the move before proceeding.

- `--[no]overwrite`

  controls whether to overwrite the target directory if it exists and is not empty.

- `--targetDir=`*value*

  specifies the top level root Sandbox destination directory. This option specifies the Sandbox folder itself and does not specify the Sandbox container directory. You cannot specify a target directory that is already registered as a Sandbox or a Subsandbox.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si createsandbox, si configuresandbox, si dropsandbox, si move
- Miscellaneous: ACL, diagnostics, options, preferences
-

si movesubproject

moves a subproject between projects or between directories of a single project

Synopsis

```
si movesubproject [--[no]confirm] [--[no]createSubprojects] [-f] [--moveWorkingFiles=[none|members|all]] [--[no|confirm]overwrite] [--targetDevpath=value] [--targetDir=value] [--targetProject=value] [--targetSandbox=value] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [--[no]preserveCase] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--cpid=ID|--changePackageId=ID] [--[no|confirm]closeCP] [--issueId=value] subproject or subsandbox
```

Description

To meet the needs of changing project configurations, you can move one or more subprojects, and all of its members and sub-subprojects, between projects, and/or directories in a single project, or variants of the same project on the same Integrity Server.

When a subproject is moved, it behaves like a shared subproject. The subproject in the new location continues to be backed by the underlying subproject in the old location and the path and name of the subproject file in the repository remains the same. Any external references (ACL names, event triggers, policy statements) to the moved subproject continue to work because they are based on the subproject's original name; however, a subproject that is moved into a new project hierarchy continues to inherit ACLs from its original hierarchy and not from the new parent project. The moved subproject also retains its configuration type (normal, variant, build). If you are moving multiple subprojects, any common directory prefix shared by the subprojects is automatically removed during the move.

Before using `si movesubproject`, note the following:

- Moving subprojects between projects on different servers is not supported.
- The moved subproject inherits the project or directory ACLs from its original location. You cannot apply the ACLs from the new location to the subproject.
- The path and name of the subproject file in the repository is permanently reserved. If you attempt to create a new subproject using the moved subproject's path and name in the repository, you are prompted to add the existing subproject. If you answer no, the create subproject operation exits without providing you with the option of creating a subproject with a different path and name.
- Deferred subproject moves are not supported.
- You can move one or more subprojects across directories within a single project.
- Moved subprojects are not displayed as shared in a Sandbox or Project view unless the subproject was shared before the move.
- When you move one or more subprojects, you cannot co-locate subprojects in the same directory. If you want to co-locate an existing subproject with another subproject, use the si sharesubproject command on the subproject you want to move, followed by si drop in the original location.
- If you move a subproject that is associated with any Integrity issues, the issues no longer display as associated issues for the project. You must edit the Integrity issues to associate them with the subprojects in their new locations.
- You can change the case of the subproject path name when you move the subproject in a Sandbox, for example, `Test/project.pj to test/project.pj`

Note

- To correctly change the subproject path name, the subproject directory and any members in it must exist on the disk containing the Sandbox. This is useful for correcting typing errors in the path name.
  - You cannot change the case of the subproject path name with transactional change packages and/or Change Package Reviews enabled. If transactional change packages and/or Change Package Reviews are enabled, drop the subproject in a change package, manually change the case of the subproject path name, and then use a new change package to add the subproject as a shared subproject.
  - You cannot use the si applycp or si resynccp command to propagate a move subproject (path name case change) operation or drop subproject/add shared subproject in a single operation
  - Before performing additional operations using the new subproject location, resynchronize the former subproject in your Sandbox.

The following is an example of the command syntax:

```
si movesubproject --confirm --targetProject=c:/Alias_Program/Capricorn/project.pj c:/Aurora_Program/bin/Libra/project.pj
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  controls whether to confirm the move before proceeding.
- `--[no]createSubprojects`

  controls whether to create subprojects in existing directories that do not contain subprojects.
- `-f`

  force the move without confirmation.
- `--moveWorkingFiles=[none|members|all]`

  controls whether to move existing working files in the subproject. This option is valid only if you are moving one or more subprojects from a source Sandbox to a target Sandbox.
- `--[no|confirm]overwrite`

  controls whether to confirm before overwriting any existing working files that exist in the new location. This option is valid only if you are moving one or more subprojects from a source Sandbox to a target Sandbox.
- `--targetDevpath=value`

  specifies the target development path (for variant projects). This is a label that was associated with a branch of the project by si createdevpath. Paths that include spaces must be enclosed by quotes. The following characters may not be used in a development path: \n, \r, \t, :, ['], #.

Note

This option cannot be used if you specify a target project using a keyword string for the `--targetProject` option.

- `--targetDir=value`

  specifies the subdirectory in the destination project or Sandbox's directory that you want to move the subproject(s) to.
- `--targetProject=value`

  specifies the name of the target project. For information on how to specify a project, see the options reference page.
- `--targetSandbox=value`

  specifies the name of the target Sandbox (can be used as a project redirector).
- `--cpid=ID`
- `--changePackageId=ID`

  identifies a change package that is notified of this action, for example, `1452:1`.

Note

- This option can only be specified if change packages are enabled.

- If the integration is enabled, but it is not mandatory to specify a change package, or if no change package is applicable, you can specify `--changePackageId=:none`
- The next time you are prompted for a change package ID, the last used change package ID is displayed by default, if *:none* was specified or at least one change package was successfully updated from the last operation.

- `--[no|confirm]closeCP`

  closes the change package after command completion.
- `--[no]preserveCase`

  specifies that the destination directory uses the same case that exists on the file system. By default, `--preserveCase` is specified. To change the case of the subproject path name, type the new subdirectory name using the `--targetDir=value` option and specify `--[no]preserveCase`. In a Project, this option is not available and the subproject path name is changed to the exact name you type.
- `--issueId=`*`value`*

  specifies the issue ID that corresponds to the change package that records the changes.
- *subproject or subsandbox*

  identifies a subproject or sub Sandbox to move; use spaces to specify more than one subproject

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si move, si sharesubproject, si createsubproject, si configuresubproject, si addsubproject
- Miscellaneous: ACL, diagnostics, options, preferences

si opencp

reopens a change package

Synopsis

`si opencp [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`issue|issue:change package id...`

Description

`si opencp` reopens a change package. For example,

```
si opencp 3433:5
```

A change package is in the `Open` state when it is first created. This command is useful for reopening a change package after it has been submitted.

The open state is the only state where you can add new entries to a change package. As part of the review workflow, change packages in the following states can be reopened: `CommitFailed`, `Rejected`, and `Discarded`.

Note

- Only a Change Package Administrator can reopen a change package in the `Closed` state. Modifying closed change package contents can compromise the integrity of the repository.
- A change package cannot be reopened if it has been propagated to another development path. For more information on propagation change packages, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *issue...issue:change package id...*

  `issue` identifies a specific issue that contains all change packages that you want to open; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to open; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si cpissues , si createcp, si viewcps, si acceptcp, si rejectcp, si discardcp
- Miscellaneous: ACL, diagnostics, options, preferences

si primeproject

populates data on the proxy for a project

Synopsis

`si primeproject [-P|--project=value] [-S|--sandbox=value] [--devpath=value] [--projectRevision=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si primeproject` populates data on the proxy for a selected normal, variant, or build configuration management project.

When users are ready to connect through the proxy to the Integrity Server, there can be a delay while the proxy loads the required projects and members. When large volumes of information must be transferred over slow connections—such as for a large software development project—the delays can be significant for users until the proxy cache has received the information.

To avoid delays for users, you can prime the proxy with the required projects before users are active on the connection. To save further delays, the priming operation can be completed during non-work hours to avoid significant delays for users.

For the same reason that you manually pre-prime the proxy before users access it for the first time, you can use the `si primeproject` command to prime the proxy for an individual new project before users access that project through the proxy. For larger projects, priming the proxy in advance means that users are not delayed by waiting for that project's data to download from the host. You can also use the `si primeproject` command any time there is significant new data for an individual project on the host. When you run the `si primeproject` command, the proxy downloads bulk data for the project you have specified. Once you have primed the proxy for a specified project, performance is improved for users when subsequently creating the associated normal, variant, or build sandbox.

For detailed information on priming the proxy, see the *PTC Integrity Server Administration Guide*.

Options

This command takes the universal options available to all `si commands`, as well as some general options. See the options reference page for descriptions.

- `-P|--project=value`

  specifies the name of the target normal project.
- `-S|--sandbox=value`

  specifies the name of the sandbox used as a project redirector.
- `--devpath=value`

  specifies the name of the target variant project.
- `--projectRevision=value`

  specifies the revision number of the target build project.

See Also

- Commands: None
- Miscellaneous: options

si print

displays member information

Synopsis

`si print [--format=value] [--headerFormat=value] [--noFormat] [--noHeaderFormat] [--noTrailerFormat] [--lockRecordFormat=value] [--lockRecordDetailFormat=value] [--lockseparator=value] [--maxTrunkRevs=value] [--rfilter=filteroptions] [(-r rev|--revision=rev)] [--range=value] [--trailerFormat=value] [--fields=field1[:width1],field2[:width2]...] [--height=value] [--width=value] [-x value] [-yvalue] [--[no]failOnAmbiguousProject] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]persist] [--quiet] member...`

Description

`si print` displays member information on the standard output. For example,

```
si print c:/Documentation/Man_Pages/xml_man/si_add.1.xml
```

If you do not specify any members, `si print` displays all project members. This command is the same as si rlog; however, it has a different set of default options. In particular, `--noFormat` and `--noTrailerFormat` are specified, and `--headerFormat=`*value* produces one line per member.

Options

See the si rlog reference page for descriptions of all options applicable to `si print`. This command takes the universal options available to all `si` commands, as well as some general options - these are described on the options reference page.

Note

The `--headerFormat=`*value* and `--trailerFormat=`*value* options only accept global fields. See si rlog for the list of global fields. Revision specific fields, such as State, cannot be used with these options.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si diff, si memberinfo, si mods, si projectinfo, si report, si revisioninfo, si rlog, si sandboxinfo, si viewhistory
- Miscellaneous: ACL, diagnostics, options, preferences

si projectadd

adds members through a project

Synopsis

```
si projectadd [(-P value|--project=value)] [--archive=value] [--author=value] [--devpath=value] [--onExistingArchive=
[confirm|cancel|sharearchive|newarchive]] [(--binaryFormat|--textFormat)] [--[no|confirm]includeFormers] [--
include=file:pattern,dir:pattern...] [--exclude=file:pattern,dir:pattern...] [(--cpid=ID|--changePackageId=ID)] [--issueId=value] [--
[no|confirm]closeCP] [(-d value|--description=value)] [--descriptionFile=value] [(-l|--lock)] [(-r value|--revision=value)] [--
[no]retainSourceFile] [--sourceFile=value] [--[no]saveTimestamp] [--[no]unexpand] [--hostname=value] [--port=value] [--
password=value] [--user=value] [(-?|--usage)] [-Fvalue] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--forceConfirm=[yes|no]] [(-F file|--
selectionFile=file)] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] nonmember...
```

Description

`si projectadd` adds members to Integrity configuration management through a project.

Note

This command can be only used with the Integrity API. It is supported in the CLI only for the purpose of prototyping Integrity API implementations. It is used to add members from third party applications to Integrity. Use the `si add` command to add members through a Sandbox.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--author=value`

  specifies an author name.
- `--onExistingArchive=[confirm|cancel|sharearchive|newarchive]`

  controls whether to allow sharing of this member's history between projects, or to create a new archive if there is already an existing archive for the member.

  `--onExistingArchive=confirm` means always ask whether to share the archive, create a new archive, or cancel the operation.

  `--onExistingArchive=cancel` means cancel the operation.

  `--onExistingArchive=sharearchive` means share the archive.

  `--onExistingArchive=newarchive` means create a new archive.
- `--binaryFormat`
- `--textFormat`

  forces the storage of the member file to occur in binary format or text format.
- `--[no|confirm]closeCP`

  controls whether to close the associated change package.

  `--nocloseCP` means do not close the change package.

  `--confirmcloseCP` means ask before closing the change package.

  `--closeCP` always closes the change package.
- `-d desc`
- `--description=desc`

  specifies a description for the new archive. This option and the `--descriptionFile` option are mutually exclusive.

  Note

  Descriptions that include spaces must be enclosed by quotes.

- `--descriptionFile=file`

  specifies a file name, `file`, containing the description text for the new archive. This option and the `-d` or `--description` options are mutually exclusive.
- `--exclude=file:pattern,dir:pattern...`

  specifies a file that contains a glob pattern for excluding members.
- `--include=file:pattern,dir:pattern...`

  specifies a file that contains a glob pattern for including members.
- `-l`
- `--lock`

  keeps locks on members.
- `--[no]retainSourceFile`

  controls whether to keep the source file after the project has been added.
- `--sourceFile=value`

  identifies the source file being added.
- `--[no]saveTimestamp`

  controls whether to set the revision timestamp to the modification date and time of the working file rather than the current date and time.
- `--[no]unexpand`

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

```
$Author: Warner, Carrie (cwarner) $
$CompanyInfo$
$Date: 2015/11/30 15:01:45EST $
$Header: si_projectadd.dita 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner) Exp  $
$Id: si_projectadd.dita 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner) Exp  $
$Locker:  $
$Log: si_projectadd.dita  $
Revision 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner)
XML tagging fixes
Revision 1.1 2015/10/29 10:24:57EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$RCSfile: si_projectadd.dita $
$Revision: 1.2 $
```

```
$SandboxSetting  $
$Setting  $
$Source: si_projectadd.dita $
$State: Exp $
```

- *nonmember...*
  identifies a specific project member; use spaces to specify more than one project member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si projectci, si projectco
- Miscellaneous: ACL, diagnostics, options, preferences

si projectci

checks in members of a project through a project

Synopsis

```
si projectci [(-L value|--label=value)] [(-P value|--project=value)] [--author=value] [--devpath=value] [--[no]branch] [--[no|confirm]branchVariant] [--[no|confirm]branchUpdate] [(--cpid=ID|--changePackageId=ID)] [--[no|confirm]differentNames] [--issueId=value] [--[no|confirm]checkinUnchanged] [--[no|confirm]closeCP] [--[no|confirm]convertArchiveOnGovernor] [(-d value|--description=value)] [--descriptionFile=value] [(-l|--[no]lock)] [--[no|confirm]moveLabel] [(-r value|--revision=value)] [--[no]retainSourceFile] [--sourceFile=value] [--[no]saveTimestamp] [(-u|--unlock)] [--onExistingRevision=[resync|resyncbycp|confirm|cancel]] [--[no]unexpand] [--[no]update] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [-Fvalue] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--forceConfirm=[yes|no]] [(-F file|--selectionFile=file)] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si projectci` checks in and saves changes to project members through a project.

Check in is an operation that adds a new revision of a file to an archive. When a file is checked in to a revision other than the head revision or a branch tip revision, a new branch is created.

Note

- This command can be only used with the Integrity API. It is supported in the CLI only for the purpose of prototyping Integrity API implementaions. It is used to check in source information from third party applications to Integrity configuration management projects. Use the `si ci` command to check in through a Sandbox
- You cannot check in symbolic link file members using this command. Symbolic link files require a Sandbox context.

`Assigning Revision Numbers`

By default, when you check in a member, Integrity automatically assigns a unique revision number to the new revision. It does this by incrementing the current revision number by one. For example, if the previous revision is 1.3, the new revision is assigned number 1.4.

You can choose the revision number of the changes you are checking in, so long as your revision number:

- is greater than the last revision number (you cannot use previously "skipped" revision numbers)
- has no leading zeros (zeros as complete revision numbers are acceptable)
- starts a new branch based on an existing revision

If you check in a revision using an already existing revision number, Integrity attempts to add one to the revision number and check it in as that revision. If that revision already exists, Integrity then chooses the next available branch number and creates a new branch.

For example, if you are checking in a new revision to an archive where the head revision is 1.7, the following numbers are valid:

- 1.8 (greater than head revision)--if you check in a revision as 1.7, which already exists, Integrity assigns it 1.8
- 1.10 (greater than head revision)
- 1.72 (none of the numbers between 7 and 72 may be used afterwards)
- 2.0
- 1.7.1.1 (if it starts a new branch)
- 1.7.0.1 (leading zero as the branch number)

The following numbers are invalid:

- 1.3 even if there was no revision 1.3 previously
- 1.08 (leading 0 in last portion)
- 02.1 is considered the same as 2.1 (leading zero in branch number)

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]differentNames`

  controls whether to allow different source file and member names.
- *-L value*
- `--label=`*value*

  identifies a label that is applied to the new revision. Note the following about using labels:
  - Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
  - Labels cannot have the same format as a valid revision number.
  - Labels that include spaces must be enclosed by quotes.
  - PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
- `--author=`*value* specifies an author name.
- `--[no]branch` controls whether to force the creation of a branch revision

  ---

  Note

  If required, a branch revision is created even if this option is set to "no", for example, if you are checking in a member to a revision other than the head revision.

- A *branch* is a revision path that diverges from the main line of development (also known as the *trunk*) in a member or project history. A branch is typically created by checking in a file to a revision other than the head revision. The most recent revision of a branch is called the *tip revision*.

  Integrity usually places new revisions at the top of the trunk, assigning them two-part revision numbers, such as 1.15. There are times, however, when you do not want your work to be checked into the trunk. You may be pursuing a line of development that will not be included in the finished product, for instance, or you may be doing post-release maintenance while development for the next release continues on the trunk.

  Divergent lines of development in the same archive are managed through the use of branches. A branch is an independent revision line that uses an existing revision as its starting point. Members of a branch revision are identified by their revision numbers. Whereas revisions on the trunk are characterized by two-part revision numbers (for example, 1.2 or 3.5), branch revision numbers are prefixed with the number of the revision they start from. For example, if a branch revision is started from revision number 1.2, the members of that branch are numbered

  ```
  1.2.1.1
   1.2.1.2
   1.2.1.3
  ```

  and so on. The first two digits of the number identify the revision where the branch diverges from the trunk, and the last two represent a position on the branch.
- `--[no|confirm]branchUpdate`

  controls whether to update the member revision, even if it is on a branch. This option stops the member revision from accidentally moving onto a branch if a branch was created during check out. This option only applies if `--update` is specified, `--revision` was not specified, and the member revision is on a different branch from the working revision.

  `--nobranchUpdate` means do not update the member revision.

--confirmbranchUpdate means ask before updating the member revision.

--branchUpdate always updates the member revision.

- --[no|confirm]checkinUnchanged
- controls whether to force the checkin so that the new revision is checked in even if it is not different from the preceding one.--nocheckinUnchanged means do not force the checkin.--confirmcheckinUnchanged means ask before forcing the checkin.--checkinUnchanged always forces the checkin.
- --[no|confirm]closeCP

controls whether to close the associated change package.

--nocloseCP means do not close the change package.

--confirmcloseCP means ask before closing the change package.

--closeCP always closes the change package.

- --[no|confirm]convertArchiveOnGovernor
- controls whether to automatically convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size. If you are using the database repository, your administrator may set the policy that determines the maximum size of text working files that can be stored in a store-by-delta archive. The policy is useful in preventing out of memory issues on the Integrity Server when Integrity attempts to difference large text files. If the file exceeds the maximum size during a check in, you can cancel the operation or convert the archive to store-by-reference.

--noconvertArchiveOnGovernor means do not convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size. This cancels the check in operation.

--confirmconvertArchiveOnGovernor means ask before converting a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size.

--convertArchiveOnGovernor always convert a text store-by-delta archive to store-by-reference if the working file exceeds the maximum size.

- -d*desc*
- --description=*desc*

specifies a description for the new revision. This option and the --descriptionFile option are mutually exclusive.

Note

Descriptions that include spaces must be enclosed by quotes.

- --descriptionFile=*file*

specifies a file name, file, containing the description text for the new revision. This option and the -d or --description options are mutually exclusive.

- -l
- --[no]lock

controls whether to keep locks on members.

- --[no|confirm]moveLabel

controls whether the application should move a label from one revision to another.

--nomoveLabel disables the moving of a label.

--confirmmoveLabel displays a confirmation message.

--moveLabel moves the label.

- --[no]retainSourceFile

controls whether to keep the source file after checkin.

- --sourceFile=*value*

identifies the source file being checked in.

- --[no]saveTimestamp

controls whether to set the revision timestamp to the modification date and time of the working file rather than the current date and time.

- -u
- --unlock

unlocks the newly checked-in revision. This is equivalent to specifying --nolock.

- --[no]unexpand

controls whether to unexpand or ignore keywords in the member file prior to checking it into the archive. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the PTC Integrity User Guide. Possible keywords are:

```
$Author: Warner, Carrie (cwarner) $
$CompanyInfo$
$Date: 2015/11/30 15:01:45EST $
$Header: si_projectci.dita 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner) Exp  $
$Id: si_projectci.dita 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner) Exp  $
$Locker:  $
$Log: si_projectci.dita  $
Revision 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner)
XML tagging fixes
Revision 1.1 2015/10/29 10:24:57EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectLabel:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$RCSfile: si_projectci.dita $
$Revision: 1.2 $
$SandboxSetting  $
$Setting  $
$Source: si_projectci.dita $
$State: Exp $
```

- --[no]update

controls whether to set the checked in revision as the project's member revision.

- --onExistingRevision=[resync|resyncbycp|confirm|cancel]

controls what happens when the revision being checked in is not the member revision in the development path.

--onExistingRevision=resync means resynchronize the member revision into the working file and move the lock (if any) to the member revision. The check in operation is not completed. You should perform additional testing on the merged file before checking it in.

--onExistingRevision=resyncbycp means resynchronize the member revision into the working file by change package, and move the lock (if any) to the member revision. The check in operation is not completed. You should perform additional testing on the merged file before checking it in.

--onExistingRevision=confirm means ask for confirmation of the action to be taken.

--onExistingRevision=cancel means cancel the operation.

- `--[no|confirm]branchVariant`

  controls whether Integrity creates a branch off the revision you are checking in, if you are working in variant Sandbox and this is the first time the member is checked in. This is useful for keeping changes in the variant separate from changes made in the mainline project.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si projectco, si projectadd
- Miscellaneous: ACL, diagnostics, options, preferences

si projectco

checks out members of a project into working files

Synopsis

```
si projectco [--lockType=[exclusive|nonexclusive|auto]] [--lineTerminator=[lf|cr|crlf|native]] [(-P value|--project=value)] [--
devpath=value] [(--cpid =ID|--changePackageId=ID)] [--issueId=value] [(-l|--[no|auto]lock)] [--[no|confirm]lockOnFreeze] [--
[no|confirm]downgradeOnLockConflict] [--targetFile=value] [(-r value|--revision=value)] [(-u|--unlock)] [--[no]update] [--
[no|un]expand] [--[no|confirm]overwriteExisting] [--[no]restoreTimestamp] [--hostname=value] [--port=value] [--password=value] [--
user=value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [(-F file|--selectionFile=file)] [--quiet] [--status=[none|gui|default]] member...
```

Description

`si projectco` checks out project members of a project into working files.

Check out is an operation that extracts the contents of a revision in an archive and copies it to a working file. You can check out any revision by specifying either its revision number or label. By default, the member revision is used. If an existing revision other than the head revision is specified, a branch from that revision is created.

Note

- This command can be only used with the Integrity API. It is supported in the CLI only for the purpose of prototyping Integrity API implementations. It is used to check out Integrity configuration management project members for use in a third party application. To check out project members in a Sandbox, use the `si co` command.
- If you check out a symbolic link file member, it is stored on the filesystem as a properties file containing a description of the link, rather than as the symbolic link file.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--lineTerminator=[lf|cr|crlf|native]`

  specifies the line terminator used for the working file. Different operating systems use different characters to indicate the end of a line.

  `--lf` for UNIX, Linux or other Posix systems.

  `--cr` for Mac OS systems.

  `--crlf` for Windows systems.

  `--native` for the line terminator that is native to the client operating system (for example, crlf on Windows, lf on UNIX)
- `--[no]branch`

  controls whether to force the creation of a branch revision. Specifying `--branch` always creates a branch.

  For details on branch revisions, see the si ci reference page.
- `--[no|confirm]branchVariant`

  controls whether Integrity should create a branch off of the revision you are checking out if this is the first time you have attempted to check it out in a particular variant. Specifying `--confirmbranchVariant` causes a prompt to be displayed so you can confirm the branching.
- `--targetFile=value`

  identifies where to check the member out to
- `-l--[no|auto]lock`
- controls whether to create locks on members.

  `--lock` means always lock the member.

  `--nolock` means never lock the member.

  `--autolock` means lock the member based on the locks policy. For information on the locks policy, contact your administrator.
- `--lockType=[exclusive|nonexclusive|auto]`

  specifies the type of lock obtained on checkout.

  `--lockType=exclusive` obtains an exclusive lock on the member. Exclusive locks enable only one member at a time to lock a specific revision.

  `--lockType=nonexclusive` obtains a non-exclusive lock on the member. Non-exclusive locks enable multiple users to check out the same revision for editing.

  `--lockType=auto` obtains a lock on the member based on the locks policy. For information on the locks policy, contact your administrator. If the locks policy does noe require a lock, but the --lock option is set, a non-exclusive lock is obtained.
- `--[no|confirm]lockOnFreeze`

  controls whether to lock the specified member even if it is frozen.

  `--lockOnFreeze` means always do it.

  `--nolockOnFreeze` means never do it.

  `--confirmlockOnFreeze` means always ask before doing it.
- `--[no|confirm]downgradeOnLockConflict`

  controls whether to downgrade your lock to non-exclusive if another user has an exclusive lock on the revision.

  `--downgradeOnLockConflict` eans always do it.

  `--nodowngradeOnLockConflict` eans never do it.

  `--confirmdowngradeOnLockConflict` means always ask before doing it.
- `-u`
- `--unlock`

  keeps the member unlocked, and is equivalent to `--nolock`.
- `--[no]update`

  controls whether to update the project's member revision to the checked out revision.
- `--[no|un]expand`

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

  ```
  $Author: Warner, Carrie (cwarner) $
  $CompanyInfo$
  $Date: 2015/11/30 15:01:47EST $
  $Header: si_projectco.dita 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner) Exp  $
  $Id: si_projectco.dita 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner) Exp  $
  $Locker:  $
  $Log: si_projectco.dita  $
  Revision 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner)
  XML tagging fixes
  ```

```
Revision 1.1 2015/10/29 10:24:57EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectLabel:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$RCSfile: si_projectco.dita $
$Revision: 1.2 $
$SandboxSetting  $
$Setting  $
$Source: si_projectco.dita $
$State: Exp $
```

- `--[no|confirm]overwriteExisting`

  controls whether to overwrite an existing target file.

  `--overwriteExisting` eans always do it

  `--nooverwriteExisting` eans never do it

  `--confirmoverwriteExisting` means always ask before doing it. When asked to confirm overwriting the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision that is being checked out.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no]restoreTimestamp`

  controls whether to restore the timestamp of the working file to the timestamp of the revision in the member history. If this is not specified, the timestamp is set to the current date and time.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si projectci, si projectadd
- Miscellaneous: ACL, diagnostics, options, preferences

si projectcpdiff

displays a list of the operations that occurred on a project between two checkpoints tracked in change packages.

Synopsis

```
si projectcpdiff [(-R|--[no|confirm]recurse)] [--fields=field1[:width1],field2[:width2]...] [--filter=value] [--format=value] [--height=value] [(-r rev)] [--width=value] [-x value] [-y value] [(-Pproject|--project=project)] [--projectRevision=rev] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--[no]failOnAmbiguousProject] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--hostname=server] [--password=password] [--[no]persist] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--user=name]
```

Description

`si projectcpdiff` displays a list of the operations that occurred on a project between two checkpoints (project revisions) or timestamps, if those operations were tracked in change packages. The displayed information can be used to inform decisions on changes that need to be propagated to a development path, to understand the work that was done on a project, or as part of analyzing a problem that occurred in a recent project revision (for example, you can identify how a member was modified that caused a problem).

The following is an example of using the command to return the entries between the checkpoints 1.2 and 1.5 for project and recursively for all the subprojects that it contains:

```
si projectcpdiff -P /specification/project.pj --projectRevision=1.2 -R -r 1.2 -r 1.5
```

The following are key considerations when using this command:

- This command only returns changes that were recorded in change packages. Operations that are not recorded in change packages, such as a restore project (or commands that used the `bypass` value for `--changePackageId`), are not represented in the command output.
- The project context (-P/--project and/or --projectRevision) is used to gain access to the project archive, but only the specified revisions (-r) are used by the command to determine the comparison.
- Modifications to member attributes are not included in the command output.
- If subprojects were added using a change package, only that operation is included in the command output. The members that were added with the subproject are not included in the command output because they were not recorded in the change package. Similarly, reconfigured subprojects appear in the command output, but modifications to the members of the subproject do not appear in the command output.
- The project comparison must be for the same line of development (the project mainline or a single development path). However, the originating revision that was branched to create the development path can also be specified the starting revision; for example, revision 1.2 can be compared with 1.2.1.1.
- The command output only displays the resulting member revision for each change package entry, not the original member revision.
- When propagations cause an update to a revision number, the intervening revisions are not included in the command output because the operation that created them is not in the change package for that time frame.
- The command accepts two revisions, one revision or no revisions. If only one revision is specified, then the supplied revision is the start revision and the end revision is the current project configuration on the development path (or mainline) to which the revision belongs. If no revisions are specified, then the start revision is the most recent checkpoint revision on the specified project configuration, and the end revision is the currently specified project configuration.
- The command ignores build subprojects since they are static and do not change over time. However, in cases where the configuration of a subproject changes over time, the subproject and its contents may be visible in the view. The command focuses only on the existing project structure at both the start and the end points specified for the command. Build subprojects or subprojects that did not exist at the start and end points are not visible in the view. Existing non-build subprojects at the start and end points are visible in the view. For example, an existing build subproject that is later converted to a variant subproject would appear in the view. A subproject created after the start period and later converted to a build subproject before the end date would not appear in the view.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format field1[:width1],field2[:width2]....

  Specifying the column [:width] (in pixels) for each field is optional. Fields are separated with a space.

  The fields available for printing can be one or more of the following:

  - `bytesadded`

    displays the number of bytes added by the operation. For text files, this field displays 0.
  - `bytesdeleted`

    displays the number of bytes deleted by the operation. For text files, this field displays 0.
  - `configpath`

    displays the configuration path of the change package entry.
  - `id`

    displays the change package ID.
  - `istext`

    indicates if the change package entry has a text archive; `true` or `false`. If `false`, the entry is a binary archive.
  - `linesadded`

    displays the number of lines added by the operation. For binary files, this field displays 0.
  - `linesdeleted`

    displays the number of lines deleted by the operation. For binary files, this field displays 0.
  - `location`

    displays the archive location for members or the location of the parent project for subprojects (the canonical path).
  - `member`

    displays the name of the member or subproject affected by the operation.
  - `membertype`

    displays the type of project element affected by the operation (member or subproject).
  - `project`

    displays the name and path of the project where the operation was performed. If the operation occurred in a shared subproject, the subproject name and path are displayed. If the operation involved two different projects (for example, moving a member from one project to another), the command returns two entries, with the first entry the originating project and the second entry the destination project.
  - `revision`

    displays the member revision number.
  - `server`

    displays the server the change package resides on.
  - `summary`

    displays the change package summary.
  - `timestamp`

    displays the timestamp of the selected change package.
  - `type`

    displays how the member was added to the change package.
  - `user`

displays the ID of the user who added the member to the change package.

- `variant`

  displays the names of variant projects associated with the member.

- `--filter=`*`value`*

  displays the change packages according to one of the following filters:

  - `user:name`

    displays change packages created by a specified username.

  - `type[:add|:addfromarchive|:drop|:import|:exclusivelock| :nonexclusivelock|:renamefrom|:renameto| :movememberfrom|:movememberto|:update| :updatearchive|:updaterevision| :createsubproject|:addsubproject| :addsharedsubproject|:configuresubprojectfrom| :configuresubprojectto|:movesubprojectfrom| :movesubprojectto|:dropsubproject]`

    displays change packages based on their entry type.

  - `member:<expression>`

    specifies a text string to filter change packages by member name.

- `--format=`*`value`*

  defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  This option and the `--fields` option are mutually exclusive.

  `--format` options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). The `--fields` option automatically adds a newline on the end, but you must supply the newline for formats with a `\n`, for example:

  ```
  si projectcpdiff --format="{project},{member},{revision}\n"
  ```

- `-r` *`rev`*

  specifies one or more project checkpoints to compare. *`rev`* can be a valid checkpoint number or label. To compare two specified revisions of the project, use the `-r` option twice. If you only use the `-r` option once, the working project (or last checkpoint on a build project's branch) is compared to the specified project checkpoint. If you do not use this option at all, the working project (or build project checkpoint) is compared to the last checkpoint (or last checkpoint on the build project's branch).

  The `-r` option when used with this command can also take the "`asof:`" identifier with a date, to return the project configuration as of a specific date. For example, `-r asof:"April 28, 2014 3:33:45 AM GMT-05:00"`.

  It is possible to specify the branch with the identifier, for example `-r asof:"April 28, 2014 3:33:45 AM GMT-05:00"@1.3.1`

  It is possible to specify the development path with the identifer using the form `asof:`*`date`*`:@:`*`devpath`*, for example `-r asof:"April 28, 2014 3:33:45 AM GMT-05:00"@:Release2`

  The following are additional forms may be displayed for configuration paths, and are represented here for information purposes:

  `asof:ts=`*`timestamp.number`* `asof:ts=`*`timestamp.number@branch`* `asof:ts=`*`timestamp.number@:devpath`*

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si viewcp, si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si projectinfo

lists project information

Synopsis

si projectinfo [--[no]acl] [--[no]attributes] [--[no]devpaths] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--[no]showCheckpointDescription] [--[no]associatedIssues] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*]

Description

si projectinfo displays current project information, for example:

```
si projectinfo -P /master_projects/SourceCode/project.pj --devpath Milestone1
```

displays

```
Variant Project Name: /master_projects/SourceCode/project.pj
Repository Location: /master_projects/SourceCode/project.pj
Server: intra-wif:7001
Development Path Information:
    Development Path: Milestone1
    Extendable Development Path
    On Live Configuration: Retain Live Configuration
    On Existing Development Path: Share Development Path
Last Checkpoint: 1.1
Configuration Path:#/master_projects#SourceCode#d=Milestone1
Restricted: false
Last Checkpoint Date: May 1, 2014 - 4:14 PM
Members: 83
Subprojects: 0
Description: Documentation Set
Checkpoint Description:
 Service Pack 1
Attributes:
 rd=/rd
Development Paths:
 Variant Project for calculator (1.1)
Associated Issues:
 78484: Feature Request - Add calculator function
 Labels:
 InceptionPhase: 2014-05-02 - The Feature Request is in the inception phase.
```

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no]acl

  controls whether to display ACL information for the project.
- --[no]attributes

  controls whether to display project attributes.
- --[no]devpaths

  controls whether to display project development paths.
- --[no]showCheckpointDescription

  controls whether to display the checkpoint description.
- --[no]associatedIssues

  controls whether to display information for any Integrity items that are associated with the project, including labels, label dates, and label summaries. Only item types that you have permission to view are displayed.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si viewproject, si viewprojecthistory
- Miscellaneous:ACL, diagnostics, options, preferences

si projectlocks

displays locks by all users on all members of a sandbox or project

Synopsis

```
si projectlocks [--fields=field1[:width1],field2[:width2]...] [--[no]failOnAmbiguousProject] [(-P project|--project=project] [(-S
sandbox|--sandbox=sandbox)] [--devpath=path] [--[no]includeSubprojects] [--hostname=server] [--port=number] [--password=password] [--
user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [--[no]persist] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [[--quiet]
```

Description

`siprojectlocks` displays a tab-separated list of locks for the project information specified.

Assume that the following is entered:

```
si projectlocks --project=TestProject/project.pj --includeSubproject
```

Example output follows:

```
/TestProject/Subproject/projet.pj servicepack.policy  mainline Non Exclusive Lock 1.2 6-Apr-2015 3:36:08 PM
/TestProject/Subproject/projet.pj logger.properties  mainline Non Exclusive Lock 1.1 31-Mar-2015 9:54:41 AM
/TestProject/Subproject/projet.pj cmSchema.properties  mainline Non Exclusive Lock 1.2 31-Mar-2015 9:54:41 AM
/TestProject/Subproject/projet.pj im.properties   mainline Non Exclusive Lock 1.2 6-Apr-2015 2:49:01 PM
```

Note
By default, `si projectlocks` displays the member names of locked members, not the working file names.

Options

This command takes some of the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  The fields available for printing can be one or more of the following:

  - `archive`

    displays the name of the locked archive.
  - `cpid`

    displays the change package ID number if it is a lock entry in a change package.
  - `devpath`

    displays the name of the development path where the lock on the revision was made from. This information is relevant when the locking policy is set to devpath, allowing a single user to have a single lock on an archive per development path.
  - `host`

    displays the hostname of the computer which the lock was performed on.
  - `member`

    displays the name of the locked member.
  - `project`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `revision`

    displays the locked revision number.
  - `locktype`

    displays the type of lock on the revision: exclusive or non-exclusive.
  - `sandbox`

    displays the name of the sandbox where the lock on the revision was made, and is relevant when viewing the information from the machine that locked it.
  - `timestamp`

    displays the time the archive was locked.
  - `user`

    displays the user holding the lock.
- `--[no]includeSuprojects`

  controls whether to show locks for subprojects to which you have permissions. `--noincludeSubprojects` shows only locks for the proejct, while `--includeSubprojects` shows locks for the project and all subprojects to which you have permissions.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si locks, si viewlocks, si projects, si viewsandbox
- Miscellaneous:

  ACL, diagnostics, options, preferences

si projects

lists projects on the Integrity Server

Synopsis

`si projects [--[no]displaySubs] [--height=value] [--manage] [--width=value] [-x value] [-y value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si projects` displays a list of projects registered to the currently connected Integrity Server. A registered project is currently visible on the Integrity Server. A project dropped with the `si dropproject` command still exists on the Integrity Server, but does not display in the list of registered projects.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]displaySubs`

  controls whether to display subprojects. By default, only top level registered projects are displayed.
- `--manage`

  enables the project manager view for those using the `-g` or `--gui` options.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createproject, si dropproject, si projectinfo, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si promote

promotes the state of a member

Synopsis

`si promote [(-r rev|--revision=rev)] [(-s state|--state=state)] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...`

Description

`si promote` promotes a project member to a higher state of development. A state is a textual description, defined by the administrator, used to classify the condition of a revision in a history. If no state is assigned to a revision at check-in, a default value of `Exp` (for Experimental) is used. See the *PTC Integrity Server Administration Guide* for details on setting up states.

At particular milestones, project members are ready to move to the next state of their development cycle (for example, from Development to Testing). This is done through states and their promotion.

Note

Promoting members is for historical purposes only. To more effectively control project workflow, PTC recommends using the Integrity integration.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `-s` *state*
- `--state=`*state*

  specifies the target state to be set for the promoted member. If no state is specified, the member is promoted to the next state.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si demote, si demoteproject, si promoteproject
- Miscellaneous:ACL, diagnostics, options, preferences

si promoteproject

promotes the state of a project

Synopsis

`si promoteproject [--[no|confirm]force] [(-s state|--state=state)] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si promoteproject` promotes a project to a higher state of development. A state is a textual description, defined by the administrator, used to classify the condition of a revision in a history. If no state is assigned to a revision at check-in, a default value of `Exp` (for Experimental) is used. See the *PTC Integrity Server Administration Guide* for details on setting up states.

Just as you can with a project member, you can promote the project itself (change its state, in other words). This can assist in management of an entire project through a development cycle, especially if the promotion functions are restricted to key project personnel. Normally, you promote a project (set its state) when you si checkpoint it, but you can do so at any time. Only the project is promoted, not the members of the project.

Note

Promoting projects is for historical purposes only. To more effectively control project workflow, PTC recommends using the Integrity integration.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]force`

  controls whether to force promotion of a project even if members have been changed.
- `-s state`
- `--state=state`

  specifies the target state to be set for the promoted project. If no state is specified, the project is promoted to the next state.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si demote, si demoteproject, si promote
- Miscellaneous:ACL, diagnostics, options, preferences

si purgeauditlog

deletes and archives specified audit log records on the Integrity Server

Synopsis

```
si purgeauditlog [--before=value] [--maxFileSize=value] [--[no]backup] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-
Ffile|--selectionFile=file)] [-g|--gui][--user=name] [--hostname=server] [(-N|--no)] [--password=password] [--port=number] [--quiet]
[--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]
```

Description

`si purgeauditlog` allows you to delete and archive specified records from the Integrity Server audit log. You can choose to delete or archive records older than a specified date, thereby controlling the size of the audit log table in the database. Each purging or archiving operation is also recorded in the audit log.

By default, the purged records are automatically archived in a backup directory on the Integrity Server. The archive file is a comma-separated values (CSV) file where each field is separated by a comma.

For example, the following command purges all audit log records prior to 10:00AM on January 20, 2007 and creates a CSV file archive of the records in the default directory:

```
si purgeauditlog --before="01/20/2007 10:00:00 AM"
```

The following command purges all audit records prior to 10:00AM on January 20, 2007 without creating a CSV file archive:

```
si purgeauditlog --before="01/20/2007 10:00:00 AM" --nobackup
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--before=`*`value`*

  where `value` specifies a single date in the format `MM/dd/yyyy h:mm:ss [AM|PM]`. All audit records created prior to the specified date are permanently deleted or archived from the database, according to the options you specify. The date value cannot be used to filter for records newer than a given date. By default, purged records are archived to the `<installdir>\data\audit\backup` directory (where `<installdir>` is the path to the directory where you installed the Integrity Server). The file is assigned the name `auditlogbackup_*.csv`.

- `--maxFileSize=`*`value`*

  specifies the maximum file size (in megabytes) allowed for the created archive audit log. By default, the maximum file size is set at 50MB. You can specify file sizes up to the maximum available on your file system.

- `--[no]backup`

  allows you to delete audit log records without first archiving those records. If you specify `--nobackup`, you are asked to confirm the deletion of the target records before the operation is completed.

See Also

- Commands: im purgeauditlog, im viewauditlog, si viewauditlog
- Miscellaneous: options

si putdbfile

puts a configuration management file into the database

Synopsis

`si putdbfile [--encoding=value] [--input=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-F file|--selectionFile=file)] string...`

Description

Although PTC recommends editing the `si.properties` file in the GUI, you can retrieve the `si.properties` file from the database for manual editing using the `si getdbfile` command. Once you are finished editing this file, you can store it in the database again using the `si putdbfile` command.

Note

Access to configuration files is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document configuration files, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management files, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all configuration files.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--encoding=value`

  specifies the code set the file is saved in, for example, `en_US` (English, United States) or `ja_JP` (Japanese, Japan).
- `--input=value`

  specifies the name of the file on the local file system containing the input.
- *`string...`*

  specifies the path and name of the file in the database. To display a list of files in the database, type `si diag --diag=listdbfiles`, for example, `config\properties\si.properties`.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: si putdbfile
- Miscellaneous: ACL, diagnostics, options, preferences

si rejectcp

rejects a change package

Synopsis

`si rejectcp [--comments=value] [--commentsFile=value] [--reviewer=[user|group:<Group-Name>|super]] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] issue|issue:change package id...`

Description

`si rejectcp` rejects a change package under review. For example,

```
si rejectcp --reviewer=super 32434:3
```

If a reviewer of a change package determines that additional development must be completed to resolve an issue, the reviewer can reject the change package. A single reviewer rejection is enough to reject a change package, even if there are multiple reviewers. The creator of the change package can reject it without being listed as a reviewer.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--comments=value`

  specifies comments recorded in the review log with the vote.
- `--commentsFile=value`

  specifies a text file that contains the comments to be recorded in the review log with the vote.
- *--reviewer=[user|group:<Group-Name>|super]*

  specifies the capacity in which you are rejecting the change package.
- `user`

  casts a vote as an individual reviewer in the reviewer list.
- `group:<Group-Name>`

  casts a vote on behalf of the entire group (only one user from a group is necessary to vote on behalf of the entire group).
- `super`

  an overriding reject vote. A super reviewer is not required to be a listed reviewer for the change package. You must have the SuperReviewer permission to use this option.
- *issue...*
- *issue:change package id...*

  `issue` identifies a specific issue that contains all submitted change packages that you want to reject; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to reject; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si co, si cpissues, si createcp, si viewcps si acceptcp, si opencp, si discardcp
- Miscellaneous:ACL, diagnostics, options, preferences

si rename

renames an existing project member

Synopsis

`si rename [(--cpid=ID|--changePackageId=ID)] [--[no|confirm]closeCP] [--[no]confirm] [--[no]defer] [-f] [--issueId=ID] [(--newName=value)] [--[no|confirm]overwrite] [--[no]renameWorkingFile] [--[no|confirm]branchVariant] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...`

Description

`si rename` renames an existing project member. For example,

```
si rename --newName=si_addmember.xml c:/Documentation/Man_Pages/xml_man/si_add.xml
```

This command only changes the basename of the member and cannot be used to move the member into a different directory or project. To move members, use the `si move` command.

Before using `si rename`, note the following:

- You can only rename members within a directory. You cannot rename a member and move it to a different directory or project.
- Renaming a locked revision in a variant project moves the lock to the duplicate revision, even if the lock exists in the master project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no|confirm]closeCP`

  controls whether to close the associated change package.

  `--nocloseCP` means do not close the change package.

  `--confirmcloseCP` means ask before closing the change package.

  `--closeCP` always closes the change package.
- `--[no]confirm`

  controls whether to confirm the rename before proceeding.
- `--[no]defer`

  controls whether to delay the rename operation in the project until the deferred operation is submitted. The rename operation in the Sandbox still takes place immediately. This option is valid only if you are performing the command from a Sandbox.
- `-f`

  force the rename without confirmation.
- `--newName=value`

  specifies the new name of the renamed member.
- `--[no|confirm]overwrite`

  controls whether to overwrite the target (new name) working file if it exists. Specifying `--overwrite` means always do it, `--nooverwrite` means never do it, and `--confirmoverwrite` means always ask before doing it. This option is valid only if you are performing the command from a Sandbox.
- `--[no]renameWorkingFile`

  controls whether to rename the working file in the Sandbox immediately.
- `--[no|confirm]branchVariant`

  controls whether Integrity creates a branch for the renamed member.
- *`member...`*

  identifies a specific member. You can only specify one member at a time.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si add, si closecp, si co, si cpissues, si createcp, si diff, si drop si edit, si lock, si move si mergebranch, si rlog, si unlock, si viewcps
- Miscellaneous:

  ACL, diagnostics, options, preferences

si report

collates SCM data into text files for reporting by a third party utility

Synopsis

si report [--basename=*value*] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--projectRevision [--[no]failOnAmbiguousProject] =*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [-F|--selectionFile=*value*] [--status=*[none|gui|default]*]

Description

The `si report` command collates SCM data into a series of text files that can then be accessed by a third party reporting tool. The third party tool is responsible for formatting the data in the text files into the desired reporting format.

⚠ Caution

This command is deprecated in Integrity; however, it can still be used with a third party reporting tool.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--basename=`*value*

  Specifies the name of generated tables.

  `si report` uses the following suffixes to indicate the contents of the table:

  ```
  .txp                    project information
  .txm                    member information
  .txa                    archive information
  .txr                    revision information
  .txs                    symbols
  .txx                    access information
  ```

  `Data Fields`

  In the generated data files, the first line of each data file contains field names.

  The fields for the project information file (`.txp`) are:

  - *ProjId*

    (integer) A unique identifier of a project (Primary Key)
  - *ProjName*

    (string) The project path/name

  The fields for the member information file (`.txm`) are:

  - *ProjId*

    (integer) A unique identifier for the project, can be used as a key. Taken from project info (Foreign Key - 1:M).
  - *MemberName*

    (string) Member file path/name.
  - *MemberType*

    (string) Valid values are:

    - `a` archive

      other (only valid for projects imported from
    - `f` older versions of Integrity (formerly MKS

      Source or Source Integrity Standard)
    - `i` sub-project
  - *Revision*

    (string) Revision number of member (can be NULL).
  - *ArchId*

    (integer) A unique identifier of the member file, may be an archive id or another project id (can be NULL). Can be used as a primary key.

  The fields for the project information file (`.txa`) are:

  - *ArchId*

    (integer) A unique identifier of the member file, may be an archive id or another project id (can be NULL). Can be used as a primary key.
  - *ArchiveName*

    (string) File path/name, only for archives.
  - *Head*

    (string) Revision number of the head revision.
  - *Branch*

    (string) Branch number of a default branch if one has been set by `si archiveinfo –b`. (can be NULL).
  - *NumLocks*

    (integer) The number of locked revisions.
  - *Format*

    (string) One of text, binary or auto-detect.
  - *Encrypted*

    (integer) Set to 1 if the archive is encrypted. This option is not supported in Integrity.
  - *TotalRevs*

    (integer) Total number of revisions in the archive.
  - *Branches*

    (integer) Total number of branches in the archive.
  - *BranchRevs*

    (integer) Total number of revisions on branches.
  - *Comment*

    (string) Comment delimiter if set (can be NULL). This field is historical.
  - *Description*

    (string) Text describing an archive. This text is entered when the archive is first created.

  The fields for the revision information file (`.txr`) are:

- *ArchId*

  (integer) A unique identifier of the member file, may be an archive id or another project id (can be NULL). Can be used as a primary key.
- *RevNum*

  (string) The revision number of this revision (unique per ArchId).
- *Date*

  (string) The date this revision was entered, in the format *yyyy/mm/dd hh:mm:ss*.
- *Author*

  (string) The author's name (usually the author's user ID).
- *State*

  (string) The state assigned to this revision; this is taken from state list items).
- *LinesAdded*

  (integer) The number of lines added by this revision; derived from delta information.
- *LinesDeleted*

  (integer) The number of lines deleted by this revision.
- *Locker*

  (string) The name of the user holding a lock on this revision (usually the user's user ID) and the time when the revision was locked.
- *LogMsg*

  (string) Text description of the changes made by this revision as entered at check in.

The fields for the symbol information file (`.txs`) are:

- *ArchId*

  (integer) A unique identifier of the member file, may be an archive id or another project id (can be NULL). Can be used as a primary key.
- *RevNum*

  (string) The revision number of a revision that has an associated symbolic label. The revision number may not be unique in a list.
- *Symbol*

  (string) The symbolic label.

The fields for the access information file (`.txx`) are:

- *ArchId*

  (integer) A unique identifier of the member file, may be an archive id or another project id (can be NULL). Can be used as a primary key.
- *UserId*

  (string) Name (user ID) of user who is included in an access list (Source Integrity Standard only) for an archive. The archive and project ID numbers (*ArchId* and *ProjId*) are unique for any set of reports (though they may change from one use of report to another). They may be used as keys.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si projectinfo, si projects, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si restoreproject

restores a project to a particular revision

Synopsis

```
si restoreproject [(-r rev|--revision=rev)] [(--reason=reason|--reasonFile=file)] [--resultingSubprojectConfiguration=
[legacy|ondevpath|build|retainconfiguration]] [--devpath=path] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--
[no]failOnAmbiguousProject] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|-
-yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [-F|--
selectionFile=value] [--status=[none|gui|default]]
```

Description

`si restoreproject` restores a project to a particular checkpoint.For example,

```
si restoreproject --project=c:/Aurora_Program/bin/Libra/project.pj --revision=1.34 --reason="Restore due to Exception Error"
```

When you apply the `si restoreproject` command, Integrity recursively checkpoints the selected project, then modifies it to reflect the member list of the target checkpoint. Any further development then proceeds from the restored checkpoint.

Restoring a project is useful when development must revert back to an earlier version and there are no plans to proceed from the current version of the project. Restoring a project is not an option when the goal is to test a particular version. The project is always checkpointed before and after the restore.

Note

- You cannot restore a build project using the `si restoreproject` command.
- The `si restoreproject` command can potentially restore and checkpoint dropped subprojects that existed at the target revision, even if they are not currently a member of the project.
- Do not use the `si restoreproject` command to create a new development branch from a project checkpoint. For new development paths, you should instead create a new development path.
- To restore a variant project to a specific project revision, the development path must exist in all subprojects referenced by the project checkpoint.

Defining the Configuration of Subprojects When Restoring a Project

When restoring a project from a reference checkpoint, you can define the resulting configuration of subprojects in the project. All subproject configuration options result in the same subprojects and member contents (taken from reference checkpoint); only the configuration of the subprojects is different. Any new subprojects that did not exist in the reference checkpoint are dropped and any subprojects that no longer exist are re-added.

The ability to specify the resulting subproject configuration is useful for scenarios where you want to control how subprojects are affected by the restore operation. For example, if you specify the `--resultingSubprojectConfiguration=ondevpath` option, only the current development path will be affected since all the subprojects (that are not configured as build in the reference checkpoint) will be configured to the current development path. This means that the member changes will be performed on the subprojects on this development path when restoring the project.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--reason=`*reason*
- `--reasonFile=`*file*

  specifies the reason text to be associated with restoring the project, or specifies a file where text can be retrieved.

  Note
  If the reason text includes spaces, enclose the text in quotes.

- `--resultingSubprojectConfiguration=[legacy|ondevpath|build|retainconfiguration]`

  specifies the resulting configuration of subprojects in the project being restored.

  `legacy` specifies that any subprojects that were configured the same as their immediate parent in the reference checkpoint are configured on the same development path as their immediate parent (or mainline if their immediate parent's resulting configuration is on the mainline). All other subprojects that are configured differently from their immediate parent are configured the same way that they were configured in the reference checkpoint.

  `ondevpath` specifies that all subprojects are configured on the same development path as the project you are restoring to (or mainline if the project is currently on the mainline). Any subprojects that were configured as build in the reference checkpoint continue to be configured as build, pointing to the revision from the reference checkpoint.

  `build` specifies that all subprojects are configured as build subprojects, pointing to the revision from the reference checkpoint. Shared subprojects are configured as shared build subprojects.

  `retainconfiguration` specifies that the current subproject configuration does not change, regardless of the configuration in the reference checkpoint. Any subprojects from the reference checkpoint that were dropped are re-added and configured as build subprojects. Any subprojects that are currently configured as build retain their configuration; however, their revision is updated to point to the same revision from the reference checkpoint.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint
- Miscellaneous: ACL, diagnostics, options, preferences

si restrictproject

restricts changes from being made to a project hierarchy

Synopsis

```
si restrictproject [--project=value] [--user=name] [--password=password] [--hostname=server] [--port=number ] [--
allowPrincipals=value] [--denyPrincipals=value] [--removePrincipals=value] [--removeAll=value] [--
[no|confirm]updateOnRestrictedByConflict] [--selectionFile=value] [--settingsUI=[gui|default]] [--
status=[none|gui|gui.actions|default]] [--forceConfirm=[yes|no]] [--gui] [--sandbox=value] [--cwd=value] [--selectionFil=value] [--
[no]batch] [--quiet] [--usage] [--settingsUI=[gui|default]] [--user] [--usage] [--yes] [--[no]failOnAmbiguousProject] [-F=file]
```

Description

`si restrictproject` restricts changes from being made to a project hierarchy. When the release is finalized, the mainline or branch of a project can be restricted recursively by the user with `RestrictProject` permission. The project can be restricted for either an individual user or for a group.

During project restriction, specific users or groups can be permitted to perform changes to a restricted project. Such users may only perform changes according to their existing permissions

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]updateOnRestrictedByConflict`

  specifies if to update project restriction even if project is already restricted by another user.
- `--allowPrincipals=user:user1, group:group1`

  specifies users or groups with allowance to perform changes according to their existing permissions to a restricted project.
- `--denyPrincipals=user:user1, group:group1`

  specifies users or groups denied to perform changes to a restricted project.
- `--removePrincipals=user:user1, group:group1`

  specifies users or groups to remove from the list of principals that are permitted or denied to modify the restricted project.
- `--removeAll`

  remove all users or groups from the list of principals that are permitted or denied to modify the restricted project.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si unrestrictproject
- Miscellaneous: ACL, diagnostics, options, preferences

si resync

updates a Sandbox with the member revision

Synopsis

`si resync [--mergeType=[confirm|cancel|automatic|manual]] [--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]] [--[no]byCP] [--[no]confirm] [--[no]confirmPopulateSparse] [--[no]includeDropped] [--[no|confirm]merge] [--[no|un]expand] [-f] [--[no|confirm]overwriteChanged] [--[no|confirm]overwriteIfPending] [--[no|confirm]overwriteDeferred] [--[no]overwriteUnchanged] [--[no|confirm]removeOutOfScope] [--[no]populate] [--[no]restoreTimestamp] [(-R|--[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--[no|confirm]downgradeOnLockConflict [--filter=filteroptions] [(-S sandbox|--sandbox=sandbox)] [--[no]awaitServer] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] current or dropped member/subproject...`

Description

`si resync` compares the contents of project members with their corresponding working files in the Sandbox and replaces the working file with an exact copy of the member revision, if differences are detected or if no working file exists. If the working file is writable, you are optionally prompted for confirmation before it is replaced. This command has no effect on working files that are identical to the member revision.

The following is an example of the command syntax:

```
si resync c:/Aurora_Program/bin/Libra/project.pj
```

Caution

This command overwrites files, even ones that you have locked and that have changed since you last checked them out. Be certain of your response to any prompts indicating that the files will be overwritten - once replaced, you cannot get back your changes. If files have been dropped from a project, resynchronizing deletes them.

If the revision that your working file is based on is locked by you, your lock is automatically moved to the member revision before the resync proceeds. If another user already has an exclusive lock on the member revision, you are prompted to downgrade your lock to non-exclusive.

If you are working in a sparse Sandbox, resynching deletes your working file, unless the revision that it is based on is locked by you.

When working in your Sandbox, you can use the resynchronize by change package option.

The Resync By Change Package Option

The Resync by CP `(--[no]byCP)` option is primarily a tool for developers. When you want to resynchronize files in your Sandbox, you usually do so by choosing individual files and then using `si resync` command. However, if the files you are resynchronizing have changes that are linked to other files, the standard resync operation would not include those related files. To resynchronize all related files, you would have to manually search for all the changes associated with the change package on the member you are resynchronizing.

The Resync by CP option automates this process by searching the change package specified on the member revision you are resynchronizing and then bringing the changes from the project to your Sandbox.

While the Resync CP option searches all files related to a selected change package, and all the change packages that may be associated with the related files, the Resync By CP option only processes the change packages associated with the member you are resynchronizing.

How Does the Resync By CP Option Work?

The functioning of Resync by CP is affected by the options you choose for si resynccp. For more information, see si resynccp.

When Should I Use the Resync By CP Option?

Developers should use Resync By CP when they want to ensure that they have all dependent changes associated with a member revision, even if these changes are contained in other files. For example, a developer needs to check out (locked) a file and modify it. The developer finds that other revisions have been checked in since the member was resynchronized in the Sandbox. Because the Sandbox is quite large and contains many unrelated changes, the developer does not want to perform a standard resynchronization. The Resync By CP option can be used in this situation.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `Note:`
- If the working file of the member you are resyncing has been modified, Integrity asks you to confirm that you want to merge your modifications into the working file.
- `--mergeType=[confirm|cancel|automatic|manual]`

  specifies how you want to merge changes from the working file in your Sandbox into the member revision.

  `--mergeType=confirm` prompts you to confirm a merge type.

  `--mergeType=cancel` cancels the selected merge type.

  `--mergeType=automatic` completes the merge process without launching the Visual Merge tool.

  Caution

  While Integrity and Visual Merge are capable of performing automatic merging, PTC cannot guarantee that the merged results are \u201ccorrect\u201d. PTC recommends that you examine and test the merged results before checking them into the repository.

  `--mergeType=manual` completes the merge operation through the Visual Merge tool or a third party merge tool, depending on your Difference and Merge Tool preferences. The merge tool launches, displaying the revisions you want to merge. For more information on the Visual Merge tool, refer to the *PTC Integrity User Guide*.
- `--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]`
- specifies what to do when conflicts occur during a merge.

  `--onMergeConflict=confirm` prompts you to confirm a merge conflict option.

  `--onMergeConflict=cancel` cancels the merge.

  `--onMergeConflict=mark` marks the working file indicating that merging is required without completing all of the merge related tasks. This provides the time you may need to investigate conflicts or edit difference blocks before finishing the merge.

  `--onMergeConflict=launchtool` launches the Visual Merge tool to resolve the conflicts; all non-conflicting blocks will already have been applied.

  `--onMergeConflict=highlight` indicates conflicts in the file with the following characters: "<<<<<<" and ">>>>>>". You are responsible for resolving merge conflicts before checking in the changes.

  Note

  `--onMergeConflict=launchtool` does not require the `-g` or `--gui` options.

  `--onMergeConflict=error` indicates merge conflicts with an error message prompt.
- `--[no]byCP`

  controls whether to cause Integrity to operate in change package mode. Integrity automatically searches the change package associated with the member and resynchronizes all member files contained in that change package.

Note

If there are no revisions in the change package to resynchronize, a normal resync is performed.

- `--[no]confirm`

  controls whether to proceed without confirmation messages when working with a change package (applies only when change packages are enabled).
- `--[no]confirmPopulateSparse`

  controls whether to proceed without confirmation messages when populating a sparse Sandbox.
- `--[no]includeDropped`

  controls whether to delete dropped members from your Sandbox.
- `--[no|confirm]merge`

  controls whether to merge changes from the working file in your Sandbox into the member revision.

  `--merge` means always merge.

  `--nomerge` means never merge.

  `--confirmmerge` means always ask before merging. When asked to confirm the merge, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being resynchronized with.

Note

- Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.
  - By default, Integrity prompts you for the type of merge (manual or automatic) to perform and what action to take if a conflict is encountered.

- `--[no|un]expand`

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

```
$Author: Warner, Carrie (cwarner) $
$CompanyInfo$
$Date: 2015/11/30 15:01:47EST $
$Header: si_resync.dita 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner) Exp  $
$Id: si_resync.dita 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner) Exp  $
$Locker:  $
$Log: si_resync.dita  $
Revision 1.2 2015/11/30 15:01:47EST Warner, Carrie (cwarner)
XML tagging fixes
Revision 1.1 2015/10/29 10:24:59EDT Flett, David (dflett)
Initial revision
Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
$Revision: 1.2 $
$Name:  $
$ProjectLabel:  $
$ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
$ProjectSetting  $
$ProjectRevision: Last Checkpoint: 1.1.1.6 $
$RCSfile: si_resync.dita $
$Revision: 1.2 $
$SandboxSetting  $
$Setting  $
$Source: si_resync.dita $
$State: Exp $
```

- `-f`

  overwrites the working file if it has changed since it was last checked in. This is equivalent to specifying `--overwriteChanged`.
- `--[no|confirm]overwriteChanged`

  controls whether to overwrite the working file if it has changed since it was last checked in.

  `--overwriteChanged` means always overwrite.

  `--nooverwriteChanged` means never overwrite.

  `--confirmoverwriteChanged`means always ask before overwriting. When asked to confirm overwriting the working file, you have the option of specifying `y`, `n` or `d`. Specifying d displays the differences between the member's working file and the revision it is being resynchronized with.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no|confirm]removeOutOfScope`

  controls whether to remove a current member's working file if it does not match the Sandbox scope definition, for example, if the scope definition or member changes (such as modified member attributes). Sandbox Scope defines what project members are included in the Sandbox, transferring specific members from the Integrity Server to the Sandbox directory when the Sandbox is created and controlling what members display in the Sandbox view.

  `--removeOutOfScope` means always remove the working file if it does not match the Sandbox scope.

  `--noremoveOutOfScope` means never remove the working file if it does not match the Sandbox scope.

  `--confirmremoveOutOfScope` means always ask before removing the working file if it does not match the Sandbox scope. When asked to confirm removing the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the member revision it is being reverted to.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no|confirm]overwriteIfPending`
- controls whether to overwrite the working file if there is a pending operation on the revision corresponding to the working file.

  `--overwriteIfPending` means always overwrite

  `--nooverwriteIfPending`means never overwrite

  `--confirmoverwriteIfPending` means always ask before overwriting. When asked to confirm overwriting the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being resynchronized with.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no|confirm]overwriteDeferred`
- controls whether to overwrite the working file if there is a deferred operation on the revision corresponding to the working file.

  `--overwriteDeferred` means always overwrite

  `--nooverwriteDeferred means never overwrite`

  `--confirmoverwriteDeferred` means always ask before overwriting. When asked to confirm overwriting the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being resynchronized with.

Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no]overwriteUnchanged`

  controls whether to overwrite files that have not been modified in the Sandbox. This may be useful for forcing files to pick up new keyword expansions or timestamps even if their content has not changed.
- `--[no]populate`

  controls whether to populate the Sandbox with a working file for the specified member. If you are working with a sparse Sandbox (see si createsandbox), specifying `--populate` with this command overrides the default sparse Sandbox behavior of removing working files.
- `--[no]restoreTimestamp`

  controls whether to restore the timestamp of the working file to the timestamp of the revision in the member history.
- `--[no|confirm]downgradeOnLockConflict`

  If the revision that your working file is based on is locked by you, your lock is automatically moved to the member revision before the resync proceeds. This option controls whether to downgrade your lock to non-exclusive if another user has an exclusive lock on the member revision.

  `--downgradeOnLockConflict` means always downgrade

  `--nodowngradeOnLockConflict` means never downgrade

  `--confirmdowngradeOnLockConflict` means always ask before downgrading
- `--[no]awaitServer`

  instructs the client to repeatedly attempt the operation if the Integrity Server does not respond.
- *current or dropped member/subproject...*

  identifies a specific member or subproject that currently exists in the project, or one that has been dropped from the project but still exists in the Sandbox. Use spaces to specify more than one. If you specify a dropped member or subproject, and if you specify `--includeDropped`, this command deletes the corresponding working file.

Note

If you specify an out of scope member with no working file, the resync operation does not create a working file for the member. For more information on Sandbox scope, see the`si configuresandbox` and `si createsandbox` commands.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si applycp, si ci, si co, si resynccp, si revert
- Miscellaneous: ACL, diagnostics, options, preferences

si resynccp

merges change package members to an Integrity sandbox

Synopsis

```
si resynccp [--allowOpenCP] [--[no]alreadyInProjectIsError] [--backFill=[cp|revision|error|skip|ask]] [--mergeType=
[confirm|cancel|automatic|manual]] [--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]] [--[no]confirm] [--
notify=[never|onCompletion|onError]] [--[no]ignoreBranches] [--[no]ignoreServer] [--[no]continueOnErrors] [--[no|confirm]merge] [--
[no|confirm]mergeOnBranch] [--[no]failOnAmbiguousProject] [--[no|confirm]createVariants] [--[no]otherProjectIsError] [--
changePackageID=value] [--cpid=ID] [(--issueID=value)] [--[no]spanProjects] [--[no]useMaster] [--[no]verbose] [(-S sandbox|--
sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--
selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]ignoreUpdateRevision] [--subprojectPropagation=[explicit|implicit]] [-
-[no|confirm]resyncIfOutOfScope] issue|issue:change package id...
```

Description

`si resynccp` allows you to preview the changes listed in change packages in the context of a sandbox before you propagate them to the project. This command searches all files related to a selected change package and all change packages that may be associated with the related files. The `si resynccp` command is most useful for incorporating new software features or bug fixes during the software development process.

Note

If you are `resynchronizing` a change package to the sandbox of an extendable development path, subprojects in the project are extended automatically, which modifies the project. This is a special case because normally the `si resynccp` command modifies the project only when a propagation change package is specified. *Extenddevpath* permission is required when operating against the sandbox of an extendable development path.

For more information on how the `si resynccp` command works and how you can use it in your development environment, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--allowOpenCP`

  allows Integrity to resynchronize open change packages. Open change packages are not allowed if a propagation change package is specified using the `--cpid` option.
- `--[no]alreadyInProjectIsError`

  Specifies whether Integrity terminates the operation if the change being applied has already been applied to the project. If this setting is negated `--noalreadyInProjectIsError`, then the information is displayed as a warning.
- `--backFill=[cp|revision|error|skip|ask]`

  Specifies the way Integrity treats historic revisions required by the specified change package. For example, `--backfill=ask` means Integrity asks you to specify the change packages you want to exclude from the operation.

  The available values for the `--backFill` option include:
- `--backFill=cp`

  recursively chooses all historic revisions required by the specified change packages and applies them by updating member revisions, adding files, or dropping files.
- `--backFill=revision`

  processes only the specified change package(s) and chooses only directly associated revisions. It does not process any change packages that are associated with intermediate revisions.
- `--backFill=error`

  terminates the operation if other change packages are required but are not specified.
- `--backFill=skip`

  causes Integrity to merge around specified backfill revisions. Because the Apply CP command does not perform merging, this is treated as an error in Apply CP, allowed in Resync CP.
- `--backFill=ask`

  allows you to specify the change packages you want to include.
- `--[no]ignoreUpdateRevision`

  ignores update revision change package entries when performing the command. The default is to include update revision entries.
- `--mergeType=[confirm|cancel|automatic|manual]`

  specifies how you want to merge changes from the specified change package revisions into the working file in your Sandbox.

  `--mergeType=confirm` prompts you to confirm a merge type.

  `--mergeType=cancel` cancels the merge.

  `--mergeType=automatic` completes the merge process without launching the Visual Merge tool.

Note

Caution: Even if a no conflict is reported as a result of a merge operation, there can still be inconsistencies in the merge results that can only be detected by someone with an understanding of the context of the change. Therefore, anytime a merge is performed, PTC recommends that you examine and test the merged results before checking them into the repository.

`--mergeType=manual` completes the merge operation through the Visual Merge tool or a third party merge tool, depending on your Difference and Merge Tool preferences. The merge tool launches, displaying the revisions you want to merge. For more information on the Visual Merge tool, refer to the *PTC Integrity User Guide*.

- `--onMergeConflict=[confirm|cancel|mark|launchtool|highlight|error]`

  specifies what to do when conflicts occur during a merge.

  `--onMergeConflict=confirm` prompts you to confirm a merge conflict option.

  `--onMergeConflict=cancel` cancels the merge.

  `--onMergeConflict=mark` marks the working file indicating that merging is required without completing all of the merge related tasks. This provides the time you may need to investigate conflicts or edit difference blocks before finishing the merge.

  `--onMergeConflict=launchtool` launches the Visual Merge tool to resolve the conflicts, all non-conflicting blocks will already have been applied.

  `--onMergeConflict=highlight` indicates conflicts in the file with the following characters: "<<<<<<" and ">>>>>>". You are responsible for resolving conflicts in the working file.

Note

`--onMergeConflict=launchtool` does not require the `-g` or `--gui`options

- `--[no]confirm`

  Confirm the actions before starting them.

  `--[no|confirm]createVariants`
- controls whether to confirm the creation of variant projects as needed.
- `--notify=[never|onCompletion|onError]`

  specifies whether to display a report when the command is complete. The report details the operations that were performed and any errors that were encountered.
- `--notify=never`

  never displays the report.
- `--notify=onCompletion`

  always displays the report.
- `--notify=onError`

  displays the report if any errors were encountered.
- `--[no]ignoreBranches`

  Causes Integrity to use the most recent revision when it encounters two revisions of the same member on two development paths in the change package being applied.
- `--[no]ignoreServer`

  Causes Integrity to perform the Apply CP operation even if the change package members reside on different servers.
- `--[no]continueOnErrors`

  Causes Integrity to continue to process the change package if errors occur while resychronizing. If the `--notify` option is specified as `onCompletion` or `onError`, any errors are reported when the command is complete.
- `--[no|confirm]merge`

  controls whether to confirm all merge operations before completing them.
- `--[no|confirm]mergeOnBranch`

  controls whether to cause Integrity to perform a merge if the target revision is on a branch. Integrity differences the two file revisions and merges any changes into the working file without modifying its revision number. You must then check in the working file to advance the revision to the next available revision number.
- `--[no]otherProjectIsError`

  Causes Integrity to terminate the command if the member or subproject is in another project, for example, if the change package contains multiple entries in different top level projects. Change package entries referring to other projects are therefore treated as an error.
- `--cpid=`*`ID`*
- `--changePackageID=`*`value`*

  specifies whether to create a propagation change package that is detached from the set of change packages it propagates (where `value` is the change package ID number). All the required operations are automatically represented as deferred entries (or pending entries for subproject operations). You can then discard entries that you do not want represented in the final application of changes to the project. You can also add or change entries in the propagation change package as required.
- `--issueID=`*`value`*

  the container ID used to find the propagation change package. If the Integrity integration is enabled, the container ID is the same as the issue ID.
- `--[no]spanProjects`

  Causes Integrity to apply the command to any member or subproject specified in the change package, even if this involves multiple projects. This option also applies across all Sandboxes and results in a search of local Sandboxes for all entries in the change package. When multiple Sandboxes are found, Integrity uses the Sandbox that contains the longest common directory prefix matching the Sandbox that the command is run from.
- `--[no]useMaster`

  causes Integrity to operate on the top-level Sandbox.
- `--[no]verbose`

  Means Integrity includes additional information to track the current state of the command.
- `--subprojectPropagation=`*`[explicit|implicit]`*

  specifies how to apply subproject changes required by the specified change packages.

  `--subprojectPropagation=`*`explicit`* creates, adds, drops, or moves a subproject only if there is a explicit command to do so in the change package.

  `--subprojectPropagation=`*`implicit`* creates, adds, drops or moves a subproject if the operation is implicitly required based on the change package entries. For example, if you are adding a member that is part of a subproject that does not exist in the targer project being updated, the subproject is added.
- `--[no|confirm]resyncIfOutOfScope`

  Specifies if to include entries for members in the change package that do not match the scope definition of the sandbox.
- *`issue...`*
- *`issue:change package id...`*specifies the identification numbers for the change package you want to apply. For example, `11804:2`.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si applycp, si resync
- Miscellaneous: ACL, diagnostics, options, preferences

si retargetsandbox

points a Sandbox at a different project configuration

Synopsis

`si retargetsandbox [--[no]openView] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--[no]failOnAmbiguousProject] [--retargetServer] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si retargetsandbox` allows you to point your Sandbox at a different project configuration. For example, a developer can change a variant Sandbox to point to a different variant project, or a buildmaster can change a build Sandbox to point to a different build. You can also change the type of project that the Sandbox points to, for example, you can change a normal Sandbox to point to a variant project.

Note

- You can only retarget a Sandbox that contains deferred member operations or locks if you are specifying the `--retargetServer` option.
- You can only retarget top level Sandboxes.
- You can only retarget to another type of the same project that the Sandbox is currently pointing to.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]openView`

  controls whether to open the Sandbox view after this Sandbox is retargeted. If `--noopenView` is used, the Sandbox view does not open.
- `--[no]failOnAmbiguousProject`

  Specifies if to abort command operation when multiple projects correspond to a flat project string.
- `--retargetServer`

  Specifies that the Sandbox use a different Integrity Server (retargets the Sandbox from its existing Integrity Server to another Integrity Server). This option can be used when the current server is offline (the new server must be online). This option cannot be specified with `--projectRevision` or `--devpath`.

  The option takes the following syntax:

  `si retargetsandbox --retargetServer --hostname=NewServer --port=NewPort SandboxPath`

  where

  - *NewServer*

    is the hostname of the Integrity Server to which the Sandbox is being retargeted.
  - *NewPort*

    is the port number of the Integrity Server to which the Sandbox is being retargeted.
  - *SandboxPath*

    is the path to the Sandbox that is being retargeted by the command.

  For example:

  `si retargetsandbox --retargetServer --hostname=production2 --port=7001 C:/Sandboxes/code/project.pj`

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si createdevpath, si createsandbox, si dropsandbox, si importsandbox, si sandboxes, si viewsandbox
- Miscellaneous: diagnostics, options, preferences

si revert

overwrites a sandbox file with a fresh copy of the working file, discarding changes

Synopsis

si revert [--[no|un]expand] [-f] [--[no|confirm]overwriteChanged] [--[no|confirm]overwriteDeferred] [--[no]overwriteUnchanged] [--[no|confirm]removeOutOfScope] [--[no]restoreTimestamp] [(-R|--[no|confirm]recurse)] [--filter=*filteroptions*] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *sandbox member...*

Description

si revert restores a working file to a state before changes were made, by overwriting the Sandbox file with a fresh copy of the working file at the same revision you currently have in your Sandbox, not the member revision. If you had the member locked, it is unlocked.

Note

If the revision you have locked is not the working revision, the lock is retained.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no|un]expand

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the *PTC Integrity User Guide*. Possible keywords are:

  ```
  $Author: Warner, Carrie (cwarner) $
  $CompanyInfo$
  $Date: 2015/11/30 16:23:08EST $
  $Header: si_revert.dita 1.3 2015/11/30 16:23:08EST Warner, Carrie (cwarner) Exp  $
  $Id: si_revert.dita 1.3 2015/11/30 16:23:08EST Warner, Carrie (cwarner) Exp  $
  $Locker:  $
  $Log: si_revert.dita  $
  Revision 1.3 2015/11/30 16:23:08EST Warner, Carrie (cwarner)
  XML tagging fixes
  Revision 1.2 2015/11/30 15:01:45EST Warner, Carrie (cwarner)
  XML tagging fixes
  Revision 1.1 2015/10/29 10:25:00EDT Flett, David (dflett)
  Initial revision
  Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
  $Revision: 1.3 $
  $Name:  $
  $ProjectLabel:  $
  $ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
  $ProjectSetting  $
  $ProjectRevision: Last Checkpoint: 1.1.1.8 $
  $RCSfile: si_revert.dita $
  $Revision: 1.3 $
  $SandboxSetting  $
  $Setting  $
  $Source: si_revert.dita $
  $State: Exp $
  ```

- -f

  overwrites the working file if it has changed since it was last checked in. This is equivalent to specifying --overwriteChanged.

- --[no|confirm]overwriteChanged

  controls whether to overwrite the working file if it has changed since it was last checked in.

  --overwriteChanged means always overwrite

  --nooverwriteChanged means never overwrite

  --confirmoverwriteChanged means always ask before overwriting. When asked to confirm overwriting the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being reverted to.

  Note

  Specifying the --yes or --no options automatically answers y or n to the confirmation question, and also automatically answers y orn to all questions asked.

- --[no|confirm]removeOutOfScope

  controls whether to remove a current member's working file if it does not match the Sandbox scope definition, for example, if the scope definition or member changes (such as modified member attributes). Sandbox Scope defines what project members are included in the Sandbox, transferring specific members from the Integrity Server to the Sandbox directory when the Sandbox is created and controlling what members display in the Sandbox view.

  --removeOutOfScope means always remove the working file if it does not match the Sandbox scope.

  --noremoveOutOfScope means never remove the working file if it does not match the Sandbox scope.

  --confirmremoveOutOfScope means always ask before removing the working file if it does not match the Sandbox scope. When asked to confirm removing the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being reverted to.

  Note

  Specifying the --yes or --no options automatically answers y or n to the confirmation question, and also automatically answers y orn to all questions asked.

- --[no|confirm]overwriteDeferred

  controls whether to overwrite the working file if there is a deferred operation on the member.

  --overwriteDeferred means always overwrite

  --nooverwriteDeferred means never overwrite

  --confirmoverwriteDeferred means always ask before overwriting. When asked to confirm overwriting the working file, you have the option of specifying y, n or d. Specifying d displays the differences between the member's working file and the revision it is being reverted to.

  Note

Specifying the `--yes` or `--no` options automatically answers `y` or `n` to the confirmation question, and also automatically answers `y` or `n` to all questions asked.

- `--[no]overwriteUnchanged`

  controls whether to overwrite files that have not been modified in the sandbox. This may be useful for forcing files to pick up new keyword expansions or timestamps even if their content has not changed.
- `--[no]restoreTimestamp`

  controls whether to restore the timestamp of the working file to the timestamp of the revision in the member history.
- *sandbox member...*

  identifies a specific member; use spaces to specify more than one member.

Note

If you specify an out of scope member with no working file, the revert operation does not create a working file for the member. For more information on Sandbox scope, see the si configuresandbox and si createsandbox commands.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si resync
- Miscellaneous: diagnostics, options, preferences

si revertcp

reverts a change package

Synopsis

`si revertcp [--cpid=ID|--changePackageId=ID] [(-f|--[no]confirm)] [--issueId=value] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--[no]spanSandboxes] [(-?|--usage)] [(-Ffile|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--hostname=server] [--password=password] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--user=name] issue|issue:change package id`

Description

`si revertcp` reverts the changes made by the entries in the specified change package. The command reverts the changes by performing new changes, which may result in new revisions of members. Even though the changes are reverted, they still appear in the history where applicable.

The `si revertcp` command does not perform an operation for every type of entry in the specified change package. The following entry types are supported:

- `Add` (results in a pending `Drop` operation)
- `Drop` (results in a pending `Add` operation)
- `Update` (results in the need for a merge)
- `Update Revision` (results in the need for a merge)
- `Add From Archive` (results in a pending `Drop` operation)

Even if an entry type is supported, there may be legitimate reasons why the changes needed to revert the entry could not occur (such as insufficient permissions assigned to you). If Integrity cannot perform the changes needed to revert an entry, the information is reported to you after the command completes. You can use that information to manually make the needed changes where possible.

Note

- The command only reverts a single closed change package.
- A destination change package is required, to record the operations needed to revert the change package.
- The command requires a Sandbox context to perform the operations needed to revert the change package.
- You may need to manually revert some of the changes yourself. The command returns a list of change package operations that could not be reverted
- If the merges are for revisions on the same member in the same Sandbox (contiguous merges), they are collapsed into one operation.
- For reverting a change package that contains an update revision, if the revision of the member being updated to in a change package is the same or later in the current project configuration, the result is a merge of the differences out of the working file. If the revision in the change package is earlier in the member history than the member revision of current project configuration, then the result will be a merge in instead of merge out. For example, if the change package has an update revision to 1.1, the revision of the member in the project is 1.3, the result is a merge in the member history. If the change package has an update revision to 1.2 and the revision of the member in the project was 1.2 (or 1.1) the result is a merge out in the member history.
- The `si revertcp` command does not implicitly extend subprojects in an extendable development path. Before using this command, you must manually extend the extendable development path to the appropriate subprojects. For more information, see si extenddevpath.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--cpid=ID`
- `--changePackageId=ID`

  specifies the change package used to record the operations needed to revert the change package selection.
- `--issueId=value`

  specifies the issue that contains the change package used to record the operations needed to revert the change package selection. The issue must contain only one open change package. The issue can contain any number of closed change packages.
- `-S sandbox`
- `--sandbox=sandbox`

  specifies the location of a Sandbox to perform the operations needed to revert the specified change package. Locations that include spaces must be enclosed by quotes.
- `--[no]spanSandboxes`

  specifies whether or not to search outside of the current Sandbox (the Sandbox that `si revertcp` is run from) to locate a Sandbox that contains the project on which to perform the operations needed to revert the change package entries. When multiple Sandboxes are found, Integrity uses the Sandbox that contains the longest common directory prefix matching the Sandbox that the `si revertcp` command is run from. By default, this option is enabled.
- *issue*
- *issue:change package id*

  specifies the change package to revert. Only a closed change package can be reverted. The change package can be specified one of the following ways:
  - *issue* identifies a specific issue that contains the change package to revert. The issue must contain only one closed change package.
  - *issue:change package id* identifies a specific change package to revert.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si viewcps
- Miscellaneous: ACL, diagnostics, options, preferences

si revisioninfo

displays revision information

Synopsis

si revisioninfo [--[no]changePackage] [--[no]labels] [(-r *rev*|--revision=*rev*)] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--lockRecordFormat=*value*] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] [--[no]locate] *member...*

Description

si revisioninfo displays revision information for a member, for example:

```
si revisioninfo c:\Documentation\Man_Pages\xml_man\si_add.1.xml
```

displays

```
Member Name: c:\Documentation\Man_Pages\xml_man\si_add.1.xml
Sandbox Name: c:\Documentation\Man_Pages\project.pj
Revision: 1.7
Created By: pmolinas on Dec 21, 2012 - 11:00 AM
State: state_a
Revision Description:
Updated for Technical Review
Has Checksum: Yes
Labels:
TechReview1
Change Package:
ID: 1:1
Member Revision:
In project C:\Aurora_Program\Documentation\project.pj, this revision
is member revision on:
development path:Service Pack 1
```

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no]changePackage

  controls whether to display change package information (applies only when change packages are enabled).
- --[no]labels

  controls whether to display member labels.
- --[no]locate

  controls whether to display the project and development path that the member is a member revision in.

  Note
  If a revision is member revision in a shared subproject, only the original (canonical) project path displays. Any projects where the subproject was added as shared do not display.

  Only projects that you have the OpenProject permission for display.
- --lockRecordFormat=*value*

  defines the format for displaying lock information for the revision. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:
  - {revision}

    displays the revision that is locked.
  - {locker}

    displays the user who locked the revision.
  - {locktype}

    displays the type of lock on the revision (exclusive or non-exclusive).
  - {locktimestamp}

    displays the time when the revision was locked.
  - {lockcpid}

    displays the change package associated with the lock on the revision.
  - {project}

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - {devpath}

    displays the name of the development path where the lock on the revision was made from.
  - {sandbox}

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
  - {hostname}

    displays the hostname of the computer that locked the the revision.
  - {hascpid}
  - displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
  - {hassandbox}

    displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
  - {hasdevpath}

    displays 1 if the lock was made from a development path, 0 if it wasn't.
  - {member}

    displays the name of the locked revision.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:
  si archiveinfo, si memberinfo, si mods, si rlog
- Miscellaneous:
  ACL, diagnostics, options, preferences

si rlog

displays the revision log

Synopsis

`si rlog [--format=value] [--headerFormat=value] [--noFormat] [--noHeaderFormat] [--noTrailerFormat] [(-r rev|--revision=rev)] [--range=value] [--trailerFormat=value] [--rfilter=filteroptions][--fields=field1[:width1],field2[:width2]...] [--[no]failOnAmbiguousProject] [--height=value] [--width=value] [--lockRecordFormat=value] [--lockseparator=value] [--maxTrunkRevs=value] [-xvalue [-yvalue [(-R |--no[confirm]recurse] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [--[no]persist] [--cwd=directory] [(-F file|--selectionFile=file)] [--forceConfirm=[yes|no]] [(-N|--no)] [--[no]batch] [--quiet] [(-?|--usage)] [(-Y|--yes)] member...`

Description

`si rlog` displays the revision log and general archive information for every revision in a member history. The formatting is, by default, suitable for user viewing but it is also programmable, suitable for use in scripts. See the `--format=value` option description below.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

For a member, first the `--headerFormat` is processed. Then each revision is processed and formatted using the `--fields` or `--format` option. Finally, the `--trailerFormat` is processed. All three sections have available the global archive information. For each revision, the fields designated as per-revision are shown.

- `--format=value`

  defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  This option and the `--fields` option are mutually exclusive.

  `--format` options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). The `--fields` option automatically adds a newline on the end, but you must supply the newline for formats with a `\n`, for example:
  `si rlog --format="{membername},{revision}\n"`
- `--headerFormat=value`

  defines the header to be used.
- `--noFormat`

  disables formatting for user-formatted text. If `--noFormat` is specified, the per-revision iteration is not performed.
- `--noHeaderFormat`

  disables the header formatting.
- `--noTrailerFormat`

  disables the trailer formatting.
- `--range=value`

  specifies a range of revision numbers. The format of `value` for a single revision is simply the revision number itself. Separate multiple non sequential revisions with a comma, and use a dash to indicate a sequential range, for example, `1.2-1.5`.

  You may also prefix or append a dash to a single revision number, meaning you want to show from 1.1 to the specified revision, or from the specified revision to the tip. For example, you would specify `-1.6` to show revisions 1.1 to 1.6. Or you could specify `1.2-` to show revisions 1.2 to the tip revision, whatever it may be.

  Another alternative is to use *`?locker[:user]`* to match revisions that are locked, or locked by a certain user. For example,
  `si rlog --noHeaderFormat --noTrailerFormat --range=?locker --format="{membername},{locker},{revision}\n"`

  displays one line per locked revision, containing the membername, locker, and the revision locked.

  The revisions are printed in order from the highest revision to the lowest.
- `--trailerFormat=value`

  defines the trailer to be used.
- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format *`field1[:width1],field2[:width2]...`*. Specifying the column *`[:width]`* (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a `space`.

  This option and the `--format` option are mutually exclusive.

  The fields available for printing can be one or more of the following:

  - `added`

    per revision: the number of lines/bytes added for the current revision.
  - `alllabels`

    global: all labels on any revision of the member, in the format *`label:revision[,label:revision]...`*
  - `archivedescription`

    global: the archive description.
  - `archivename`

    global: the name of the archive to which the member refers.
  - `archiveshared`

    global: indicators for members that share another member's archive; 1 if shared, 0 if not shared.
  - `attributes`

    global: member attributes in the form *`attr=value[,attr=value]`*.
  - `author`

    per revision: author of the revision.
  - `branchcount`

    global: number of branches.
  - `branchid`

    global: default branch.
  - `branchtiprev`

    global: tip of the branch on which the member revision is located.
  - `cpid`

    per revision: the associated change package identifier (applies only when change packages are enabled).
  - `cpsummary`

    per revision: the associated change package summary (applies only when change packages are enabled).
  - `date`

    per revision: date on the revision.
  - `deleted`

per revision: the number of lines/bytes deleted on the current revision.

- `description`

  per revision: the revision description.
- `exclusivelockmandatory`

  global: if exclusive lock mandatory; 1 if exclusive lock is mandatory, 0 if exclusive lock is not mandatory.
- `format`

  global: the storage format; 1 if stored as text, 0 if stored in binary format.
- `formattedlabels`

  global: all labels formatted into 80 column lines.
- `frozen`

  global: indicators for frozen members; 1 if frozen, or 0 if not frozen.
- `hasexclusivelocker`

  global: indicators for exclusively locked members, 1 if has exclusive locker, 0 if not.
- `haslabels`

  global: 1 if alllabels is non-null, 0 if null.
- `haslocker`

  global: 1 if has one or more lockers, 0 if null.

  per revision: 1 if labels is non-null, 0 if null.
- `hasnewrevdelta`

  global: displays 1 if the member revision is not on a tip.
- `haspackage`

  per revision: displays 1 if `cpid` is set, indicating the presence of a change package.
- `headrev`

  global: displays the head revision number.
- `isreference`

  global: displays 1 if the member belongs to a shared or moved subproject, 0 if the member belongs to a subproject that is not shared or moved.
- `issandbox`

  global: displays 1 if it is a Sandbox, 0 if a project.
- `labels`

  per revision: a comma separated list of labels on this revision.
- `lockrecord`

  per revision: lock information for each lock. By default, the locker and lock type display in a comma separated list. You can customize the lock information that displays by using the `--lockRecordFormat` option. You can customize the separator between locks by using the `--lockseparator` option.
- `memberdevbranch`

  global: displays the member's development branch.
- `membername`

  global: displays the name of the member.
- `memberrev`

  global: displays the member revision number.
- `memberrule`

  global: displays the member rule on one line (in the header, if it exists).
- `projectname`

  global: from a project, displays the name of the current project or subproject that the member belongs to. From a Sandbox, displays the name of the Sandbox or sub Sandbox that the member belongs to.
- `projecttype`

  global: displays 1 if the project is a regular project, 2 if the project is a build project, 3 if the project is a variant project.
- `rawmemberrule`

  global: displays the member rule using the CLI syntax (in the header section, if it exists).
- `referenceproject`

  global: project references.
- `relativemembername`

  global: displays the member name relative to the top level project.
- `revision`

  per revision: revision number.
- `revisioncount`

  global: displays the count of the revisions.
- `revisionsonbranchcount`

  global: the number of revisions on branches.
- `state`

  per revision: displays the state.
- `storageformat`

  global: displays 1 if stored by reference, 0 if stored by delta.
- `trunktip`

  global: displays the trunk tip revision number.
- `type`

  global: always 0 to indicate a normal file member.
- `workingarchive`

  global: displays the name of the archive from which the working file is derived .
- `workingfile`

  global: displays the working file name; *`null`* in a project.
- `workingfileexists`

  global: displays 1 if a Sandbox and working file exists, 0 if operating on a project.
- `workingfilerevunknown`

  global: displays 1 in a Sandbox if the working revision is unknown, or 0 if known. Always displays 0 in a project.
- `workingrevdelta`

global: displays 1 in a Sandbox if the working revision is not the same as the member revision, or displays 0 in a project.

- `workingfilewritable`

  global: displays 1 if a Sandbox and working file are writable.

- `workingrev`

  global: displays the working revision.

- `--rfilter=`*`filteroptions`*

  allows you to display any revisions of the current member that meet the selection criteria specified in the *`filteroptions`*. Comma separated expressions are combined using an OR operator. Multiple occurrences of an option are combined using an AND operator. A leading ! negates any simple filter that follows. For example,

  `--rfilter=labeled:Release 3.0,branchboundaries --rfilter=!label:obsolete`

  displays all revisions with a label of Release 3.0 or a revision at a branch boundary and whose label is not Obsolete. The *`filteroptions`* can be one or more of the following:

  - `range:`*`low-high`*

    selects revisions that are in the specified revision number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the range keyword must be repeated, for example

    `--rfilter=range:1.100-1.200,range:1:400-1:500`

  - `branchrange`*`:low-high`*

    selects branched revisions that are in the specified revision number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the branchrange keyword must be repeated, for example

    `--rfilter=branchrange:1.2.1.1-1.2.1.5,branchrange:1.4.1.1-1.4.1.5`

  - `daterange:`*`<date>`*`-`*`<date>`*`|:past:`*`<nb>`*`:years|months|days|hours`

    selects revisions that were created during a specified time range or during a specified amount of time in the past. *`<date>`* specifies a date in any of the local date/time formats. For more information on date/time formats, see the options reference page. *`<nb>`* is used to specify the number of units to include in the range for a specified amount of time in the past.

    Note

    If the date/time format contains either "-" or "," you must quote each date. For example: `--rfilter=daterange:"June 21, 2009 - 6:07:28 PM"-"June 25, 2009 - 6:07:28 PM"`

    When in a shell, you must use two kinds of quotes so that the shell does not remove them. For example:

    `--rfilter=daterange:'"June 21, 2009 - 6:07:28 PM"'-'"June 25, 2009 - 6:07:28 PM"'`

    or

    `--rfilter=daterange:"'June 21, 2009 - 6:07:28 PM'"-"'June 25, 2009 - 6:07:28 PM'"`

  - `branch[::`*`current`*`|:`*`name`*`]`

    selects revisions that are on the specified branch.

  - `labeled`*`:name`*

    selects revisions with labels. If a label name is specified, only revisions with that label are displayed. If no label name is specified, all labeled revisions are displayed.

  - `labellike`*`:pattern`*

    selects revisions that match the specified pattern. Your administrator determines whether `glob` or `regex` patterns are used for matching.

  - `locked[::`*`me`*`|`*`::anyone`*`|`*`userName`*`]`

    selects revisions that are locked. If `me` or a user name is specified, only revisions locked by that user are displayed. If no user or *`anyone`* is specified, all locked revisions are displayed.

  - `locktype[:`*`exclusive`*`|`*`:nonexclusive`*`|`*`:any`*`]`

    selects revisions that are locked with the specified lock type. If no lock type or *`any`* is specified, all locked revisions are displayed.

  - `state`*`:name`*

    selects revisions that are at the specified state.

  - `author[::`*`me`*`|`*`:userName`*`]`

    selects revisions created by the specified author.

  - `pending`

    selects revisions that are pending.

  - `anyspecial`

    selects working revisions.

  - `branchboundaries`

    selects revisions that are branched, or at the base or tip of a branch.

- `--lockRecordFormat=`*`value`*defines the format for displaying lock information in the `--lockrecord` field. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:

  - `{revision}`

    displays the revision that is locked.

  - `{locker}`

    displays the user who locked the revision.

  - `{locktype}`

    displays the type of lock on the revision (exclusive or non-exclusive).

  - `{locktimestamp}`

    displays the time when the revision was locked.

  - `{lockcpid}`

    displays the change package associated with the lock on the revision.

  - `{project}`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.

  - `{devpath}`

    displays the name of the development path where the lock on the revision was made from.

  - `{sandbox}`

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.

  - `{hostname}`

    displays the hostname of the computer that locked the the revision.

  - `{hascpid}`

    displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.

- `{hassandbox}`

  displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
- `{hasdevpath}`

  displays 1 if the lock was made from a development path, 0 if it wasn't.
- `{member}`

  displays the name of the locked revision.

- `--lockseparator=`*`value`*

  defines the separator used between each lock displayed in the `--lockrecord` field.
- `--maxTrunkRevs=`*`value`*

  specifies the maximum number of revisions along the mainline project to display. If you limit the number of revisions to be displayed, and the maximum is reached, a trailing message displays at the bottom of the list. The maximum number of revisions is based on the number of visible mainline revisions; branched revisions are not included in the count.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.
- *`member...`*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si diff, si memberinfo, si mods, si print, si projectinfo, si report, si revisioninfo, si sandboxinfo, si viewhistory
- Miscellaneous: ACL, diagnostics, options, preferences

si sandboxes

lists Sandboxes on the client

Synopsis

```
si sandboxes [--[no]displaySubs] [--height=value] [--manage] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [-F|--selectionFile=value] [--status=[none|gui|default]]
```

Description

`si sandboxes` displays Sandboxes currently registered on the Integrity Client and the corresponding project, server name, and port number, for example:

```
c:\demoapp\project.pj -> /projects/demoapp/project.pj
(server.intra-wif:7001)
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]displaySubs`

  controls whether to display sub Sandboxes.
- `--manage`

  enables the Sandbox manager view for those using the `-g` or `--gui` options.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or `si viewprefs`, you are able to set or view the preference keys for this command.

See Also

- Commands: si createsandbox, si dropsandbox, si importsandbox, si sandboxinfo, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si sandboxinfo

lists information about a Sandbox

Synopsis

```
si sandboxinfo [--[no]attributes] [(-S sandbox|--sandbox=sandbox)] [--[no]failOnAmbiguousProject] [--[no]showCheckpointDescription]
[--[no]associatedIssues] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--
yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [-F|--
selectionFile=value] [--status=[none|gui|default]]
```

Description

`si sandboxinfo` displays current information about a Sandbox, for example:

```
Variant Sandbox Name: c:\SourceCode\project.pj
Project Name: /master_projects/SourceCode/project.pj
Repository Location: /master_projects/SourceCode/project.pj
Configuration Path: #/master_projects#SourceCode#d=Milestone1
Server: intra-wif:7001
Development Path Information:
    Development Path: Milestone1
    Extendable Development Path
    On Live Configuration: Retain Live Configuration
    On Existing Development Path: Share Development Path
Restricted: false
Last Checkpoint: 1.4
Last Checkpoint Date: May 1, 2014 - 3:28 PM
Members: 0
Subsandboxes: 3
Line Terminator: native
Computes Checksums of Working Files
Scope: is with attribute Beta
Project Description:
Checkpoint Description: Service Pack 1
Project Attributes: none
Sandbox Attributes: none
Associated Issues: 78484: Feature Request - Add calculator function
Labels:
InceptionPhase: 2014-05-02 - The Feature Request is in the inception phase.
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]attributes`

  controls whether to display project and Sandbox attributes.
- `--[no]showCheckpointDescription`

  controls whether to display the checkpoint description of the master project.
- `--[no]associatedIssues`

  controls whether to display information for any Integrity items that are associated with the project, including labels, label dates, and label summaries. Only item types that you have permission to view are displayed.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si ci, si configuresandbox, si createsandbox,si memberinfo, si mods, si projectinfo, si rlog, si sandboxes, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si serveralerts

displays Integrity Server alert messages for all currently connected servers

Synopsis

```
si serveralerts [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]
```

Description

`si serveralerts` displays Integrity Srver alert messages for all servers that you are currently connected to. The alert message displays who sent the message, the server it came from, when it was sent, and the message. If you are not connected to any servers, a message informs you that there are no alert messages. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

Note

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

See Also

- Commands: si viewserveralert, si admingui, si servers
- Miscellaneous: options

si servers

displays the current connections to an Integrity Server

Synopsis

`si servers [--[no]showVersion] [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist]] [--quiet] [-F|--selectionFile=value] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`si servers` displays active server connections in the format user@host_name:port.

The default server connection is indicated by user@host_name:port(default).

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showVersion` ontrols whether to show build version information for the connected server. The presentation of this information is in the format `[Build: 2015]`.
- `--[no]persist` controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:si connect, si disconnect
- Miscellaneous:diagnostics, options, preferences

si setmemberrule

sets the member rule for one or more members

Synopsis

```
si setmemberrule [--clear] [--[no]confirm] [--[no]expand] [-f] [--[no|confirm]overrideRule] [(-r rev|--revision=rev)] [(-R|--[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

A member rule is a revision--typically a symbolic revision--attached to a member that you can use regularly to update the member's revision or use it with any command that allows you to specify the member rule as a pre-defined revision. In the graphical user interface, only the Check Out and Update Member Revision commands allow you to specify a member rule. In the command line interface, any command that accepts the `-r|--revision` option allows you to specify a member rule. For example, you can check out a revision corresponding to the member rule in the graphical user interface or add a label to a revision corresponding to the member rule in the command line interface. After you create a rule for a member, you can also view the rule using the `si memberinfo` command.

Caution

By design, applying a rule as a member revision to a member does not dynamically update the member revision of the corresponding member in an external project configuration (normal, variant, build). For example, if member 1's member revision is updated in project A, the corresponding member in a variant of project A with the rule configured to link to project A is not updated with the same member revision. To update the corresponding member in the variant according to the member rule, you must use the `si updaterevision` command with the member rule, or your administrator can configure the `ClientLink.js` sample event trigger script that enables dynamic updating of linked member revisions under certain conditions. For more information, contact your administrator.

Example:

1. A project administrator defines a rule based on a label and implements it:

   ```
   si setmemberrule -rReadyForUse demoapp.c demoapp.h
   ```

   ```
   si updaterevision -r:rule demoapp.c demoapp.h
   ```

   ```
   si freeze demoapp.c demoapp.h
   ```

2. Developers work as usual, using

   `:member`.

   Note
   In this scenario, it is not expected that members with the rule are modified locally.

   ---

3. From time to time, the project administrator re-applies the rule, based on new revisions marked *ReadyForUse*:

   ```
   si thaw demoapp.c demoapp.h
   ```

   ```
   si updaterevision -r:rule demoapp.c demoapp.h
   ```

   ```
   si freeze demoapp.c demoapp.h
   ```

   Several rule filters are available so you can easily work with members that contain rules. For more information, see the `--filter` option in the options reference page.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--clear`

  clears the member rule and removes it from the specified members.
- `--[no]confirm`

  causes Integrity to confirm that the rule will be cleared.
- `--[no]expand`

  controls whether to expand the revision before setting the rule.
- `-f`

  forces removal of the rule without confirmation.
- `--[no|confirm]overrideRule`

  controls whether to override the existing member rule.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si ci, si co, si edit, si resync, si revert, si updatearchive, si updaterevision
- Miscellaneous:

  ACL, diagnostics, options, preferences

si setpolicysection

sets policy settings for a new or existing policy section (global or project)

Synopsis

`si setpolicysection [--[no|confirm]createPolicySection] [--policy=policy key[;attribute]=value...] [--resetPolicy=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] policy section name...`

Description

`si setpolicysection` sets policy settings for a new or existing policy section (global or project).

A *policy* is a collection of configuration management statements. Integrity uses policies to establish a shared working environment and to provide cross-platform compatibility. Integrity references these policies whenever a user invokes a configuration management operation. Because policies establish sitewide and project-wide standards for Integrity, they should be maintained by the site administrator.

You can set policies at the global and project levels. By modifying policy options on the Integrity Server, you can configure configuration management functionality across an entire site. Global policies apply for all projects hosted on the Integrity Server. Project policies apply to individual target projects and establish the way Integrity operates when dealing with those projects.

Note

- The `ViewPolicy` and `EditPolicy` permissions are required.
- By using `si setpolicysection` and the other policy commands with scripts, you can automate the setup of configuration management policies. To manage your configuration management policies, PTC recommends using the Integrity Administration Client.

Policy Syntax

Each policy uses the following syntax:

```
policyname=value
```

and you can set attributes in the following syntax:

```
policyname;attribute=value
```

Note

A semi-colon is used in formatting attributes.

Currently, there is only one policy attribute available: `locked`. The locked attribute stops a more precise policy from overriding the locked policy and also stops a client from manually overriding that policy.

For example, at some sites it is desirable to set configuration management rules that apply to everyone on the system. This might be the case in a multi-user, networked environment, where you may decide that a strict-locking policy should be in effect for all users.

Contents of a Policy

A policy has multiple sections. The first section is the set of global or default policies that apply server-wide.

Any created project policies use the format */path/project file name*`.pj`, for example, `/src/teama/project.pj`. These are project-specific policy overrides where each project policy affects all nested subprojects of that project unless the project policy itself is specifically overridden by a lower-level subproject.

The following example sets a global policy to ignore missing archive descriptions except for the sample project `/src/teama/project.pj`, where `ArchiveDescriptionRequired` is set and cannot be changed:

```
ArchiveDescriptionRequired=false

[/src/teama/project.pj]
ArchiveDescriptionRequired=true
ArchiveDescriptionRequired;locked=true
```

Note

When working with shared subprojects, Integrity uses the actual name of the subproject in the repository rather than its relative name in the project hierarchy.

Setting Policies for a Development Path

You can also manually create a section that allows for policy overrides within the context of a development path or variant project. This type of policy override uses the format */<path>/<project file name>*`.pj;`*<development path name>*, for example, `[/src/teama/project.pj;Release_1a]`. Development path policy overrides can be entered at the global level or at the level of a project or subproject.

Caution

Global policy sections for development paths are only consulted if the configuration path of the object being queried corresponds to a top level variant project. For example, if you have a configuration where a subproject is on `devpath2` and the top level project is on the mainline (or another development path), then the policy section `[;devpath2]` is not consulted when establishing the policies for the subproject, even though the subproject is on `devpath2`.

The following shows the syntax that sets a global policy to enable the workflow and document integration for all variant projects and subprojects under the `Release_1a` development path:

```
[;Release_1a]
IntegrityManagerEnabled=true
```

In this example, change packages are explicitly disabled for the build project under the `Release_1a` development path:

```
[/src/build/project.pj;Release_1a]
ChangePackagesEnabled=false
```

Syntax for Policy Options

This section describes the syntax for available policy options.

For detailed descriptions of policies and policy options, see "Server Policies for SCM" in the *PTC Integrity Server Administration Guide* or the Integrity online help.

General Policies:

- `RevisionDescriptionRequired=true|false`
- `ArchiveDescriptionRequired=true|false`
- `StoreTextByReference=true|false`
- `DeferredOperationsMandatory=true|false`
- `RestrictNewRevDeltaToVariant=true|false`

- CheckMemberPermissions=true|false
- MemberAlwaysWriteable=true|false
- SymbolicLinksEnabled=true|false
- RevisionChecksumsEnabled=true|false
- StoreSBRRevisionsToVault=true|false
- DisableVisualDiffAndMergeForStoreByReferenceMembers=true|false
- TextWorkingFileSizeGovernor=0
- IncludeFilter=file:pattern
- ExcludeFilter=file:pattern
- BinaryArchivePatterns=.file_pattern

Change Package Policies:

- ChangePackagesEnabled=true|false
- UseChangePackageTrackingLabels=true|false
- ChangePackagesMandatory=true|false
- ChangePackagesTransactional=true|false
- ChangePackageReviewEnabled=true|false
- IntegrityManagerEnabled=true|false
- IntegrityManagerIssueMandatory=true|false
- ChangePackageReviewRule=*UserGroupList*
- ChangePackageWatcherRule=*UserGroupList*
- ChangePackageDescriptionTemplateEnabled=true|false
- ChangePackageDescriptionTemplate=template_text
- si.cp.issues.fields=*ListofFields*

Lock Policies:

- TextArchiveLockingPolicy=nonexclusive|exclusive|none
- BinaryArchiveLockingPolicy=nonexclusive|exclusive|none

States Policy

States=*SpaceDelimitedList*

Keywords Policy

Keywords=*ListofKeywords*

IgnoreKeywords=true|false

Line Terminator Policies:

- PreserveLineTerminatorsInTextArchives=true|false
- si.lineterminators=crlf|lf|native

Integrity Client Default Policies:

Server-side command and view policies set client-side command and view preferences. See the si setprefs command for details.

Additional Policies:

Additional policies are policies that are not rendered graphically, certain custom or workflow and document (IM) policies for maintaining backwards compatibility, for setting client-side triggers, or permission inheritance policies for development path ACLs.

other=*other*

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --[no|confirm]createPolicySection

  specifies whether or not to create a new policy section, if the specified policy section does not exist.
- --policy=*=policy key[;attribute]=value*

  specifies the policy setting name (policy key), policy attribute (if applicable), and policy setting option (value) for the policy section. To set multiple policies, specify this option for each policy.

  Note

  Policy keys are case-sensitive.
- --resetPolicy=*value*

  specifies to reset the specified policy option in the specified policy section to the default. This deletes the specified policy from the specified policy section (as well as the lock or append attributes from the policy, if set).

  Note

  - This option is mutually exclusive with the --policy option. This means that you cannot set one policy and reset another policy in the same command.
  - To reset multiple policies, use this option for each policy.
  - Resetting each policy in a policy section is equivalent to deleting the policy section.

  For example, to reset the value of the IntegrityManagerEnabled policy and unlock it in the global policy section, type:

  ```
  si setpolicysection --resetPolicy=IntegrityManagerEnabled :global
  ```
- *policy section name...*

  specifies the policy section to set, where *policy section name* is of the form *project,devpath*. For example:

  - :global for a global policy.
  - <*project*> for a project, for example, /demo/project.pj.
  - <*project,development path*> for a development path on a project, for example, /demo/project.pj;Variant1.
  - *development path* for all projects that are configured on the development path at the top level, for example, ;Variant1.

  Note

  You may need to escape the ; in your command line environment, for example, enclose it in "" or escape the individual character with a /.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si copypolicysection, si deletepolicysection, si setpolicysection si viewpolicysection, si viewpolicysections
- Miscellaneous: diagnostics, options, preferences

si setprefs

sets preferences

Synopsis

```
si setprefs [--command=value] [--[no]resetToDefault] [--[no]save] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-F
file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] pref[=value]...
```

Description

`si setprefs` sets preferences and configuration options. These settings are used to determine default behaviors for other commands - each option on each command has a preference key associated with it. The si viewprefs command lists the commands and preference keys. Changes to your preferences are either for the current client session (until si exit is used) or may be permanently saved in your system's `home` directory, into a file named `Integrity`, using the `--save` option.

To permanently enable keyword expansion when resynchronizing members, for example, you would specify:

```
si setprefs --command=resync --save keywordExpand=expand
```

Your administrator can also lock certain preferences from the Integrity Server, preventing you from configuring them using the si setprefs command. Preferences that are locked -- viewable with si viewprefs -- display (locked) at the end of the output line. The following preferences can be locked from the server by editing Integrity policies in the Administration Client:

Note

Tip: For information on editing Integrity policies, see the *PTC Integrity Server Administration Guide*.

Preference	Affected Commands
`branchIfVariant`	si ci
`breakLock`	si unlock
`changePackageID`	si co, si lock
`moveLock`	si co
`onExistingArchive`	si add
`restoreTimestamp`	si resync, si revert
`retainWorkingFile`	si add, si ci
`saveTimestamp`	si add, si ci
`sparse`	si importsandbox
`updateMemberRev`	si ci

Caution

Do not edit the `IntegrityClient.rc` file manually, because preferences that appear more than once in the `IntegrityClient.rc` file can cause Integrity to behave unpredictably.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--command=`*`value`*

  identifies the command to be set. For an easy way to see a list of commands and values that may be set, simply type the si viewprefs command, either piped through `|more` or redirected to a file, for example:
  ```
  si viewprefs --global --showValidValues >prefs.txt
  ```
  The commands and preference keys are also listed on the preferences reference page.

- `--[no]resetToDefault`

  controls whether to revert specified settings to the default values as shipped with the Integrity Client. If specifying `--resetToDefault`, you must not specify *`=value`* for each preference.

- `--[no]save`

  controls whether changes should be permanently saved.

- `--[no]ask`

  controls prompts to the user for specific preferences. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:

  - `once`

    asks the user the first time only, and then uses the provided value every time after for the duration of that command.
  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).
  - `element-last`

    asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.

  For example, to set the server host for si connect to a specific host name, you specify something like:

  ```
  si setprefs --command=connect
  server.hostname=specific.hostname.com
  ```

  but to set the preference to ask for a host name when using si connect, you specify something like:

  ```
  si setprefs --command=connect
  --ask   server.hostname=element-last
  ```

- `--ui=[unspecified|gui|cli|api]`

  controls whether to apply the preference to the graphical user interface, the command line interface, or when the interface is unspecified. By default, `--ui=`*`cli`* is implied when issuing the `si setprefs`. To set preferences for GUI behavior, however, you should specify `--ui=`*`gui`*. For example, to set the *`manage`* preference to be true in the GUI for the si sandboxes command, you would type:
  ```
  si setprefs --command=sandboxes --ui=gui manage=true
  ```
  These correlate to settings in the `IntegrityClient.rc` file, which can be seen as having the *`gui.si.`* or *`cli.si.`* prefix, or simply the *`si.`* prefix when it is unspecified.

- *`pref[=value]...`*

identifies the preference string. If you specified the `--resetToDefault` option, then you only need to specify the preference name; otherwise specify a value for the preference. Use spaces to specify multiple preferences.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si loadrc, si viewprefs
- Miscellaneous: diagnostics, options, preferences

si setprojectdescription

sets the project description

Synopsis

```
si setprojectdescription [--description=desc] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--
[no]failOnAmbiguousProject] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)]
[(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--
settingsUI=[gui|default]] [-F|--selectionFile=value] [--status=[none|gui|default]]
```

Description

`si setprojectdescription` sets the one line description of a project. For example:

```
si setprojectdescription --project=c:/Aurora_Program/bin/Libra/project.pj --description="Requirements for product component HFD3-3433"
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--description=`*desc*

  specifies the new description text to be set as the project description.

Note

Descriptions that include spaces must be enclosed by quotes.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addprojectlabel, si appendrevdesc, si checkpoint, si projectinfo, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si setproperty

sets configuration management properties stored in the database

Synopsis

`si setproperty [--comment=value] [--restoreDefault] [--value=value] [-?|--usage] [-N|--no] [-Y|--yes] [--hostname=server] [--port=number] [--user=name] [--password=password] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [-Ffile|--selectionFile=file] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...`

Description

`si setproperty` sets configuration management properties stored in the database. Properties specify information that affects the operation of the Integrity Server, workflows and documents, and configuration management. Other properties are stored and configured in properties files on the server's file system. For a complete list of configurable properties and possible values, see the *PTC Integrity Server Administration Guide*.

Note

- Some properties require the server to be restarted for the changes to take effect.
- Access to configuring properties is based on permissions. An administrator with the `AdminServer` or `DebugServer` permission for workflows and documents can edit workflow and document properties, an administrator with the `AdminServer` or `DebugServer` permission for configuration management can edit configuration management properties, and an administrator with the Integrity Server `AdminServer` or `DebugServer` permission can edit all properties.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--comment=value`

  specifies a comment associated with the change made to the property. While PTC recommends specifying a comment for troubleshooting purposes, a comment is optional.
- `--restoreDefault`

  restores the property to the default value.
- `--value=value`

  specifies the new value of the property.
- *`string...`*

  specifies the name of the property you want to configure.

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: si admingui, integrity gui
- Miscellaneous: diagnostics, options, preferences

si sharesubproject

shares an existing subproject between multiple projects

Synopsis

si sharesubproject [--sharedProject=*project location*] [--[no]preserveCase] [-r *subproject revision*|--subprojectRevision=*subproject revision*] [--subprojectDevelopmentPath=*subproject development path name*|--variant=*subproject development path name*] [--type=[normal|variant|build|default]] [--[no|confirm]closeCP] [--cpid|--changePackageId=*ID*] [--issueID=*value*] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--devpath=*development path name*] [--hostname=*server*] [--port=*number*] [--password =*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *subproject location*

Description

A shared subproject is a subproject that is a member of more than one project. Integrity allows you to share a subproject between two or more projects by referencing the original subproject. A shared subproject allows you to access common members across many projects. Shared subprojects are not required to be located within the same directory structure or project hierarchy.

A shared subproject functions the same as an unshared subproject and is accessible by the same commands.

Shared subprojects that were created in a previous version of Integrity (formerly MKS Source or Source Integrity Standard Edition) are detected as they are accessed by Integrity without disrupting the operation. The format of these subprojects is retained until you change or update it to the new format by re-configuring it.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --sharedProject=*project location*

  specifies the path and name of the subproject you want to share. For details on how to specify a project, see the -P option on the the options reference page.
- --[no]preserveCase

  uses the same case that exists on the file system when specifying a destination directory in a Sandbox. By default, the destination directory uses the same case.
- --[no|confirm]closeCP

  closes the change package after command completion.
- -r*subproject revision*
- --subprojectRevision=*subproject revision*
- specifies the checkpoint number (for build subprojects). For example, 1.5. This option is used with --type=build.

Note
This option cannot be used if you specify a project using a keyword string for the --sharedProject option. This option is also mutually exclusive with the --subprojectDevelopmentPath or --variant options.

- --subprojectDevelopmentPath=*subproject development path name*
- --variant=*subproject development path name*identifies the development path of a variant subproject. This is a label that was associated with a branch of the subproject by si createdevpath. Paths that include spaces must be enclosed by quotes. The following characters may not be used in a development path: \n, \r, \t, :, ['], #. This option is used with --type=*variant*

Note
This option cannot be used if you specify a project using a keyword string for the --sharedProject option. It is also mutually exclusive with the --subprojectRevision option.

- --type=*[normal|variant|build|default]*

  specifies the new configuration type for the subproject. If you specify the --sharedProject option using a flat string, by default the subproject type is the same as the parent project type. If you specify the --sharedProject option using a keyword-based string, by default the subproject type is determined by the specified configuration. Use this option to specify a configuration type other than the default.

  --type=normal configures the subproject to the master (non-variant) version of the subproject.

  --type=variant configures the subproject based upon a specific development path. The option is used with--variant=*subproject development path name* or --subprojectDevelopmentPath=*subproject development path name*.

  --type=build configures the subproject as a static subproject based upon a specific checkpoint of the master project that is used for building or testing the project, but not for further development. This option is used with -r *subproject revision* or --subprojectRevision=*subproject revision*.

  --type=default configures the subproject to be the same as the parent project that you are adding the subproject to. For example, if you add a subproject to a normal project, the subproject is added as a normal type. For information on what the default type is, see your administrator.
- *subproject location*

  specifies where in the project you want the shared subproject to appear, for example, c:/Aurora_Program/bin/Libra/.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si addsubproject, si checkpoint, si configuresubproject, si createproject, si createsandbox, si createsubproject, si movesubproject, si drop, si dropproject, si projectinfo, si projects, si viewproject
- Miscellaneous: ACL, diagnostics, options, preferences

si snapshot

records the state of a sandbox.

Synopsis

`si snapshot [--targetDevpath=`*`value`*`] [--author=`*`value`*`] [-d|--description=`*`value`*`] [--descriptionFile=`*`snapshot`*`] [-L|--label=`*`value`*`] [--[no]notify] [--[no]failOnAmbiguousProject] [-s|--state=`*`value`*`] [--[no]stateMembers] [-S|--sandbox=`*`value`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`si snapshot` records a capture of the current state of a Sandbox, where each element in the Sandbox can be identified with a pre-existing entity in the repository on the Integrity Server. The Sandbox snapshot creates a project checkpoint that you can create a build Sandbox or a development path from. The revision number of a checkpoint created by a snapshot includes the revision number of the last checkpoint. For example, if the head revision of the project is 1.1, then the project checkpoint created by the snapshot will be 1.1.1.1. The Sandbox snapshot displays as a branched project checkpoint in the project history.

Note

`si snapshot` records the state of a sandbox, while `si checkpoint` records the state of a project. Checkpointing is recommended for recording milestones during the development of a project; however, there may be situations where you may need to take a snapshot of a sandbox to record the state for later use.

The state of a sandbox is defined by its set of elements, which include the following:

- sandbox members identified with an archive and working revision from which the working file was created
- former members that were dropped, but are still present in your sandbox
- sub Sandboxes, identified by project name and type
- former sub Sandboxes that were dropped but are still present in your Sandbox

Note the following about how `si snapshot` handles the set of Sandbox elements:

- To be included in the snapshot, all working files must be checked in. There must be no working file changes in the Sandbox.
- If the working file revision differs from the member revision, it is the working file revision that is included in the snapshot.
- Members with no working files are not included in the snapshot.
- Former members that still have working files in the Sandbox directory appear as members in the snapshot.
- Former subprojects that are still in the Sandbox directory appear as restored subprojects in the snapshot.

For more information on Sandbox snapshots, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--targetDevpath=`*`value`*

  specifies the development path to create the snapshot from. With this option you can only specify an existing development path.
- `--author=`*`value`*

  specifies the author of the snapshot revision.
- `-d`
- `--description value`

  specifies the description for the snapshot revision.
- `--descriptionFile=`*`value`*

  specifies a file containing the description for the snapshot revision.
- `-L`
- `--label=value`

  specifies a label for the snapshot revision. Note the following about using labels:
  - Labels cannot contain colons(:), square brackets ([ ]), or leading spaces.
  - Labels cannot have the same format as a valid revision number.
  - Labels that include spaces must be enclosed by quotes.
  - PTC recommends not using hyphens (-) in labels. Using hyphens may cause some *si* commands to fail.
- `--[no]notify`

  provides a notification when the snapshot has been created.
- `-s`
- `--state=`*`value`*

  specifies the state for the project checkpoint created by the snapshot.
- `--[no]stateMembers`
- specifies whether to apply the state to all members or only the project.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewproject, si viewprojecthistory
- Miscellaneous:

  ACL, diagnostics, options, preferences

si submit

submits a deferred operation

Synopsis

```
si submit [(--cpid|--changePackageId=ID)] [--issueId=ID] [--[no|confirm]closeCP] [-f] [--[no]overrideCPID] [(-R|--[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--filter=filteroptions] [--[no|confirm]recurse] [(-S sandbox|--sandbox=sandbox)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [-Fvalue] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [-F|--selectionFile=value] sandbox member...
```

Description

`si submit` submits a deferred operation in your Sandbox. Submitting the operation completes the command and makes it visible in the associated project. If reviews are mandatory, submitting a deferred operation creates a pending entry in the associated change package.

Note

If strict locking is enabled, you cannot submit a member with a deferred check in operation if you do not have a lock on the member.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--cpid=`*ID*
- `--changePackageId=`*ID*

  identifies a change package that is notified of this action, for example, `145:2`. By default, the change package that was specified during the deferred operation is used. You can override this change package using the `--overrideCPID` option.
- `--[no|confirm]closeCP`

  controls whether to close the associated change package.
  - `--nocloseCP` means do not close the change package.
  - `--confirmcloseCP` means ask before closing the change package.
  - `--closeCP` always closes the change package.
- `-f`

  force all elements to be submitted with the specified change package ID override.
- `--issueId=`*ID*

  identifies an Integrity issue that is notified of this action. This option is another way of redirecting to a change package. If you have an issue assigned to you that contains `one` open change package, you can specify the issue ID instead of the change package ID. This option can only be specified if you are using Integrity.
- `--[no]overrideCPID`

  force all elements to be submitted with the specified change package ID override.
- *sandbox member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si drop, si move, si rename
- Miscellaneous: ACL, diagnostics, options, preferences

si submitcp

submits a change package.

Synopsis

`si submitcp [--[no|confirm]closeCP] [--[no]showSubmitSuccessful] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] [--[no|confirm]commit] [--[no|confirm]deferredIsError] [--[no|confirm]ignoreNoDeferred] `*`issue|issue:change package id...`*

Description

`si submitcp` submits the uncommitted operations in a change package. For example:

```
si submitcp 2:34
```

Although you can submit operations individually using `si submit`, you can ensure that no operations are missed by submitting a change package that contains deferred operations. Deferred operations are visible only from the Integrity Client, and are not committed to the project or repository until they are submitted.

Note

If strict locking is enabled, you cannot submit a change package that contains deferred check in or work in progress entries that do not have corresponding locks.

When reviews are mandatory, submitting a change package begins the review process. For more information, see the *PTC Integrity User Guide*.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]closeCP`

  specifies whether to close the change package after it has been submitted.
- `--[no]showSubmitSuccessful`

  specifies to display a confirmation dialog box indicating the successful submission of a change package. This option is enabled by default.
- `--[no|confirm]commit`

  specifies whether to commit deferred or pending changes to the server repository, or submit them for review. The default is to commit without confirmation.
- `--[no|confirm]deferredIsError`

  specifies whether to proceed with the submit operation if there are deferred change package entries.
- `--[no|confirm]ignoreNoDeferred`

  specifies whether to proceed with the submit operation if there are no deferred, lock, or work in progress change package entries.
- *`issue...`*
- *`issue:change package id...`*

  `issue` identifies a specific issue that contains all open change packages that you want to close; use spaces to specify more than one issue.

  `issue:change package id` identifies a specific change package to close; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si acceptcp, si rejectcp, si closecp, si add, si ci, si co, si cpissues, si createcp, si drop, si viewcps, si viewcp.
- Miscellaneous:

  ACL, diagnostics, options, preferences

si thaw

thaws a project member

Synopsis

`si thaw [(-R|--[no|confirm]recurse)] [--filter=`*`filteroptions`*`] [(-P `*`project`*`|--project=`*`project`*`)] [--[no]failOnAmbiguousProject] [(-S `*`sandbox`*`|--sandbox=`*`sandbox`*`)] [--devpath=`*`path`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`member...`*

Description

`si thaw` thaws project members previously frozen with the si freeze command. Once thawed, all operations normally associated with a member can be performed again.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- *member* identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si checkpoint, si freeze
- Miscellaneous: ACL, diagnostics, options, preferences

si unlock

unlocks a member

Synopsis

```
si unlock [--[no|confirm]breakLock] [--action=[downgrade|remove]] [--locker=name] [(-r rev|--revision=rev)] [(-R|--
[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--
sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name]
[(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]]
[(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si unlock` removes or downgrades the lock on a member. If you do not specify any members, `si unlock` applies to all members. By default, `si unlock` downgrades your lock on the member revision of each archived member.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--action=[downgrade|remove]`

  controls whether you want to remove or downgrade the lock on the member.

  `--action=downgrade` means downgrade an exclusive lock to a non-exclusive lock.

  `--action=remove`

  means remove the lock on the member.
- `--locker=name`

  specifies the name of another user whose lock you want to remove or downgrade. If this option is not specified, the command removes or downgrades your lock.
- `--[no|confirm]breakLock`

  controls whether to remove or downgrade someone else's lock. If someone else has the member locked and you have the `DowngradeOtherUserLock` or `RemoveOtherUserLock` permission (see ACL), you will normally be prompted to confirm whether you want to remove or downgrade their locks.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the `diagnostics` reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si edit, si lock, si rlog
- Miscellaneous: ACL, diagnostics, options, preferences

si unlockarchive

unlocks a revision in an archive.

Synopsis

```
si unlockarchive [--archive=value] [-r|--revision=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [-F|--selectionFile=value] [--status=[none|gui|default]]
```

Description

`si unlockarchive` unlocks a revision in an archive. You can use this command to unlock members that have been dropped from a project, and members locked in a project for which you longer have permission to view.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--archive=value`

  specifies the server side archive name.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewprojecthistory, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si unrestrictproject

removes restriction preventing changes being made to a project hierarchy

Synopsis

```
si unrestrictproject [--project=value] [--user=name] [--password=password] [--hostname=server] [--port=number ] [--
[no|confirm]updateOnRestrictedByConflict] [--selectionFile=value] [--settingsUI=[gui|default]] [--
status=[none|gui|gui.actions|default]] [--forceConfirm=[yes|no]] [--gui] [--sandbox=value] [--cwd=value] [--selectionFile=value] [--
[no]batch] [--quiet] [--usage] [--settingsUI=[gui|default]] [--user] [--usage] [--yes] [--[no]failOnAmbiguousProject] [-F=file]
```

Description

`si unrestrictproject` removes the restriction that prevents changes from being made to a project hierarchy. When the release is being finalized, the mainline or a branch of a project can be restricted or unrestricted recursively by the user with restrict project permissions.

The user with restrict project permissions can unrestrict a project or branch and reset the permissions to what they were prior to being restricted, which means all the users can perform changes to the project according to their normal permissions.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]updateOnRestrictedByConflict` specifies if to update project restriction even if project is already restricted by another user.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si restrict project
- Miscellaneous: ACL, diagnostics, options, preferences

si updatearchive

manipulates various member archive attributes

Synopsis

```
si updatearchive [--archiveDescription=description] [--archiveType=[binary]] [--defaultBranch=value] [--
storageFormat=[delta|reference]] [--[no]exclusiveLockMandatory] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--
project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--
hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-
Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] member...
```

Description

`si updatearchive` manipulates various member archive attributes.

Note

This command affects the base archive, and, therefore, all projects and development paths referring to the archive.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--archiveDescription=`*`description`*

  sets the archive description for a member. `description` is stored in the member's meta data as an archive description. If `description` has spaces then it must be enclosed by quotes.
- `--archiveType=`*`[binary]`*

  converts a text type archive to a binary type archive.

  Note

  You must have the `changeArchiveType` permission to use this option. You can only perform this conversion on a database type repository.

  Caution

  Once you have converted an archive to binary format, you cannot convert it back to text format.
- `--defaultBranch=`*`value`*

  sets the default branch Integrity uses to check in files. If unspecified, files are checked in to the trunk.
- `--storageFormat=`*`[delta|reference]`*

  identifies what type of format to use in the archive. This is recommended for use by the system ADMIN only. *`delta`* is the traditional Integrity reverse delta format, while *`reference`* means to store member history by reference -- each revision is stored as a separate file. This feature can enhance performance for text file archives, and the option can be used for archives stored in either a file system or database repository.
- `--[no]exclusiveLockMandatory`

  controls whether an exclusive locks policy is in effect for the archive. With an exclusive locks policy, you must have an exclusive lock before checking in any changes.
- *`member...`*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si createdevpath, si edit, si resync, si revert, si updaterevision
- Miscellaneous: ACL, diagnostics, options, preferences

si updateclient

updates the Integrity Client with a Service Pack

Synopsis

`si updateclient [--[no|confirm]download] [--[no|confirm]shutdown] [--[no|confirm]rollback] [--[no|confirm]rollbackshutdown] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`]`

Description

`si updateclient` updates the Integrity Client with a Service Pack from the server if one is available. A Service Pack may be designated as required to address a known issue, or may provide enhancements. Client side Service Pack numbers are designated with a "C", for example, C04030003.

Options

This command takes the universal options available to some `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]download`
  automatically downloads a Service Pack if one is available.
- `--[no|confirm]shutdown`
  automatically shutdowns the client if a Service Pack is downloaded.
- `--[no|confirm]rollback`
  automatically initiates a service pack rollback, if required to connect to the Integrity Server.
- `--[no|confirm]rollbackshutdown`
  automatically shutdowns the client if a service pack rollback is initiated.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si connect, si disconnect
- Miscellaneous: ACL, diagnostics, options, preferences

si updaterevision

updates a project member revision to a specific revision

Synopsis

```
si updaterevision [--cpid|--changepackageId=ID] [--[no|confirm]closeCP] [--[no]defer] [--issueId=value] [(-r rev|--revision=rev)] [(-R|--[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]save] member...
```

Description

`si updaterevision` updates a member revision to a specific pre-existing revision. This is useful when you want the member revisions in a project to reflect revisions based on a symbolic location in the development path (working file, head revision, trunk tip, member branch tip), property (state, label, timestamp), current project configuration, or external project configuration.

Note

You cannot update a frozen member's revision. You must first thaw the member (

---

si thaw) and then update it.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the `options` reference page for descriptions.

- `--[no]save`

  Used to save any revision specified by `-r` (see options). Although existing saved revisions can be referenced using the `:rule` symbolic revision, this option has been deprecated in favor of the member rule. PTC recommends using the member rule instead of the `--[no]save` option. For more information about the member rule, see si setmemberrule).
- `--[no|confirm]closeCP`

  closes the change package after command completion.
- `--[no]defer`

  controls whether to delay the operation in the project until the deferred operation is submitted. The operation in the Sandbox still takes place immediately.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si edit, si resync, si revert, si updatearchive, si setmemberrule
- Miscellaneous: ACL, diagnostics, options, preferences

si viewauditlog

view a record of operations performed on the Integrity Server

Synopsis

`si viewauditlog [--fields=field1[:width1],field2[:width2]...] [--filter[=id:<expression>] [--[no]batch] [--height=value] [--maxRows=value] [--width=value] [-x value] [-y value] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui][--user=name] [--hostname=server] [(-N|--no)] [--password=password] [--[no]persist] [--port=number] [(-Ffile|--selectionFile=file)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-?|--usage)] [(-Y|--yes)]`

Description

`si viewauditlog` allows you to view a record of operations performed on the Integrity Server

Auditing categories are organized into independent hierarchies, with an individual Integrity Server component at the root of each category (workflows and documents–`im`, configuration management–`si`, or the Integrity Server–`is`). These root component categories are independent of one another.

For example, the following command returns all records where "co" (check out) is in the operation name, including "recomputehistory":

```
si viewauditlog --filter=operation:co
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...]`

  where `fieldn` can be any of:

  - `id`

    specifies the ID number of an individual audit log detail. This is the ID number of an individual record found in a previous search.
  - `parentid`

    specifies the ID of the audit detail that spawned the specified operation (that is, all operations that have been spawned by the audit detail with the specified ID number).
  - `user`

    specifies a user ID. This is the login ID of the user who performed the operation.
  - `state`

    specifies the status of the operation being audited. Valid states are `completed` or `failed`.
  - `category`

    specifies the root component of Integrity. For example, workflows and documents is `im`, configuration management is `si`, and the Integrity Server is `is`.
  - `operation`

    specifies the operation or command. This can also include the operation that spawned the audit or sub-operation.
  - `contexttype`

    applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project`, `si.buildProject`, and `si.variantProject`.
  - `context`

    applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project` in the format *<project id>*, `si.buildProject` in the format *<project id>(<revision ID>*), and `si.variantProject` in the format *<project id>(<variant name>)*.
  - `date`

    specifies the timestamp information for the target operation.
  - `detail`

    specifies the audit log detail.
  - `targettype`

    specifies the type of object the specified operation is acting on. For example, `si.member`, `im.state`, and `im.issue`.
  - `target`

    specifies the object that the specified operation acts on. Optional field.
  - `parameters`

    specifies the parameters defined in the operation. Parameters take the form of `key=value` pairs in a comma-separated string. Parameter keys and values depend on the operation being recorded, for example, `label=Version2,saveTimestamp=true`. For configuration management, arguments should match those created for the corresponding trigger scripts. Optional field.
  - `resulttype`

    specifies the type of operation result. Valid result types include `exception` in the format *<exception class>*;, `im.issue` in the format *<issue ID>*, `si.revision` in the format *<revision ID>*.
  - `result`

    specifies the operation result. Valid results include `exception`, `im.issue`, and `si.revision`. Optional field.
- `--filter[=value:<expression>]`

  specifies filter to apply when viewing the audit log results. Valid filters include the following:

  - `id:<expression>`

    specifies the ID number of an individual audit log detail. This allows you to search for an individual record found in a previous search.
  - `parentid:<expression>`

    specifies the ID of the audit detail that spawned the specified operation (that is, all operations that have been spawned by the audit detail with the specified ID number).
  - `user:<expression>`

    specifies a user ID to search for when filtering the audit log. This is the login ID of the user who performed the operation.
  - `state:<expression>`

    specifies the status of the operation being audited. Valid states are `completed` or `failed`.
  - `timestamp:<expression>`

    specifies the timestamp information for the target operation. Valid filter options are `today¦tomorrow¦yesterday`, `last¦next<total number of days`, `between <MMM DD\, YYYY>` and `<MMM DD\, YYYY>`.
  - `category:<expression>`

    specifies the root component of Integrity you want to filter for. For example, workflows and documents is `im`, configuration management is `si`, and the Integrity Server is `is`. To refine the level of filtering, you can also specify a subcomponent. For example, to filter the audit log for workflow and document administrative operations use `im.admin` as the expression. To filter for member operations, use `si.member` as the expression.
  - `operation:<expression>`

    specifies the operation or command being filtered in audit log. This can also include the operation that spawned the audit or sub-operation.
  - `contexttype:<expression>`

    applies for applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project`, `si.buildProject`, and `si.variantProject`.

- `context:<expression>`

  applies for configuration management only. Applies only when the target of the operation is a member and the project must be given to uniquely identify that member. Valid contexts are `si.project` in the format *<project id>*, `si.buildProject` in the format *<project id>(<revision ID>)*, and `si.variantProject` in the format *<project id>(<variant name>)*.
- `targettype:<expression>`

  specifies the type of object the specified operation is acting on. For example, `si.member`, `im.state`, and `im.issue`.
- `target:<expression>`

  specifies the object that the specified operation acts on. Optional field.
- `parameters:<expression>`

  specifies the parameters defined in the operation. Parameters take the form of `key=value` pairs in a comma-separated string. Parameter keys and values depend on the operation being recorded, for example, `label=Version2,saveTimestamp=true`. For configuration management, arguments should match those created for the corresponding trigger scripts. Optional field.
- `resulttype:<expression>`

  specifies the type of operation result to filter for. Valid result types include `exception` in the format *<exception class>*;, `im.issue` in the format *<issue ID>*, `si.revision` in the format *<revision ID>*.
- `result:<expression>`

  specifies the operation result to filter for. Valid results include `exception`, `im.issue`, and `si.revision`. Optional field.

- `--maxRows=value`

  specifies the maximum number of audit log view records to return from the server.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

See Also

- Commands: im purgeauditlog, im viewauditlog, si purgeauditlog
- Miscellaneous: options

si viewcp

displays details for an Integrity configuration management change package

Synopsis

si viewcp [--fields=*field1[:width1],field2[:width2]...*] [--[no]showCommitted] [--[no]showUncommitted] [--[no]showPending] [--[no]showReviewLog] [--[no]showPropagationInfo] [--format=*value*] [--height=*value*] [--width=*value*] [-x *value*] [-y *value*] [--headerFormat=*value*] [--noFormat=*value*] [--noHeaderFormat=*value*] [--noTrailerFormat=*value*] [--trailerFormat=*value*] [--width=*value*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F file|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [--quiet] [(-g|--gui)] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] issue|issue:change package id...

Description

si viewcp allows you to view details on any Integrity configuration management change package you select. For example:

```
si viewcp 2:23
```

You can select the change package using an issue ID or change package ID, and the change package does not have to be assigned to you.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --fields=*field1[:width1],field2[:width2]...*

  allows you to select fields to be printed, specified in the format field1[:width1],field2[:width2]... Specifying the column [:width] (in pixels) for each field is optional. Fields are separated with a space.

  The fields available for printing can be one or more of the following:

- bytesadded

  displays the number of bytes added by the operation. For text files, this field displays 0.

- bytesdeleted

  displays the number of bytes deleted by the operation. For text files, this field displays 0.

- closeddate

  displays the date the change package was closed on.

- cptype

  displays the change package type.

- configpath

  displays the configuration path of the change package entry (repository location of the member/subproject).

- creationdate

  displays date the change package was created..

- description

  displays the description of the selected change package.

- haspropagated

  displays 1 if the change package has propagated other change packages, 0 if not.

- haspropagatedby

  displays 1 if the change package has been propagated by other change packages, 0 if not.

- hasreverted

  displays 1 if the change package has reverted at least one other change package, 0 if not.

- hasrevertedby

  displays 1 if the change package was reverted by at least one other change package, 0 if not.

- id

  displays the ID of the issue associated with the selected change package.

- isclosed

  indicates if the change package is closed.

- istext

  indicates if the change package entry has a text archive.

- linesadded

  displays the number of lines added by the operation. For binary files, this field displays 0.

- linesdeleted

  displays the number of lines deleted by the operation. For binary files, this field displays 0.

- location

  displays the archive location for members or the location of the parent project for subprojects.

- member

  displays the name of the member or subproject affected by the operation.

- membertype

  displays the type of project element affected by the operation (member or subproject).

- project

  displays the name and path of the project where the operation was performed. If the operation occurred in a shared subproject, the subproject name and path are displayed. If the operation involved two different projects (for example, moving a member from one project to another), information for both projects is displayed.

- propagated

  displays a list of the change packages that have been propagated by the selected change package.

- propagatedby

  displays a list of the change packages that propagated the selected change package.

- reverted

  displays the IDs of the change packages that were reverted by the change package being viewed.

- revertedby

  displays the IDs of the change packages that reverted the change package being viewed.

- revision

  displays the member revision number.

- sandbox

  displays the name of the sandbox where the deferred operation or check out took place.

- server

  displays the server the change package resides on.

- showpropagationinfo

displays the propagation information for the change package.

- `siserver`

  displays the Integrity Server the change package resides on.
- `state`

  displays the status of the change package.
- `summary`

  displays the change package summary.
- `timestamp`

  displays the timestamp of the selected change package.
- `type`

  displays how the member was added to the change package.
- `user`

  displays the ID of the user who added the member to the change package.
- `variant`

  displays the names of variant projects associated with the member.
- `--[no]showPropagationInfo`

  specifies whether to display a list of the change packages that have been propagated by or reverted by the selected change package, and a list of the change packages that propagated or reverted the selected change package.
- `--[no]showCommitted`

  specifies whether to display committed operations. Committed operations appear as change package entries that contain changes committed to the server repository.
- `--[no]showUncommitted`

  specifies whether to display uncommitted operations. Uncommitted operations are work in progress, lock or deferred entries that can be submitted to the server repository.
- `--[no]showPending`

  specifies whether to display pending entries that have not yet committed to the repository.
- `--[no]showReviewLog`

  specifies whether to display review information. Integrity provides review logs as part of a complete review audit process. A log consists of individual records for each reviewer vote cast. Each time the change package is submitted for review, a new log begins.

  Each review record contains the following information:

  `Review States` The possible review states are:

  - `Review Pending` the change package is still in a state of Submitted and there are outstanding votes to be cast by reviewers.
  - `Accepted` the change package is in a state of Accepted, and all of the individual reviewers and a user from each reviewer group have accepted the change package.
  - `Rejected` the change package is in a state of Rejected, and at least one reviewer has rejected the change package.

  `Reviewer Type` The possible reviewer types are:

  - `User Reviewer` is a user voting in an individual capacity.
  - `Group Reviewer` is a user voting in a group capacity on behalf of that group.
  - `Super Reviewer` is a user casting an overriding vote in a super reviewer capacity, and is not required to be a user on the reviewer list or in a group on the group reviewer list.

  `Votes` The possible vote values are:

  - `Pending` signifies that the reviewer or group reviewer has not yet cast a vote.
  - `Accepted` signifies that the user has cast an accept vote for the change package.
  - `Rejected` signifies that the user has cast a reject vote for the change package.

  `Comments` Displays information that reviewers optionally provide to clarify their votes.

  `Changes` Displays in tabular form the changes that were made to upon submission of the change package.

  `si viewcp` also includes options that deal with formatting for the view.

- `--format=`*`value`*
- defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  This option and the `--fields` option are mutually exclusive.

  `--format` options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires { } to surround each field). The --fields option automatically adds a newline on the end, but you must supply the newline for formats with a *n*, for example:
  `si viewcp --format="{member},{revision}\n"`
- `--headerFormat=`*`value`*

  header for user-formatted text (see `--format`).
- `--noFormat=`*`value`*

  format for user-formatted text (see `--format`).
- `--noHeaderFormat=`*`value`*

  header for user-formatted text (see `--format`).
- `--noTrailerFormat=`*`value`*

  trailer for user-formatted text (see `--format`).
- `--trailerFormat=`*`value`*

  trailer for user-formatted text (see `--format`).
- *issue...*
- *issue:change package id...*

  `issue` identifies a specific issue that contains all change packages that you want to view; use spaces to specify more than one issue.

  issue:change package id identifies a specific change package to view; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si closecp, si co, si cpissues, si createcp, si drop, si lock
- Miscellaneous: ACL, diagnostics, options, preferences

si viewcpentries

displays the net effect of a set of change packages, or the net effect of all of the change packages on a selection of items.

Synopsis

`si viewcpentries [--fields=field1[:width1],field2[:width2]...] [--format=value] [--height=value] [--width=value] [-x value] [-y value] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--hostname=server] [--password=password] [--[no]persist] [--port=number] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--user=name]`

Description

`si viewcpentries` displays a list of the change operations that occurred on a set of change packages or on all change packages for a selection of items, if those operations were tracked in change packages or related to change packages. The displayed information can be used to identify changes related to a certain feature, report on the changes that went into a release or component, or as part of investigating defects that may have resulted from those change operations.

The following is an example of using the command to return a list of all affected members and subprojects in a selection of change packages and items:

```
si viewcpentries 1 2 3 2:1 3:4 5
```

The following are key considerations when using this command:

- This command accepts a selection of either change package IDs or Integrity Item IDs or both.
- This command returns changes that were recorded in selected change packages or all change packages for a selection of items.
- Operations that do not reside on the server, including deferred entries, are not represented in the command output. Also, operations that are not recorded in change packages, such as a restore project (or commands that used the `bypass` value for `--changePackageId`), are not represented in the command output.
- Modifications to member attributes are not included in the command output.
- If subprojects were added using a change package, only that operation is included in the command output. The members that were added with the subproject are not included in the command output because they were not recorded in the change package. Similarly, reconfigured subprojects appear in the command output, but modifications to the members of the subproject do not appear in the command output.
- When propagations cause an update to a revision number, the intervening revisions are not included in the command output because the operation that created them is not in the change package for that time frame.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`
- allows you to select fields to be printed, specified in the format field1[:width1],field2[:width2]....

  Specifying the column [:width] (in pixels) for each field is optional. Fields are separated with a space.

  The fields available for printing can be one or more of the following:
- `bytesadded`

  displays the number of bytes added by the operation. For text files, this field displays 0.
- `bytesdeleted`

  displays the number of bytes deleted by the operation. For text files, this field displays 0.
- `configpath`

  displays the configuration path of the change package entry.
- `id`

  displays the change package ID.
- `istext`

  indicates if the change package entry has a text archive; `true` or `false`. If `false`, the entry is a binary archive.
- `linesadded`

  displays the number of lines added by the operation. For binary files, this field displays 0.
- `linesdeleted`

  displays the number of lines deleted by the operation. For binary files, this field displays 0.
- `location`

  displays the archive location for members or the location of the parent project for subprojects (the canonical path).
- `member`

  displays the name of the member or subproject affected by the operation.
- `membertype`

  displays the type of project element affected by the operation (member or subproject).
- `project`

  displays the name and path of the project where the operation was performed. If the operation occurred in a shared subproject, the subproject name and path are displayed. If the operation involved two different projects (for example, moving a member from one project to another), the command returns two entries, with the first entry the originating project and the second entry the destination project.
- `revision`

  displays the member revision number.
- `server`

  displays the server the change package resides on.
- `summary`

  displays the change package summary.
- `timestamp`

  displays the timestamp of the selected change package.
- `type`

  displays how the member was added to the change package.
- `user`

  displays the ID of the user who added the member to the change package.
- `variant`

  displays the names of variant projects associated with the member.
- `--format=value`

  defines an output format for user-formatted text. The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.
  This option and the `--fields` option are mutually exclusive.

  --format options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires { } to surround each field). The `--fields` option automatically adds a newline on the end, but you must supply the newline for formats with a \n, for example: `si viewcpentries --format="{project},{member},{revision}\n"`

Diagnostics

See the diagnostics reference page for possible exit status values.

See Also

- Commands: si viewcp, si viewcps
- Miscellaneous: diagnostics, options

si viewcps

displays all open change packages created by you

Synopsis

`si viewcps [--fields=field1[:width1],field2[:width2]...] [--filter=value] [--height=value] [--width=value] [-x value] [-y value] [--query=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]persist] [--quiet] [--myReviews] [(-g|--gui)] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [--[no]persist] issue|issue:change package id...`

Description

`si viewcps` displays all open change packages created by you.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--myReviews`

  displays change packages you are to review (which may include change packages not created by you). Change packages only appear in the list after they have been submitted for review. After you vote on the change package, it no longer appears in this list.
- `--query=value`

  the Integrity query used to find change packages. For information on creating and using queries, see the reference page for this command.
- `--[no]persist`

  controls the persistence of CLI views.
- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format *field1[:width1],field2[:width2]...*. Specifying the column *[:width]* (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a `space`.

  The fields available for printing can be one or more of the following:
  - `closeddate`

    displays the date the change package was closed.
  - `cptype`

    displays the change package type.
  - `creationdate`

    displays the date the change package was created.
  - `description`

    displays change package descriptions.
  - `id`

    displays change packages IDs.
  - `summary`

    displays change package summaries.
  - `issue`

    displays the issue ID if the Integrity integration is enabled.
  - `propagated`

    displays the list of change packages propagated by the change package (the list is displayed by default for propagation change packages).
  - `propagatedby`

    displays the list of change packages that propagated the change package.
  - `reverted`

    displays the IDs of the change packages that were reverted by the change package being viewed.
  - `revertedby`

    displays the IDs of the change packages that reverted the change package being viewed.
  - `state`

    displays the current state of the change package.
  - `siserver`

    displays the Integrity server the change package resides on.
  - `state`

    displays the status of the change package.
  - `user`

    displays the username who created the change package.
- `--filter=value`

  displays the change packages according to one of the following filters:
  - `issueID:<issue>`

    displays change packages based on specified issue ID.
  - `state[:closed|:open|:submitted|:accepted|:rejected|:discarded|:commitfailed]`

    displays change packages according to their state, which can be one of the following:
    - `:closed` specifies that the change package is in a closed state (all changes committed to repository).
    - `:open` specifies that the change package has in an open state (work in progress, available to add entries too).
    - `:submitted` specifies that the change package has been submitted for review.
    - `:accepted` specifies that the change package has been accepted by all reviewers.
    - `:rejected` specifies that the change package has been rejected by at least one reviewer.
    - `:discarded` specifies that the change package has been discarded.
    - `:commitfailed` specifies that the change package has failed to commit to the repository.
  - `cptype[:development|:propagation|:resolution]`

    displays change packages based on the type of change package.
  - `closeddate:<date>`

    displays closed change packages based on a specified closed date. Valid date formats are the following: between *MM/dd/yyyy* and *MM/dd/yyyy*, `in the last|in the next` *number* `days|months|years yesterday|today|tomorrow`.
  - `creationdate:<date>`

    displays change packages based on a specified creation date. Valid date formats are the following: between *MM/dd/yyyy* and *MM/dd/yyyy*, `in the last|in the next` *number* `days|months|years yesterday|today|tomorrow`.
  - `membertype[:member|:subproject];`

    displays change packages based on the type of project element they contain.

- member:<expression>

  specifies a text string to filter change packages by member name.
- project:<expression>

  specifies a text string to find change packages with members belonging to a specified project. This includes all change packages on the mainline of the specified project, as well as those on variants of the project.
- mainline

  finds change packages with members that exist only on the mainline (or main trunk) of any project. To find change packages on the mainline of a specific project, you must add the project filter.
- variant:<expression>

  specifies a text string to find change packages with members that exist only on a specified project variant.
- description:<expression>

  specifies a text string to filter change packages by their description.
- summary:<expression>

  specifies a text string to filter change packages by their summary.
- type[:add|:addfromarchive|:drop|:import|:exclusivelock|:nonexclusivelock|:movememberto|:movememberfrom |:renamefrom|:renameto|:update|:updatearchive|:updaterevision|:createsubproject |:addsubproject|:addsharedsubproject|:configuresubprojectfrom|:configuresubprojectto |:movesubprojectfrom|:movesubprojectto|:dropsubproject]

  displays change packages based on their entry type.
- typemodifier[:committed|:pending]

  displays change packages based on their entry category.
- hasissue

  displays change packages that have an associated issue.
- pendingreviewby:name

  displays change packages that have not yet been accepted or rejected by a specified reviewer.
- user:name

  displays change packages created by a specified username.
- acceptedby:name[;date]

  displays change packages accepted by a specified username on a specified date. Valid date formats are the following: between *MM/dd/yyyy* and *MM/dd/yyyy*, in the last|in the next number days|months|years yesterday|today|tomorrow.
- rejectedby:name[;date]
- displays change packages rejected by a specified username on a specified date. Valid date formats are the following: between *MM/dd/yyyy* and *MM/dd/yyyy*, in the last|in the next number days|months|years yesterday|today|tomorrow.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si add, si ci, si closecp, si co, si cpissues, si createcp, si drop, si lock
- Miscellaneous: ACL, diagnostics, options, preferences

si viewhistory

displays a member's history

Synopsis

si viewhistory [--fields=*field1[:width1],field2[:width2]...*] [--height=*value*] [--width=*value*] [-x *value*] [-y *value*] [--filter=*filteroptions*] [--rfilter=*filteroptions*][(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--lockRecordFormat=*value*] [--lockRecordDetailFormat=*value*] [--maxTrunkRevs=*value*] [--checkpointLabelFilter=*value*] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--maxTrunkRevs=*value*] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*] *member...*

Description

si viewhistory displays a member's history, one line per revision, per member. This is similar to si print and si rlog, but with different defaults.

While Sandboxes and projects allow you to manage and access project members and the contents of individual members, the history of changes are saved in member histories.

Integrity lets you save and recreate every stage (or revision) in the development of each member you use. When you make changes to a project member and check it back in, your changes are automatically added to the member history.

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --fields=*field1[:width1],field2[:width2]...*

  allows you to select fields to be printed, specified in the format field1[:width1],field2[:width2]... Specifying the column [:width] (in pixels) for each field is optional - widths are only available with the -g or --gui options. Under the CLI the fields are separated with a space.

  The fields available for printing can be one or more of the following:

  - author

    displays author names.
  - checkpointlabels

    displays labels of checkpoints in which the member revision participated. This field is not displayed by default. Can be used with --checkpointLabelFilter to filter the checkpoint labels displayed.
  - cpid

    displays the associated change package identifier (applies only if change packages are enabled).
  - date

    displays the revision date.
  - description

    displays revision descriptions.
  - labels

    displays labels.
  - lockrecord

    a comma separated list of locks on the revision. The locker and lock type are displayed by default for each lock. You can customize the lock information that displays in the List Member History view in the GUI by using the --lockRecordFormat option. You can customize the lock information that displays in the Details panel of the Graphical Member History view in the GUI by using --lockRecordDetailFormat option.
  - state

    displays the state.
- --[no]persist

  controls whether this presentation of information should continue to be updated as new information becomes available. --nopersist forces a static "snapshot" of information, while --persist gives real-time updates.
- --rfilter=*filteroptions*

  allows you to display any revisions of the current member that meet the selection criteria specified in the filteroptions. Comma separated expressions are combined using an OR operator. Multiple occurrences of an option are combined using an AND operator. A leading ! negates any simple filter that follows. For example,

```
--rfilter=labeled:Release 3.0,branchboundaries --rfilter=!labeled:obsolete
```

displays all revisions with a label of "Release 3.0" or at a branch boundary and whose label is not "Obsolete". The *filteroptions* can be one or more of the following:

- range:*low-high*

  selects revisions that are in the specified revision number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the range keyword must be repeated, for example --rfilter=range:1.100-1.200,range:1:400-1:500
- branchrange:*low-high*

  selects branched revisions that are in the specified revision number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the branchrange keyword must be repeated, for example --rfilter=branchrange:1.2.1.1-1.2.1.5,branchrange:1.4.1.1-1.4.1.5
- daterange:*<date>-<date>*|*:past:<nb>:s*years|months|days|hour

  selects revisions that were created during a specified time range or during a specified amount of time in the past. <date> specifies a date in any of the local date/time formats. For more information on date/time formats, see the options reference page. <nb> is used to specify the number of units to include in the range for a specified amount of time in the past.

Note

If the date/time format contains either "-" or "," you must quote each date. For example: --rfilter=daterange:"June 21, 2009 - 6:07:28 PM"-"June 25, 2009 - 6:07:28 PM"

When in a shell, you must use two kinds of quotes so that the shell does not remove them. For example: --rfilter=daterange:'"June 21, 2009 - 6:07:28 PM"'-'"June 27, 2005 - 6:07:28 PM"' or--rfilter=daterange:"'June 21, 2009 - 6:07:28 PM'"-"'June 25, 2009 - 6:07:28 PM'"

- branch[::current|*:name*]

  selects revisions that are on the specified branch.
- labeled*:name*

  selects revisions with labels. If a label name is specified, only revisions with that label are displayed. If no label name is specified, all labeled revisions are displayed.
- labellike*:pattern*

  selects revisions that match the specified pattern. Your administrator determines whether glob or regex patterns are used for matching.
- locked[::me|::anyone|*userName*]

  selects revisions that are locked. If me or a user name is specified, only revisions locked by that user are displayed. If no user or anyone is specified, all locked revisions are displayed.

- `locktype[:exclusive|:nonexclusive|:any]`

  selects revisions that are locked with the specified lock type. If no lock type or `any` is specified, all locked revisions are displayed.
- `state:`*`name`*

  selects revisions that are at the specified state.
- `author[::me|:`*`userName`*`]`

  selects revisions created by the specified author.
- `pending`

  selects revisions that are pending.
- `anyspecial`

  selects working and member revisions.
- `branchboundaries`

  selects revisions that are branched, or at the base or tip of a branch.

`--lockRecordFormat=`*`value`*

defines the format for displaying lock information in the `--lockrecord` field of the List Member History view in the GUI. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:

- `{revision}`

  displays the revision that is locked.
- `{locker}`

  displays the user who locked the revision.
- `{locktype}`

  displays the type of lock on the revision (exclusive or non-exclusive).
- `{locktimestamp}`

  displays the time when the revision was locked.
- `{lockcpid}`

  displays the change package associated with the lock on the revision.
- `{project}`

  displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
- `{devpath}`

  displays the name of the development path where the lock on the revision was made from.
- `{sandbox}`

  displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
- `{hostname}`

  displays the hostname of the computer that locked the the revision.
- `{hascpid}`

  displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
- `{hassandbox}`

  displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
- `{hasdevpath}`

  displays 1 if the lock was made from a development path, 0 if it wasn't.
- `{member}`

  displays the name of the locked revision.

`--lockRecordDetailFormat=`*`value`*

defines the format for displaying lock information in the `--lockrecord` field of the Graphical Member History view. Specify a format string using keywords to represent the information you want to display. See the `--lockRecordFormat` option for a list of keywords.

`--maxTrunkRevs=`*`value`*

specifies the maximum number of revisions to display. If you limit the number of revisions to be displayed, and the maximum is reached, a trailing message displays at the bottom of the list. The maximum number of revisions is based on the number of visible mainline revisions; branched revisions are not included in the count.

`--checkpointLabelFilter=`*`value`*

specifies a regular expression used to filter the checkpoint labels displayed for the member revisions. Used with `--fields=checkpointlabels`, for example:

```
si viewhistory --checkpointLabelFilter=IS_.* --fields=revision,author,date,checkpointlabels Console.java
```

`member...`

identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewlabels, si viewproject, si viewprojecthistory, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si viewlabels

displays a member's labels

Synopsis

```
si viewlabels [--fields=field1[:width1],field2[:width2]...] [--height=value] [--width=value] [-x value] [-y value] [(-R|--[no|confirm]recurse)] [--[no]failOnAmbiguousProject] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si viewlabels` displays a member's labels: one label per line, showing the label and its revision number.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`
- allows you to select fields to be printed, specified in the format `field1[:width1],field2[:width2]...`. Specifying the column `[:width]` (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a space.

  The fields available for printing can be one or more of the following:
- `cpid`

  displays the associated change package identifier (applies only if change packages are enabled).
- `label`

  displays labels.
- `revision`

  displays the revision number.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.
- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewproject, si viewprojecthistory, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si viewlocks

displays locks

Synopsis

```
si viewlocks [--format=value] [--headerFormat=value] [--noFormat] [--noHeaderFormat] [--noTrailerFormat] [(-r rev|--revision=rev)] [--range=value] [--trailerFormat=value] [--fields=field1[:width1],field2[:width2]...] [--[no]failOnAmbiguousProject] [--maxTrunkRevs=value] [--lockRecordFormat=value] [--lockRecordDetailFormat=value] [--lockseparator=value] [--rfilter=filteroptions] [--height=value] [--width=value] [-x value] [-y value] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [--[no]persist] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si viewlocks` displays the locks held in a project and some information about the members, for example:

```
c:\app\connect.txt 1.4 mkern
Jan 9, 2002 - 4:06 PM
c:\app\source.txt 1.3 mkern
Jan 9, 2002 - 4:06 PM
c:\app\config.c 1.3 mkern
Jan 8, 2002 - 2:26 PM
```

Note

By default, `si viewlocks` displays the member names of locked members, not the working file names.

Options

This command takes some of the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

See the si rlog reference page for descriptions of all options applicable to `si viewlocks`.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewprojecthistory, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si viewmetricsinfo

displays information about the metrics being tracked

Synopsis

`si viewmetricsinfo [--fields=`*`field1[:width1],field2[:width2]...`*`] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F `*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`]`

Description

`si viewmetricsinfo` displays information about the metrics created using `si createmetricinfo`.

Note

Metrics are only supported for database type repositories.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=`*`field1[:width1],field2[:width2]...`*

  allows you to select fields to be printed, specified in the format *`field1[:width1],field2[:width2]...`*. Specifying the column *`[:width]`* (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a `space`.

  The fields available for printing can be one or more of the following:

- `description`

  displays the description of the metric.

- `metric`

  displays the value of the metric.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si createmetricinfo, si calculateprojectmetrics, si viewprojectmetrics, si addprojectmetric
- Miscellaneous: diagnostics, options, preferences

si viewnonmembers

displays non-members in a Sandbox

Synopsis

`si viewnonmembers [--[no|confirm]includeFormers] [--exclude=`*`file:pattern,dir:pattern...`*`] [--include=`*`file:pattern,dir:pattern...`*`] [--fields=`*`field1[:width1],field2[:width2]...`*`] [-R|--[no|confirm]recurse] [-S|--sandbox=`*`value`*`] [--[no]failOnAmbiguousProject] [--hostname=`*`server`*`] [--port=`*`number`*`] [--password=`*`password`*`] [--user=`*`name`*`] [(-?|--usage)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [(-g|--gui)] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`nonmember...`

Description

`si viewnonmembers` displays non-members in a Sandbox.

From Integrity, you can view non-member files existent in your Sandbox directory. The Non-Members view is useful when used recursively to identify all of the files that need to be placed under source control. By default, former members are not displayed in the Non-Members view.

The Non-Members view does not display files that are:

- deferred imported members
- deferred add members from archive
- pending members (add, add from archive, import)
- working files from members that have been renamed but not resynchronized

Non-members can only be viewed in a Sandbox context, and not from any project view operation.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|confirm]includeFormers`

  specifies if to include members that have been dropped from the project, but where there still exists a working file in the Sandbox directory..
- `--exclude=`*`file:pattern,dir:pattern...`*

  specifies a file that contains a glob pattern for excluding members.
- `--include=`*`file:pattern,dir:pattern...`*

  specifies a file that contains a glob pattern for including members.
- `--fields=`*`field1[:width1],field2[:width2]...`*

  allows you to select fields to be displayed, specified in the format *`field1[:width1],field2[:width2]...`*. Specifying the column *`[:width]`* (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options.

  The fields available for printing can be one or more of the following:

  - `absolutepath`

    displays the absolute file path of the file.
  - `closestproject`

    displays the project associated with the Sandbox that is closest to the directory containing the file.
  - `closestsandbox`

    displays the Sandbox that is closest to the directory containing the file.
  - `lastmodified`

    displays the date that the file was last modified.
  - `memberid`

    displays the default member name for the file as it would appear if it was added to the nearest project. In the case where the nearest project is subproject, the relative path is displayed with the member name.
  - `size`

    displays the size of the file in bytes.
  - `symboliclink`

    specifies whether the member is a symbolic link file member.
- *`nonmember...`*

  identifies nonmembers to display; use spaces to specify more than one change package.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si viewsandbox, si viewsproject, si add
- Miscellaneous:

  ACL, diagnostics, options, preferences

si viewpolicysection

displays the policy settings for an existing policy section (global or project)

Synopsis

```
si viewpolicysection [--height=value] [--width=value] [-x value] [-y value] [--hostname=server] [--port=number] [--password=password]
[--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] policy section name
```

Description

`si viewpolicysection` displays the policy settings for an existing policy section (global or project). For example, the following displays the policy settings for the global policy section on a server:

```
ChangePackagesEnabled=false
ChangePackagesMandatory=false
IntegrityManagerEnabled=false
IntegrityManagerIssueMandatory=false
ChangePackageReviewRule;append=true
ChangePackageWatcherRule;append=true
```

Note

The `ViewPolicy` permission is required.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--height=`*value*

  specifies the height of the GUI window, in pixels; *value* must be a whole number.
- `--width=`*value*

  specifies the width of the GUI window, in pixels; *value* must be a whole number.
- `-x` *value*

  specifies the location of the GUI window on the x axis, in pixels; *value* must be a whole number.
- `-y` *value*

  specifies the location of the GUI window on the y axis, in pixels; *value* must be a whole number.
- *policy section name*

  specifies the policy section to display, where *policy section name* is of the form *project,devpath*. For example:
  - `:global` for a global policy.
  - *project* for a project, for example, `/demo/project.pj`.
  - *project;development* path for a development path on a project, for example, `/demo/project.pj;Variant1`.
  - *development path* for all projects that are configured on the development path at the top level, for example, `;Variant1`.

  Note

  You may need to escape the ; in your command line environment, for example, enclose it in "" or escape the individual character with a /.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si copypolicysection, si deletepolicysection, si setpolicysection, si viewpolicysections
- Miscellaneous: diagnostics, options, preferences

si viewpolicysections

displays the existing policies (global or project) on an Integrity Server

Synopsis

```
si viewpolicysections [--[no]showPolicies=value] [--height=value] [--width=value] [-x value] [-y value] [--hostname=server] [--
port=number] [--password=password] [--user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--
status=[none|gui|default]] policy section name...
```

Description

`si viewpolicysections` displays the existing policy sections (global or project) on an Integrity Server. For example, the following global and project policy sections exist on a server:

```
ChangePackagesEnabled=false
ChangePackagesMandatory=false
IntegrityManagerEnabled=false
IntegrityManagerIssueMandatory=false
ChangePackageReviewRule;append=true
ChangePackageWatcherRule;append=true

/alpha/project1.pj
k=v
/service_pack/project.pj/project.pj
k=v
```

Note

The `ViewPolicy` permission is required.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showPolicies=`*value*

  specifies whether or not to display the policy settings in each policy section. If you do not specify this option, only the policy sections display.
- `--height=`*value*

  specifies the height of the GUI window, in pixels; *value* must be a whole number.
- `--width=`*value*

  specifies the width of the GUI window, in pixels; *value* must be a whole number.
- `-x` *value*

  specifies the location of the GUI window on the x axis, in pixels; *value* must be a whole number.
- `-y`*value*

  specifies the location of the GUI window on the y axis, in pixels; *value* must be a whole number.
- *policy section name...*

  specifies the policy section to display, where *policy section name* is of the form *project,devpath*. For example:
  - `:global` for a global policy.
  - *project* for a project, for example, `/demo/project.pj`.
  - *project;development path* for a development path on a project, for example, `/demo/project.pj;Variant1`.
  - *d;evelopment path* for all projects that are configured on the development path at the top level, for example, `;Variant1`.

    Note

    You may need to escape the ; in your command line environment, for example, enclose it in "" or escape the individual character with a /.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si copypolicysection, si deletepolicysection, si setpolicysection si viewpolicysection
- Miscellaneous: diagnostics, options, preferences

si viewprefs

displays preferences

Synopsis

```
si viewprefs [--[no]global] [--command=value] [--[no]showValidValues] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [(-F file|--selectionFile=file)] [--settingsUI=[gui|default]] [--status=[none|gui|default]]
```

Description

`si viewprefs` displays preferences and configuration options. These settings are used to determine default behaviors for other commands.

Your administrator can lock certain preferences from the Integrity Server, preventing you from configuring them using the si setprefs command. Preferences that are locked display (locked) at the end of the output line. The following preferences can be locked from the server by editing Integrity policies in the Administration Client:

Tip
For information on editing the Integrity policies, see the PTC Integrity Server Administration Guide.

Preference
branchIfVariant
breakLock
changePackageID
moveLock
onExistingArchive
restoreTimestamp
retainWorkingFile
saveTimestamp
sparse
updateMemberRev

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions. For an easy way to see a list of commands and values that may be set, simply type the `si viewprefs` command, either piped through `|more` or redirected to a file, for example:

```
si viewprefs --global --showValidValues >prefs.txt
```

Alternatively, the `--gui` option presents a simple-to-use dialog box that lets you view and configure the preferences.

- `--[no]global`

  controls whether to view all preferences.
- `--command=`*`value`*

  identifies the command preference to be viewed.
- `--[no]showValidValues`

  controls whether to list valid values for the preferences.
- `--[no]ask`

  controls whether to view the ask control over the preference options. See the description for `--[no]ask` on the si setprefs command.
- `-ui=-`*`[unspecified|gui|cli|api]`*

  controls whether to view the preference as it applies to the graphical user interface, the command line interface, application programming interface, or when the interface is unspecified. By default, `--ui=cli` is implied when issuing the `si viewprefs`. To view preferences for GUI behavior, however, you should specify `--ui=gui`.

  The preferences you see correlate to settings in the `IntegrityClient.rc` file, which can be seen as having the *gui.si.* or *cli.si.* prefix, or simply the *si.* prefix when it is unspecified.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si loadrc, si setprefs
- Miscellaneous: ACL, diagnostics, options, preferences

si viewproject

displays the contents of a project

Synopsis

`si viewproject [--fields=field1[:width1],field2[:width2]...] [--[no]filterSubs] [--height=value] [--width=value] [-x value] [-y value] [(-R|--[no|confirm]recurse)] [--filter=filteroptions] [(-P project|--project=project)] [--[no]failOnAmbiguousProject] [(-S sandbox|--sandbox=sandbox)] [--devpath=path] [--projectRevision=rev] [--lockRecordFormat=value] [(-F file|--selectionFile=file)] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member/subproject...`

Description

`si viewproject` displays the contents of a project and some information about the members, for example:

```
si viewproject c:/Aurora_Program/bin/Libra/project.pj
```

displays

```
connect.txt 1.4 archived
source.txt 1.3 archived
config.c 1.3 archived
```

Note

Specifying `si viewproject -S sandbox` does not view the Sandbox; it redirects through the Sandbox to view the project. Use si viewsandbox to view a Sandbox.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format *field1[:width1],field2[:width2]...*. Specifying the column *[:width]* (in pixels) for each field is optional. Widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a space.

  The fields available for printing can be one or more of the following:

  - `archiveshared`

    displays indicators for members that share another member's archive. This column is valid only if you are using the database repository.
  - `attributes`

    displays member attributes.
  - `context`

    When used with the Integrity API, displays the name of the project, and indicates if the project is a build or variant.

    Note

    This field is not valid for use with the CLI.

  - `cpid`

    displays the change package associated with the operation that set the member revision.
  - `creationcpid`

    displays the change package that created the revision that is currently the member revision. This revision may be different from the Member CPID if an import, add member from archive, or set member revision operation was used.
  - `frozen`

    displays indicators for frozen members.
  - `indent`

    indents a field. For example:

    ```
    si viewproject --fields=ident, name, type, memberrev
    ```

    indents the `name` field, followed by `type` and `memberrev` fields, indenting as you recurse into subprojects.

    Note

    This field is not valid for use with the Integrity API.

  - `labels`

    displays labels.
  - `lockrecord`

    a comma separated list of locks on the member. The locker and lock type are displayed by default for each lock. You can customize the lock information that displays by using the `--lockRecordFormat` option.
  - `memberarchive`

    displays the name and path of the member archive.
  - `memberdescription`

    displays the member description.
  - `memberrev`

    displays the member revision.
  - `membertimestamp`

    displays the member timestamp.
  - `name`

    displays the member name.
  - `newrevdelta`

    displays indicators for new revisions.
  - `pendingcpid`

    displays the change package associated with a pending operation.
  - `state`

    displays the member state.
  - `symboliclink`

    specifies whether the member is a symbolic link file member.
  - `type`

displays the type of each item in the project: project, subproject, shared-subproject, shared-variant-subproject, shared-build-subproject, or member.

- `--[no]filterSubs`

  controls whether to filter empty subprojects or subsandboxes from the view.

- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.

- `--lockRecordFormat=`*`value`*

  defines the format for displaying lock information in the `--lockrecord` field. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:

  - `{revision}`

    displays the revision that is locked.
  - `{locker}`

    displays the user who locked the revision.
  - `{locktype}`

    displays the type of lock on the revision (exclusive or non-exclusive).
  - `{locktimestamp}`

    displays the time when the revision was locked.
  - `{lockcpid}`

    displays the change package associated with the lock on the revision.
  - `{project}`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `{devpath}`

    displays the name of the development path where the lock on the revision was made from.
  - `{sandbox}`

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
  - `{hostname}`

    displays the hostname of the computer that locked the the revision.
  - `{hascpid}`

    displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
  - `{hassandbox}`

    displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
  - `{hasdevpath}`

    displays 1 if the lock was made from a development path, 0 if it wasn't.
  - `{member}`

    displays the name of the locked revision.

- *`member/subproject...`*

  identifies a specific member or subproject; use spaces to specify more than one. The project context cannot be specified in the selection (use the `-P` or `--project` option if the project context is required).

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewprojecthistory, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si viewprojecthistory

displays a history of the project

Synopsis

si viewprojecthistory [--fields=*field1[:width1],field2[:width2]...*] [--height=*value*] [--width=*value*] [-x *value*] [-y *value*] [(-P *project*|--project=*project*)] [--[no]failOnAmbiguousProject] [--rfilter=*filteroptions*] [(-S *sandbox*|--sandbox=*sandbox*)] [--devpath=*path*] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd =*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--maxTrunkRevs=*value*] [--[no]persist] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*][--[no]hideCheckpointsOnInactiveVariants]

Description

si viewprojecthistory displays a history of the project, including information on each historical checkpoint. For example:

si viewprojecthistory --project=c:/Aurora_Program/bin/Libra/project.pj

Options

This command takes the universal options available to all si commands, as well as some general options. See the options reference page for descriptions.

- --fields=*field1[:width1],field2[:width2]...*

  allows you to select fields to be printed, specified in the format *field1[:width1],field2[:width2]...*. Specifying the column *[:width]* (in pixels) for each field is optional - widths are only available with the -g or --gui options. Under the CLI the fields are separated with a space.

  The fields available for printing can be one or more of the following:

  - associatedIssues

    displays the ID and summary of any Integrity items associated with the project. Only item types that you have permission to view are displayed.
  - author

    displays author names.
  - date

    displays the project checkpoint date.
  - description

    displays checkpoint and project descriptions.
  - labels

    displays labels.
  - revision

    displays the checkpoint number.
  - state

    displays the checkpoint state.
- --[no]persist

  controls whether this presentation of information should continue to be updated as new information becomes available. --nopersist forces a static "snapshot" of information, while --persist gives real-time updates.
- --rfilter=*filteroptions*

  allows you to display any checkpoints of the current project that meet the selection criteria specified in the *filteroptions*. Comma separated expressions are combined using an OR operator. Multiple occurrences of an option are combined using an AND operator. A leading ! negates any simple filter that follows. For example,

  --rfilter=labeled:Release 3.0,branchboundaries --rfilter=!labeled:obsolete

  displays all checkpoints with a label of "Release 3.0" or at a branch boundary and whose label is not "obsolete". The *filteroptions* can be one or more of the following:

  - range*:low-high*

    selects checkpoints that are in the specified number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the range keyword must be repeated, for example --revisionFilter=range:1.100-1.200,range:1:400-1:500
  - branchrange*:low-high*

    selects branched checkpoints that are in the specified number range. If required, you can specify just one end of the range. The range includes the low and high values. To specify multiple ranges, the branchrange keyword must be repeated, for example --rfilter=branchrange:1.2.1.1-1.2.1.5,branchrange:1.4.1.1-1.4.1.5
  - daterange:*<date>*-*<date>*|:past:*<nb>*:years|months|days|hours

    selects checkpoints that were created during a specified time range or during a specified amount of time in the past. *<date>* specifies a date in any of the local date/time formats. For more information on date/time formats, see the options reference page. *<nb>* is used to specify the number of units to include in the range for a specified amount of time in the past.

    Note

    If the date/time format contains either "-" or "," you must quote each date. For example: --rfilter=daterange:"June 21, 2005 - 6:07:28 PM"-"June 25, 2005 - 6:07:28 PM"

    When in a shell, you must use two kinds of quotes so that the shell does not remove them. For example: --rfilter=daterange:'"June 21, 2005 - 6:07:28 PM"'-'"June 25, 2005 - 6:07:28 PM"'

    or

    --rfilter=daterange:"'June 21, 2005 - 6:07:28 PM'"-"'June 25, 2005 - 6:07:28 PM'"

  - devpath[::current|*:name*]

    selects checkpoints that are on the specified development path.
  - branch[::current|*:name*]

    selects checkpoints that are on the specified branch.
  - labeled*:name*

    selects checkpoints with labels. If a label name is specified, only checkpoints with that label are displayed. If no label name is specified, all labeled checkpoints are displayed.
  - labellike*:pattern*

    selects checkpoints that match the specified pattern. Your administrator determines whether glob or regex patterns are used for matching.
  - state*:name*

    selects checkpoints that are at the specified state.
  - author[::me|*:userName*]

    selects checkpoints created by the specified author.
  - anyspecial

    selects base project checkpoints.
  - branchboundaries

selects checkpoints that are branched, or at the base or tip of a branch.

- `--maxTrunkRevs=`*value*

  specifies the maximum number of checkpoints along the mainline project to display. If you limit the number of checkpoints to be displayed, and the maximum is reached, a trailing message displays at the bottom of the list. The maximum number of checkpoints is based on the number of visible mainline checkpoints; branched checkpoints are not included in the count.

- `--[no]hideCheckpointsOnInactiveVariants`

  controls whether checkpoints for inactive development paths are hidden. `--nohide` shows these checkpoints, while `--hide` hides them. If you do not set this option, checkpoints for inactive development paths are shown.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewproject, si viewsandbox
- Miscellaneous: ACL, diagnostics, options, preferences

si viewprojectmetrics

displays all the metrics for a specified project checkpoint

Synopsis

si viewprojectmetrics [--fields=*field1[:width1],field2[:width2]...*] [--metrics=*value*] [(-P *project*|--project=*project*)] [(-S *sandbox*|--sandbox=*sandbox*)] [--[no]failOnAmbiguousProject] [--projectRevision=*rev*] [--hostname=*server*] [--port=*number*] [--password=*password*] [--user=*name*] [(-?|--usage)] [(-F *file*|--selectionFile=*file*)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=*directory*] [--forceConfirm=*[yes|no]*] [(-g|--gui)] [--quiet] [--settingsUI=*[gui|default]*] [--status=*[none|gui|default]*]

Description

`si viewprojectmetrics` displays all the metrics for the specified project checkpoint. For example:

```
si viewprojectmetrics --project=c:/Aurora_Program/bin/Libra/project.pj --projectRevision=1.2
```

You can track metrics for top-level Integrity projects and calculate them through event triggers. Calculated metrics can be viewed for an Integrity configuration management project or through a computed field on an Integrity Workflows and Documents project issue. Metrics provide information on a project as of a specific checkpoint.

Integrity provides some standard physical metrics that you can use. You can also create your own metrics using a third party tool. The `--projectRevision` option is mandatory. Metrics can only be viewed for a build project..

Note

Metrics are only supported for database type repositories.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--metrics=`*value*

  specifies the metrics to display information for. The `value` is a comma-delimited list of metric names, for example, `lines,functions,bytes`. If you do not specify a value, all metrics are displayed.

  Each metric is displayed on a line by itself, displaying the fields specified by `--fields` If no fields are specified, all fields are displayed except for `description`.

- `--fields=`*field1[:width1],field2[:width2]...*

  allows you to select fields to be printed, specified in the format `field1[:width1],field2[:width2]...`. Specifying the column `[:width]` (in pixels) for each field is optional - widths are only available with the `-g` or `--gui` options. Under the CLI the fields are separated with a `space`.

  The fields available for printing can be one or more of the following:

  - `average`

    displays the average for the metric.
  - `count`

    displays the number of issues that this metric is tracked for.
  - `metric`

    displays the name of the metric.
  - `description`

    displays the description of the metric.
  - `value`

    displays the value of the metric.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands:

  si createmetricinfo, si calculateprojectmetrics, si addprojectmetric, si viewmetricsinfo
- Miscellaneous:

  ACL, diagnostics, options, preferences

si viewrevision

opens a revision for viewing

Synopsis

```
si viewrevision [--[no|un]expand] [(-rrev|--revision=rev)] [(-P project|--project=project)] [(-S sandbox|--sandbox=sandbox)] [--
[no]failOnAmbiguousProject] [--devpath=path] [--projectRevision=rev] [--hostname=server] [--port=number] [--password=password] [--
user=name] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--
forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] member...
```

Description

`si viewrevision` opens a member revision for viewing.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no|un]expand`

  controls whether to expand, unexpand, or ignore keywords in the member file. Keyword expansion is only available in text archives, not binary archives. For descriptions of the Integrity keywords, see the `PTC Integrity User Guide`. Possible keywords are:

  ```
  $Author: Warner, Carrie (cwarner) $
  $CompanyInfo$
  $Date: 2015/11/30 16:23:14EST $
  $Header: si_viewrevision.dita 1.2 2015/11/30 16:23:14EST Warner, Carrie (cwarner) Exp  $
  $Id: si_viewrevision.dita 1.2 2015/11/30 16:23:14EST Warner, Carrie (cwarner) Exp  $
  $Locker:  $
  $Log: si_viewrevision.dita  $
  Revision 1.2 2015/11/30 16:23:14EST Warner, Carrie (cwarner)
  XML tagging fixes
  Revision 1.1 2015/10/29 10:25:05EDT Flett, David (dflett)
  Initial revision
  Member added to project /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj
  $Revision: 1.2 $
  $Name:  $
  $ProjectLabel:  $
  $ProjectName: /rd/doc/Strategic/xmldocs/en/int-man_pages/si_ref/project.pj $
  $ProjectSetting  $
  $ProjectRevision: Last Checkpoint: 1.1.1.8 $
  $RCSfile: si_viewrevision.dita $
  $Revision: 1.2 $
  $SandboxSetting  $
  $Setting  $
  $Source: si_viewrevision.dita $
  $State: Exp $
  ```

- *member...*

  identifies a specific member; use spaces to specify more than one member.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si ci, si co, si edit, si lock, si unlock
- Miscellaneous: ACL, diagnostics, options, preferences

si viewsandbox

displays the contents of a Sandbox

Synopsis

```
si viewsandbox [--[no]includeDropped] [--fields=field1[:width1],field2[:width2]...] [--[no]filterSubs] [--height=value] [--
width=value] [-x value] [-y value] [(-R |--[no|confirm]recurse)] [--filter=filteroptions] [--[no]failOnAmbiguousProject] [(-S
sandbox|--sandbox=sandbox)] [--lockRecordFormat=value] [--hostname=server] [--port=number] [--password=password] [--user=name] [(-?|-
-usage)] [(-F file|--selectionFile=file)] [(-N |--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-g|--
gui)] [--[no]persist] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] current or dropped member/subproject...
```

Description

`si viewsandbox` displays the contents of a Sandbox, for example:

```
c:\Documentation\xml_man\si_about.1.xml archived 1.6
 c:\Documentation\Man_Pages\xml_man\si_acv.1.xml archived 1.7
 c:\Documentation\Man_Pages\xml_man\si_add.1.xml archived 1.7 sueq Jun 21, 2009 - 10:14 AM
 Working file 2,739 bytes smaller, older (Jun 20, 2009 - 1:46:27 PM)
 c:\Documentation\Man_Pages\xml_man\si_addlabel.1.xml archived 1.6 sueq Jun 21, 2009 - 10:14 AM
 Working file 7,344 bytes smaller, older (Jun 20, 2009 - 2:49:28 PM)
 c:\Documentation\Man_Pages\xml_man\si_addmemberattr.1.xml archived 1.4 sueq Jun 21, 2009 - 10:14 AM
 Working file 5,795 bytes smaller, older (Jun 20, 2009 - 3:16:29 PM)
 c:\Documentation\Man_Pages\xml_man\si_addprojectattr.1.xml archived 1.4
 c:\Documentation\Man_Pages\xml_man\si_addprojectlabel.1.xml archived 1.4
 c:\Documentation\Man_Pages\xml_man\si_appendrevdesc.1.xml archived 1.6
 c:\Documentation\Man_Pages\xml_man\si_archiveinfo.1.xml archived 1.5
```

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]includeDropped`

  controls whether to include members dropped from the project that still have working files in your Sandbox. Dropped members will be marked as `dropped`.
- `--fields=field1[:width1],field2[:width2]...`

  allows you to select fields to be printed, specified in the format `field1[:width1],field2[:width2]...`. Specifying the column `[: width]` (in pixels) for each field is optional - widths are only available with the options. Under the CLI the fields are separated with a space>.

  The fields available for printing can be one or more of the following:
  - `archiveshared`

    displays indicators for members that share another member's archive. This column is valid only if you are using the database repository.
  - `attributes`

    displays member and project attributes.
  - `context`

    When used with the Integrity API, displays the name of the project, and indicates if the project is a build or variant.

    Note

    This field is not valid for use with the CLI.

  - `cpid`

    displays the change package associated with the operation that set the member revision.
  - `creationcpid`

    displays the change package that created the revision that is currently the member revision. This revision may be different from the Member CPID if an import, add member from archive, or set member revision operation was used.
  - `deferred`

    displays all deferred operations.
  - `frozen`

    displays indicators for frozen members.
  - `indent`

    indents a field. For example:

    `si viewsandbox --fields=indent, name, type, memberrev`

    indents the `name` field, followed by `type` and `memberrev` fields, indenting as you recurse into subprojects.

    Note

    This field is not valid for use with the Integrity API.

  - `labels`

    displays labels.
  - `lockrecord`

    a comma separated list of locks on the member. The locker and lock type are displayed by default for each lock. You can customize the lock information that displays by using the `--lockRecordFormat` option.
  - `memberarchive`

    display the name of the archive to which the member refers.
  - `memberdescription`

    displays the revision description assigned to the Sandbox member.
  - `memberrev`

    displays the member revision.
  - `membertimestamp`

    displays the date and time the member revision is set.
  - `merge`

    displays any merge information between member revisions.
  - `name`

    displays the member name.
  - `newrevdelta`

    displays indicators for new revisions.
  - `pendingcpid`

    displays the change package associated with a pending operation.

- `revsyncdelta`

  displays an indicator for out of sync members.
- `state`

  displays the member state.
- `symboliclink`

  specifies whether the member is a symbolic link file member.
- `type`

  displays the type of each item in the Sandbox: Sandbox, sub Sandbox, or member.
- `wfdelta`

  displays an indicator when the working file is different from the member revision.
- `workingarchive`

  displays the name of the archive from which the working file is derived.
- `workingcpid`

  displays the change package associated with a deferred or a lock operation performed by the current user from the current Sandbox.
- `workingrev`

  displays the working file revision.

- `--[no]filterSubs`

  controls whether to display sub Sandboxes and directories that do not contain members matching the current filter.
- `--[no]persist`

  controls whether this presentation of information should continue to be updated as new information becomes available. `--nopersist` forces a static "snapshot" of information, while `--persist` gives real-time updates.
- `--lockRecordFormat=`*`value`*

  defines the format for displaying lock information in the `--lockrecord` field. Specify a format string using keywords to represent the information you want to display. You can specify any of the following keywords:
  - `{revision}`

    displays the revision that is locked.
  - `{locker}`

    displays the user who locked the revision.
  - `{locktype}`

    displays the type of lock on the revision (exclusive or non-exclusive).
  - `{locktimestamp}`

    displays the time when the revision was locked.
  -
  - `{lockcpid}`

    displays the change package associated with the lock on the revision.
  - `{project}`

    displays the name and path of the project where the member revision was locked from. If the member revision was locked from a shared subproject, it is the subproject name and path that are displayed.
  - `{devpath}`

    displays the name of the development path where the lock on the revision was made from.
  - `{sandbox}`

    displays the name of the Sandbox where the lock on the revision was made. This is relevant when viewing the information from the locker host.
  - `{hostname}`

    displays the hostname of the computer that locked the the revision.
  - `{hascpid}`

    displays 1 if the lock has a change package associated with it, 0 if there is no associated change package.
  - `{hassandbox}`

    displays 1 if there is Sandbox information available for the lock, 0 if no Sandbox information is available.
  - `{hasdevpath}`

    displays 1 if the lock was made from a development path, 0 if it wasn't.
  - `{member}`

    displays the name of the locked revision.
- *`current or dropped member/subproject...`*

  identifies a specific member or subproject that currently exists in the project, or one that has been dropped from the project. Use spaces to specify more than one.

Diagnostics

See the diagnostics reference page for possible exit status values.

Preferences

Using si setprefs or si viewprefs, you are able to set or view the preference keys for this command.

See Also

- Commands: si archiveinfo, si memberinfo, si mods, si print, si revisioninfo, si rlog, si sandboxinfo, si viewhistory, si viewlabels, si viewproject, si viewprojecthistory
- Miscellaneous: ACL, diagnostics, options, preferences

si viewserveralert

displays Integrity Server alert messages for a target server and all related servers

Synopsis

`si viewserveralert [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [(-Ffile|--selectionFile=file)] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]]`

Description

`si viewserveralert` displays Integrity Server alert messages for a target sever and all related servers, for example, a configuration management server and a proxy server. The alert message displays who sent the message, the server it came from, when it was sent, and the message. Alert messages are sent by your administrator and are useful for notifying users about important information, such as an impending server upgrade in which the server will be shut down.

- In the Web interface, the date displayed for an alert message is the server's date, time, and time zone. In the GUI and CLI, the date displayed for an alert message is the client's date, time, and time zone.
- To avoid manually checking alert messages from the command line, launch the alert messages dialog box from the command line by specifying `-g` or `--gui` and keep the dialog box open. The dialog box automatically refreshes to display new alert messages.
- A connection to the target server is required for the `si viewserveralert` command to display alert messages.

Options

This command takes the universal options available to all `si` commands, as well as some general options. See the options reference page for descriptions.

- `--hostname=`*`server`* specifies the host name of the target Integrity Server to retrieve alert messages from. If this option is not specified, the default server is used.
- `--port=`*`number`* specifies the port of the target Integrity Server to retrieve alert messages from.

See Also

- Commands: si serveralerts, si admingui, si servers
- Miscellaneous: options

tm createresult

creates test results for the specified test cases

Synopsis

```
tm createresult [--sessionID=value] [--addAttachment=value] [--addRelatedItem=[FieldName]:ItemID[relationshipFlags][,...]] [--
verdict=value] [--stepVerdict=stepID=value[:verdict=value][:annotation=value]] [--annotation=value] [--[no]forceEdit] [--
hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--
quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--
forceConfirm=[yes|no]] test id...
```

Description

This command allows you to record the same test results for a group of test cases in a test session. The test results are created with the same result, annotation, attachments and related items.

For example,

```
im createresult --sessionID=8017 --verdict=Passed --annotation="Works as designed" 9075 9025 9001
```

creates test results with a verict of `Passed` and a annotation of `Works as designed` for test cases 9075, 9025 and 9001 in test session 8017.

This command reports status (progress) from the server and you can cancel the operation. If you cancel the command, no test results are created.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the `options` reference page for descriptions.

- `--sessionID=`*value*

  the ID of the test session that the test cases belong to. This option is mandatory.

  Note
  The test session must be in a state that allows test results to be created, and the test result policy for the session must allow the user to modify test results.

- `--addAttachment=`*value*

  adds attachments, where `value` is of the form "*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]* " .

  Note
  The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

  For example,

```
addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
```

  adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

  If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. A test result cannot have more than one attachment with the same name.

  Note
  Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

`--addRelatedItem=[`*FieldName*`]:`*ItemID[relationshipFlags*`][,...]`
the related items to add to the test results for the specified test cases.

If no field name is specified, the `Forward Relationships` field is used. You can use this option multiple times to specify multiple related items.

Related items are created as a result of test case failure.

Note
Adding a related item is only permitted if your administrator has allowed relationships for the item type.

`--verdict=`*value*the outcome of the test cases, for example, `passed`, `failed`, or `skipped`. Contact your administrator for the available values for this option.

Note
If no verdict is specified, the pre-definedunspecified verdict is used.

`--stepVerdict`=stepID=*value*[:verdict=*value*][:annotation=*value*]

the outcome of the test case steps, for example, `passed`, `failed`, or `skipped`. Contact your administrator for the available values for this option. If required, you can also specify an annotation for the test case step.

If you are creating results for more than one test step, you must specify the ID of the test step or the command will fail.

You can use this option multiple times to specify the verdict and annotation for multiple steps.

Note
If an annotation is specified but no verdict is specified, the pre-defined `unspecified` verdict is used.

`--annotation=`*value*

notes about the test results, for example, reasons for test failure.

Note
Annotations are limited to 4000 characters.

`--[no]forceEdit`

if the test result being created already exists, edits it based on the new result. Forced edit should be used when automatically creating test results using the `setresults` command.

`test id...`

specifies the test case IDs to create test results for. The test cases must all belong to the session specified in the `--sessionID` option. If you do not specify test case IDs, the results are recorded for all test cases in all test suites for the specified session. If there is a test case that already had a result in the specified session, the command will fail.

See Also

- Commands: tm results, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm createverdict

creates a test verdict

Synopsis

`tm createverdict [--name=value] [--image=[none|default|<path>] [--description=value] [--position=[<number>|first|last|before:<name>|after:<name>] [--verdicttype=[pass|fail|other] [--password=password] [--user=value] [--hostname=value] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`createverdict` creates an Integrity test verdict. For example:

```
tm createverdict --hostname=abcFinancial --user=jriley --verdicttype=failure\u00a0--name="Does Not Match Requirement"
```

Options

This command takes the universal options available to all CLI commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the name of the verdict. Names can be a maximum of 100 characters and cannot contain square brackets. This option is mandatory.
- `--image=[none|default|<path>]`

  specifies whether an image displays for the verdict.

  `--image=none` does not display an image for the verdict.

  `--image=default` displays the default image appears for the verdict.

  `--image=<path>` specifies the path and name of a custom image for the verdict, for example, `c:\images\fail_verdict_icon.gif`.

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.
- `--description=value`

  specifies a description of the verdict.
- `--position=[<number>|first|last|before:<name>|after:<name>]`

  specifies the position in the list of verdicts.
- `--verdicttype=[pass|fail|other]`

  specifies the type of test outcome that the verdict describes.

See Also

- Commands: tm editverdict, tm deleteverdict, tm viewverdict, tm verdicts
- Miscellaneous: options

tm deleteresult

deletes test results for the specified test cases

Synopsis

```
tm deleteresult [--sessionID= value] [--height= value] [--width= value][--x= value] [--y= value] [--hostname= value] [--port=
value] [--password= value][--user= value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile= value)][--quiet] [--settingsUI=
[gui|default]] [--status=[none|gui|default]][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd= value][--forceConfirm=
[yes|no]]caseID|sessionID:caseID|sessionID:caseID:stepID...
```

Description

This command deletes test results for test cases.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=value`

  deletes all test results in the specified test session. If you use this option, do not specify test results using the `caseID|sessionID:caseID`.

- *caseID|sessionID:caseID|sessionID:caseID:stepID...*

  specifies the test case IDs to delete test results for. The test cases must all belong to the session specified in the `--sessionID` option. If the session is not specified in the `--sessionID` option, you must specify the session ID and case ID for each test case you want to delete test results for. You can also specify individual test steps in a test case to delete results for. If you do not specify this option, you must specify the `--sessionID` option, and results are deleted for all results in all test cases for that session.

Note

The test session must be in a state that allows test results to be modified, and the test result policy for the session must allow the user to modify test results.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm results, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm deleteverdict

deletes a test verdict in Integrity

Synopsis

`tm deleteverdict [--[no]confirm] [--user=`*`name`*`] [--hostname=`*`server`*`] [--password=`*`password`*`] [--port=`*`number`*`] [(-?|--usage)] [(-F`*`file`*`|--selectionFile=`*`file`*`)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`directory`*`] [--forceConfirm=`*`[yes|no]`*`] [-g|--gui] [--quiet] [--settingsUI=`*`[gui|default]`*`] [--status=`*`[none|gui|default]`*`] `*`test verdict...`

Description

`tm deleteverdict` deletes an Integrity test verdict.

You cannot delete the default test verdicts that are shipped with Integrity.

Options

This command takes the universal options available to all CLI commands, as well as some general options. See the options reference page for descriptions.

- `--[no]confirm`

  specifies if to display a confirmer when deleting a verdict.
- *`test verdict`*

  specifies the name of the verdict to delete.

See Also

- Commands: tm createverdict, tm editverdict, tm viewverdict, tm verdicts
- Miscellaneous: options

tm editresult

edits test results for the specified test cases

Synopsis

```
tm editresult [--sessionID=value] [--addAttachment=value] [--removeAttachment=value] [--addRelatedItem=
[FieldName]:ItemID[relationshipFlags][,...]] [--removeRelatedItem==value] [--verdict=value] [--
stepVerdict=stepID=value[:verdict=value][:annotation=value]] [--deleteStepResult=stepID=value [--annotation=value] [--
[no]forceCreate] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--
selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--
cwd=value] [--forceConfirm=[yes|no]] caseID|sessionID:caseID...
```

Description

This command allows you to modify the test results for a group of test cases in a test session. You can edit the result, annotation, attachments, test steps, and related items for the group of specified test cases.

For example,

```
tm editresult --verdict=Failed --sessionID=8071 9023
```

edits test case 9023 in test session 8071 to give it a verdict of `Failed`.

Options

This command takes the universal options available to `tm` commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=value`

  the ID of the test session item that the test cases belong to.

  Note
  The test session must be in a state that allows test results to be modified, and the test result policy for the test session must allow the user to modify test results.

- `--addAttachment=value`

  adds attachments, where *value* is of the form "*field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]* " .

  Note
  The "*pathToFile*" must include the path and filename. The "*nameOfAttachment*" is optional, and gives the attachment a different name than the name of the file specified in "*pathToFile*".

  For example,

  ```
  addAttachment="field=Attachments,path=c:/temp/notes.txt,name=notes123.txt,summary="Notes for issue 123""
  ```

  adds the existing `notes.txt` file as an attachment with the name of `notes123.txt`.

  If you do not specify the name of the attachment, the file name is used as the attachment name. You can use this option multiple times to specify multiple attachments. A test result cannot have more than one attachment with the same name.

Note
Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

`--removeAttachment=field=fieldName,path=pathToFile[,name=nameOfAttachment][,summary=shortDescription]`

removes attachments from the test results for the specified test cases, where *value* is of the form "[*field=fieldName,name=nameOfAttachment*]".If no attachment field is specified the default `Attachments` field is used.

You can use this option multiple times to specify multiple attachments to remove. If none of the specified test cases have the attachment, the command fails.

Note
Attachment size limits are set by your administrator. The default attachment size limit is 4 MB.

`--addRelatedItem=[FieldName]:ItemID[relationshipFlags][,...]`

the related items to add to the test results for the specified test cases.

If no field name is specified, the `Forward Relationships` field is used. You can use this option multiple times to specify multiple related items.

Related items are created as a result of test case failure.

Note
Adding a related item is only permitted if your administrator has allowed relationships for the item type.

`--removeRelatedItem=[FieldName]:ItemID[relationshipFlags][,...]`

removes related issues from the test results for the specific test cases, where `value` is of the form [`fieldName`:]`id`[,...]. If no field name is specified, the `Forward Relationships` field is used.

You can use this option multiple times to specify multiple related items to remove. If none of the specified test cases have the related item, the command fails.

`--verdict=value`

the outcome of the test cases, for example, `passed`, `failed`, or `skipped`. Contact your administrator for the available values for this option.

`--stepVerdict=stepID=value[:verdict=value][:annotation=value]`

the outcome of the test case steps, for example, `passed`, `failed`, or `skipped`. Contact your administrator for the available values for this option. You can also add or edit an annotation for the test case step.

If you are editing results for more than one test case, you must specify the ID of the test step or the command will fail.

You can use this option multiple times to edit the verdict and annotation for multiple steps.

Note
You can add or edit a test step annotation without editing the test step verdict.

`--deleteStepResult=stepID=`*`value`*

the test step to remove from the test results for the specified test cases.

You can use this option multiple times to specify multiple steps to remove. If none of the specified test cases have the step, the command fails.

`--annotation=`*`value`*

notes about the test results, for example, reasons for test failure.

`--[no]forceCreate`

forces the creation of the test result being edited if it doesn't exist. Forced create should be used when automatically editing test results using the `setresults` command.

`caseID|sessionID:caseID...`

specifies the test case IDs to edit test results for. The test cases must all belong to the session specified in the `--sessionID` option. If the session is not specified in the `--sessionID` option, you must specify the session ID and case ID for each test case. If you do not specify this option, you must specify the `--sessionID` option, and results are edited for all test cases in all test suites for that session.

Note

The test session must be in a state that allows test results to be modified, and the test result policy for the session must allow the user to modify test results.

See Also

- Commands: tm createresult, tm viewresult, tm extractattachments, tm deleteresult, tm results, tm stepsresults, tm resulteditor, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm editverdict

edits an test verdict

Synopsis

`tm editverdict [--name=value] [--image=[none|default|<path>] [--description=value] [--position=[<number>|first|last|before:<name>|after:<name>] [--verdicttype=[pass|fail|other] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] test verdict`

Description

`tm editverdict` edits the properties of a test verdict.

Options

This command takes the universal options available to all CLI commands, as well as some general options. See the options reference page for descriptions.

- `--name=value`

  specifies the name of the verdict. Names can be a maximum of 100 characters and cannot contain square brackets.
- `--image=[none|default|<path>]`

  specifies whether an image displays for the verdict.

  `--image=none` does not display an image for the verdict.

  `--image=default` displays the default image appears for the verdict.

  `--image=<path>` specifies the path and name of a custom image for the verdict, for example, `c:\images\fail_verdict_icon.gif`

  Note

  Images must be GIF or JPEG format, and no larger than 16 by 24 pixels.

- `--description=value`

  specifies a description of the verdict.
- `--position=[<number>|first|last|before:<name>|after:<name>]`

  specifies the position in the list of verdicts.
- `--verdicttype=[pass|fail|other]`

  specifies the type of test outcome that the verdict describes.
- *test verdict*

  specifies the name of the verdict you want to edict. Verdict names are case sensitive.
-

See Also

- Commands: tm createverdict, tm deleteverdict, tm viewverdict, tm verdicts
- Miscellaneous: options

tm extractattachments

saves an attachment from a test case result

Synopsis

`tm extractattachments [--resultID=`*`sessionID`*`:`*`caseID`*`] [--outputFile=`*`value`*`] [--[no|confirm]overwriteExisting] [--hostname=`*`value`*`] [--port=`*`value`*`] [--password=`*`value`*`] [--user=`*`value`*`] [(-?|--usage)] [(-g|--gui)] [(-F `*`value`*`|--selectionFile=`*`value`*`)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=`*`value`*`] [--forceConfirm=[yes|no]] `*`attachment`*

Description

`tm extractattachments` saves an attachment from a test case result for later viewing or printing.

Options

This command takes the universal options available to `tm` commands, as well as some general options. See the `options` reference page for descriptions.

- `--resultID=`*`sessionID`*`:`*`caseID`*

  the ID of the test session and test case that you want to extract result attachments from.
- `--outputFile=`*`value`*

  specifies the name of the file that the attachment is extracted to. If not specified, the name of the attachment is used as the output file.
- *`attachment`*

  specifies the name of the attachment to extract, for example, `test_spec.htm`. If no attachment is specified, then all attachments for the result are extracted.

Note

If `--cwd` is not specified, the attachment is saved to the current working directory.

See Also

- Commands: tm createresult, tm editresult, tm extractattachments, tm results, tm deleteresult, tm resulteditor, tm stepsresults, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm purgeresults

purges test results for test cases

Synopsis

```
tm purgeresult [--before=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F
value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--
[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] sessionID...
```

Description

This command purges test results for the test cases in a test session. You can specify the test session to purge results for, or purge results for all test sessions that were created before a specified date.

For example,

```
tm purgeresults --before=01/01/2008
```

purges all test results created before January 1, 2008.

When a test result is deleted, its attachments and test steps are also deleted.

This command confirms all actions by displaying the number of results and number of sessions to be purged. Once the command is started, it cannot be cancelled.

Note

You must have the `PurgeTestResult` permission to run this command.

Options

This command takes the universal options available to `tm` commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=`*value*

  the ID of the test session that the test cases belong to. You must specify either this option or the `--before` option.

  Note

  The test result policy for the test session must allow the user to modify test results.

- `--before=`*value*

  deletes all test results in all test sessions that were created before the specified date, where *value* is of the form:
  MM/dd/yyy h:mm:ss [AM|PM]

  See the *:time:timestamp* option in options for additional date formats.

  Note

  In order to use this option, you cannot use the `--sessionID` option or specify any test cases using *test id...*.

- *sessionID...*

  the ID of the test session that you want to purge test results for. You must specify either a session or the `--before` option.

  Note

  The test session must be in a state that allows test results to be modified, and the test result policy for the session must allow the user to modify test results.

See Also

- Miscellaneous: options

tm resulteditor

allows you to manage the results of a test session

Synopsis

```
tm resulteditor [--[no]applyDisplayPattern] [--[no]showOutline[--[no]showDetails [--[no]splitVertical [--[no]displayOutline][--asOf=
[<date>|label:<label>][--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...][--expandRelationshipDirection=
[forward|backward|both]][--expandRelationshipFields=field,field,...] [--[no]batch][--[no]showFieldNodes] [--
structureFieldDisplayFormat=field][--structureFieldIconDisplayField=value] [--query=[user:]query][--[no]showXHTML] [--
expandLevel=value] [--focusIssueID=value][--height=value] [--width=value] [-x value] [-y value] [--hostname=value][--port=value] [--
password=value] [--user=value] [(-g|--gui)] [(-?|--usage)][(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]][-
-status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--cwd=value][--forceConfirm=[yes|no]] issue id...
```

Description

The `tm resulteditor` command displays an editor for working with test results for a test session. You can expand the tree to show test cases for the root issue(s) and enter or change the test result for the test cases.

For example,

```
tm resulteditor --splitVertical --fields=ID,summary 80713
```

displays the `ID` and `summary` for all test cases contained in test session 80713. The editor displays both the Outline and details panes.

This command is only supported with the `-g` or `--gui` option.

Options

This command takes the universal options available to `tm` commands, as well as some general options. See the options reference page for descriptions.

- `--[no]applyDisplayPattern`

  specifies whether to apply a display pattern to numeric fields. Display patterns are configured by your administrator and allow you to quantify integer and floating point field values, for example, as currency or percentages. `--applyDisplayPattern` is enabled by default.

Note

If you use scripts, PTC recommends using the `--noapplyDisplayPattern` option to avoid being impacted by administrative changes to display patterns.

- `--[no]displayOutline`

  specifies whether to display the Outline, a tree-like relationship hierarchy stemming from the test session you used to launch the view.
- `--[no]displayDetails`

  specifies whether to display the details of the test case selected in the Outline.
- `--[no]splitVertical`

  specifies whether to split the test editor vertically, displaying the Outline on the left side and the test case details on the right side.
- `--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...`

  allows you to select fields to be displayed in the test result editor, specified in the format *`field[:width],field[:width],...`*. Specifying the column *`[:width]`* (in pixels) for each field is optional.

  For rich content fields, you can specify display patterns to display field values as rich content (`:rich`) or plain text (`:plain`). The rich content display pattern displays the underlying HTML elements and attributes in the rich content field. For example, `--fields=Description::rich` displays the Description field value with HTML elements and attributes. By default, rich content fields display plain text.
- `--expandRelationshipDirection=[forward|backward|both]`

  specifies the type of relationship fields to display and expand: forward, backward, or both.
- `--expandRelationshipFields=field,field,...`

  specifies the relationship fields to display and expand.

  Note the following:

  - Relationship fields include standard forward and backward relationship fields, as well as query backed relationship (QBR) and item backed picklist (IBPL) fields (considered as forward relationship fields).
  - The `--expandRelationshipDirection` and `--expandRelationshipFields` options are mutually exclusive. If both options are specified, the `im relationships` command fails and displays an error message with the reason.
- `--asOf=[<date>|label:<label>]`

  allows you to view test cases as of a historical date or label. For example, to view test cases in a test session as of a specific date, type

  ```
  tm resulteditor --asOf="January 8, 2007 10:00:00 AM EST" 123
  ```

  If no value is provided the test cases display as of the date in the test session field specified by your administrator, or as of the server's current time if no test session date is specified. This field is optional.
- `--structureFieldDisplayFormat=value`

  specifies the fields and style that should be displayed for the tree nodes.

  The default formatting is suitable for interpretation by most users; the various formatting options are provided for programmatic control.

  `--structureFieldDisplayFormat` options use the same values as `--fields`, but similar to a JAVA MessageFormat string (that is, it requires `{ }` to surround each field). For example:

  ```
  tm resulteditor --structureFieldDisplayFormat="{ID},{Summary}"
  ```
- `--structureFieldIconDisplayField=value`

  specifies the field from which the icon is taken. This will specify the icon that will be displayed for each node in the Outline pane.
- `--[no]showXHTML`

  specifies whether rich text fields display in XHTML.
- `--expandLevel=value`

  specifies to expand the nodes to a specified level, for example, 1, 2, 3. The default is 1.
- `--focusIssueID=value`

  specifies the issue ID to focus on in the editor.
- `--query=[user:]query`specifies the query to use to populate the issue selection. If not specified, Integrity uses the most recently run query.
- *`issue id...`*

  specifies the ID of the test session issue(s) you want to manage results for. Use spaces to specify more than one issue, for example `34 23`.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm results, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm resultfields

displays a list of administrator-defined custom test result fields

Synopsis

`tm resultfields [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--[no]asAdmin] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] field...`

Description

`tm resultfields` displays a list of defined test result fields. By default, all fields are selected to be displayed.

Options

This command takes the universal options available to all `im` commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  where field`n` can be any of the following:

  - `allowedTypes`

    displays the issue types that can be linked to the relationship field.
  - `cycleDetection`

    displays whether or not the system will prevent relationship loops from occurring in the relationship field.
  - `name`

    displays the name of the field. By default, only the name is shown.
  - `displayName`

    displays the name assigned as the display name of the field.
  - `pairedField`

    displays the field name of the paired relationship field. For forward relationship fields, the corresponding backward relationship field displays; for backward relationship fields the corresponding forward relationship field displays.
  - `paramSubstitution`

    displays whether or not parameter references in the text field are replaced with parameter values when you view the item through a view or report that supports parameter substitution. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
  - `isMultiValued`

    displays whether or not the field can contain multiple values.
  - `description`

    displays a description of the field.
  - `type`

    displays the field data type.
  - `default`

    displays default value for the field (CLI only).
  - `defaultAttachmentField`

    displays default attachment field for the rich content field.
  - `defaultBrowseQuery`

    displays the default Admin query to use when adding a related item by browsing to the relationship field.
  - `displayAsLink`

    displays whether or not the selected value in the item backed pick list field displays as a hyperlink to the backing item in the GUI and the Web UI.
  - `min`

    displays the minimum value for the field for integer, float, and date fields (CLI only).
  - `max`

    displays the maximum value for the field for integer, float, and date fields (CLI only).
  - `displayAsProgress`

    displays whether or not the value of the integer field is displayed as progress bar in the GUI.
  - `displayLocation`

    displays the name of the issue tab where the relationship field is displayed (field or relationship).
  - `picks`

    displays the list of valid values for a pick list field, of the form *`text:value:image`* (CLI only).
  - `suggestions`

    displays the list of suggested values for a short text field (CLI only).
  - `textindex`

    displays whether or not queries against the field will be treated as word searches.
  - `trace`

    displays whether or not the field is a trace relationship. Trace relationships are defined via field pairs and are presented to the user in domain-specific language, for example, Test and Requirements. To learn more about trace relationships, see the *PTC Integrity User Guide*.
  - `relevanceRule`

    displays the relevance rule for the field (CLI only).
  - `editabilityRule`

    displays the editability rule for the field (CLI only).
  - `id`

    displays the database ID of the field. This is for PTC - Integrity Support only.
  - `isForward`

    displays whether or not the field is a forward relationship field.
  - `linkFlags`

    displays relationship flags for the relationship field.
  - `maxLength`

    displays the maximum length of a long or short text field (CLI only).
  - `displayRows`

    displays the number of rows used for a long text field or a relationship field.
  - `displayStyle`

    displays the format used for the field: table or csv (comma separated values).
  - `loggingText`

displays the logging type of field. Valid for long text fields only (CLI only).

- `position`

  displays the position in the list of fields.
- `computation`

  displays the expression for the computed field.
- `staticComputation`

  displays whether the computed field is a static or dynamic computation.
- `storeToHistoryFrequency`

  displays the frequency at which the computed field is calculated and stored to issue history.
- `lastcompute`

  displays the date and time that the computed field was last calculated.
- `references`

  displays a list of object references.
- `associatedField`

  displays the field associated with a field value attribute.
- `backedBy`

  displays the issue backed picklist that backs the field value attribute.
- `backingStates`

  displays the active states that back the issue backed picklist.
- `backingTextField`

  displays the short text field containing text for an issue backed picklist.
- `backingTextFormat`

  displays the fields containing values that are linked together to form the picklist values for an issue backed picklist.
- `backingType`

  displays the type that backs an issue backed picklist.
- `backingFilter`

  displays the filter applied to values in an issue backed picklist.
- `correlation`

  displays the field contained in the type containing the query backed relationship field and a field in the issues returned by the query.
- `defaultColumns`

  displays the default columns for the relationship field.
- `displayPattern`

  displays the display pattern for a numeric field.
- `phases`

  displays the phases for a phase field.
- `query`

  displays the query for the query backed relationship field.
- `ranges`

  displays the ranges for the range field.
- `richContent`

  displays whether the field supports rich content and the specified screen and printer CSS files.
- `sortIBPLDescending`

  displays the sort order of the field on the backing item type in the IBPL. This is only displayed when a sort field and a direction have been set on an IBPL field.
- `sortIBPLField`

  displays the field on the backing item to sort by. This is only displayed when a sort field and a direction have been set on an IBPL field.

- `--[no]asAdmin`

  allows non-admin access to test result fields. This option is used specifically for third party integrations using the Integrity API. The default setting is `--asAdmin` which requires ViewAdmin permissions. Specifying `--noasAdmin` prevents visibility to fields that give information about other users. This option overrides the fields specified using the `--fields` option. Accessible and non-accessible fields are defined by the system for this command.

  To learn more about the API, see the *PTC Integrity Integrations Builder Guide*.
- `--fieldsDelim=`*`value`*

  specifies the string to be used as a delimiter between test result fields displayed in the CLI.
- *`field...`*

  specifies the name of the test result fields to display.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm results

displays test results for test cases

Synopsis

```
tm results [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value][--relatedToItemID=value] [--[no]sortAscending][--[no]showSharedResults] [--[no]lastResult] [--sortField=field]--verdictFilter=value --filter=value [--suiteID=value][--sessionID=value] [--caseID=value] [--height=value] [--width=value] [--x=value] [--y=value] [--hostname=value] [--filter=value] [--port=value][--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)][(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]][--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch][--cwd=value] [--forceConfirm=[yes|no]] caseID|sessionID:caseID...
```

Description

This command displays test results for test cases in a list format. You can specify which test result field values display. Attachments and related items display as a comma separated list of attachment names and related item IDs.

For example:

```
tm results --fields=State:case --sessionID=25 --filter="(testresult[TMDSAMPLEFIELD_Float1]=20.001)" --hostname=localhost --port=7001 --user=test1 --password=xx xxxxxx
```

Note

The test session item type and the test case item type must be visible to you in order for you to view the test results.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  the test result fields, and their respective widths, to be displayed. Fields can be any of `caseID`, `result`, `annotation`, `hasAttachment`, `hasRelatedItem`, `hasStepResult`, `modifiedDate`, `modifiedUser`, `sessionID`, `sharedCaseID`, `verdict`, `verdictIcon`, `verdictType`, or `verdictTypeIcon`.

  - Use commas to specify more than one field.
  - `TestCase` and TestSession item fields need to be annotated respectively, for example, `State:case`, `Summary:session`.
- `--fieldsDelim=value`

  the string to be used as a delimiter between the fields in the display.

- `--relatedToItemID=value`

  displays test results that are related to the specified item. If you use this option, do not specify test results using the `--sessionID`, `--caseID`, or `--suiteID` option or `sessionID:caseID`.

- `--verdictFilter=value`

  filters the results based on their verdict, for example, passed or failed.

- `--sortField=fields`specifies the field to sort results by, for example, result. You can sort by any of the following fields: caseID, result, annotation, hasAttachment, hasRelatedItem, hasStepResult, modifiedDate, modifiedUser, sessionID, sharedCaseID, verdictIcon, verdictType, or verdictTypeIcon.The ability to sort by Item fields, for example, Test Case and Test Session is not currently supported.

- `--[no]sortAscending`

  specifies whether to sort the specified sort field in ascending or descending order.

- `--[no]showSharedResults`

  if viewing results for a test case that is related to multiple test group documents, specifies whether to include results entered against the group.

- `--[no]lastResult`

  if viewing results for multiple tet sessions, specifies whether to only show the latest result.

- `--filter=value`

  specifies a string for the filtering of test results.

  The filter must be in the following format:

  <rule>is defined as (<*filtergroup*>)

  (<*filtergroup*> and <*filtergroup*> and ...)

  (<*filtergroup*> or <*filtergroup*> or ...)

  ((<*filter*>) and (<*filter*>) and ...)

  ((<*filter*>) or (<*filter*>) or ...)

  where

  <*filter*> is defined as disabled (<*filter*>)

  <*filter*> is defined as not (<*filter*>)

  <*filter*> is defined as <*testresult[ID].hasrelateditem*>

  <*testresult*> is defined as *testresult[ID]comparison* where `ID` can be one of `Verdict`, `Verdict Type`, `Modified By`, `Modified Date`,`Session ID`,`Annotation`, `Session ID`, or any test result fields defined by the administrator.

Note

- If the user provides both the `--verdictFilter` and `--filter` option an error will be presented indicating that only one of the options can be used.
- The ability to filter by Item fields, for example, Test Case and Test Session is not currently supported.

- `--suiteID=value`

  displays all test results in the specified test suite for the session specified in `--sessionID`. If you use this option, do not specify test results using the `--sessionID`, `--caseID` or `relatedToItemID` options, or `caseID|sessionID:caseID`.

- `--sessionID=value`

  displays all test results in the specified test session. If you use this option, do not specify test results using the `--suiteID`, `--caseID` or `relatedToItemID` options, or `caseID|sessionID:caseID`.

- `--caseID=value`

  displays test results for the specified test case. If you use this option, do not specify test results using the `--suiteID`, `--sessionID` or `relatedToItemID` options, or `caseID|sessionID:caseID`.

- *caseID|sessionID:caseID...*

  specifies the test case IDs to view test results for. The test cases must all belong to the session specified in the `--sessionID` option. If the session is not specified in the `--sessionID` option, you must specify the session ID and case ID for each test case. If you do not specify this option, you must specify the `--sessionID` option, and results are displayed for all test cases in that session.

Note

The test session item type and the test case item type must be visible to you in order for you to view the test results.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm setprefs

sets preferences for test management commands

Synopsis

`tm setprefs [--command=value] [--[no]resetToDefault] [--[no]save] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-F value|--selectionFile=value)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] string...`

Description

This command sets preference options for Integrity test management commands. These settings are used to determine default behaviors for other test management commands - each command option has a preference key associated with it. The `tm viewprefs` command lists the commands and preference keys. Changes to your preferences are either for the current client session (until `im exit` is used) or can be permanently saved in your system's `home` directory, in the file named `IntegrityClient.rc`, using the `--save` option.

For example:

```
tm setprefs --command=resulteditor --save substituteParameters=true
```

sets the preferences for the `tm resulteditor` command to always substitute parameters in the test result editor.

Note

Do not edit the `IntegrityClient.rc` file manually. Preferences that appear more than once in the `IntegrityClient.rc` file can cause unpredictable behavior in Integrity.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the `options` reference page for descriptions.

- `--command=`*value*

  identifies the command to be set.
- `--[no]resetToDefault`

  controls whether to revert specified settings to the default values as shipped with Integrity Client. If specifying `--resetToDefault`, you must not specify individual preferences.
- `--[no]save`

  controls whether changes should be permanently saved.
- `--[no]ask`

  controls prompts to the user for specific preferences. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:
  - `once`

    asks the user the first time only, and then uses the provided value every time after.
  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).
  - `element-last`

    asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.
- `--ui=[unspecified|gui|cli|api`

  controls whether to apply the preference to the graphical user interface, the command line interface, or when the interface is unspecified. By default, `--ui=cli` is implied when using `tm setprefs`. To set preferences for GUI behavior, however, you should specify `--ui=gui`. For example, to set the `substituteParameters` preference to be true in the GUI for the `tm resulteditor` command, you would type:

  ```
  tm setprefs --command=resulteditor --ui=gui substituteParameters=true
  ```

  These correlate to settings in the`IntegrityClient.rc` file that have the `gui.tm.` or `cli.tm.` prefix, or the `tm.` prefix when it is unspecified.
- *string...*

  identifies the preference string. If you specified the `--resetToDefault` option, then you only need to specify the preference name; otherwise specify a value for the preference. Use spaces to specify multiple preferences.

See Also

- Commands: tm viewprefs. tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm results, tm viewuntested
- Miscellaneous: options

tm setresults

creates or edits test results for a test session

Synopsis

```
tm setresults [--sessionID=value] [--actionDefinition=value] [--actionDefinitionFile=value] [--hostname=value] [--port=value] [--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]]
```

Description

This command creates or edits test results for a test session based on an action definition string or file. It is intended to be used for test automation or integrations with third party automated testing tools.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=value`

  the ID of the test session item that the test cases belong to. The session ID must be specified either in this option or in the XML for `--actionDefinition`.

Note

The test session must be in a state that allows test results to be created, and the test result policy for the session must allow the user to modify test results.

- `--actionDefinition=value`

  the complete action definition for a single operation in formatted XML that conforms to a DTD. Use either this option or the `--actionDefinitionFile` option to define the test result operation to be performed.

  The XML must conform to the following DTD:

```
<?xml version='1.0' encoding='UTF-8'?>
<!ELEMENT ActionList (SetResult)*>

<!ATTLIST ActionList

sessionID CDATA #IMPLIED>

<!ELEMENT SetResult (Verdict?,Annotation?,
(AddRelatedItem*|RemoveRelatedItem*|AddAttachment*|RemoveAttachment*|SetStepResult*)*)>
<!ATTLIST SetResult
caseID CDATA #REQUIRED>

<!ELEMENT Annotation (#PCDATA)>

<!ELEMENT AddAttachment (Description)?>
<!ATTLIST AddAttachment
name CDATA #IMPLIED
file CDATA #REQUIRED

<!ELEMENT RemoveAttachment EMPTY>
<!ATTLIST RemoveAttachment
name CDATA #REQUIRED>

<!ELEMENT Description (#PCDATA)

AddRelatedItemEMPTY>
<!ATTLIST AddRelatedItem
id CDATA #REQUIRED>

<!ELEMENT RemoveRelatedItem EMPTY>
<!ATTLIST RemoveRelatedItem
id CDATA #REQUIRED>

<!ELEMENT SetStepResult (Result?,Annotation?)>
<!ATTLIST SetStepResult
stepID CDATA #REQUIRED>

<!ELEMENT Verdict(#PCDATA)>
```

  Example of an action definition XML file:

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE ActionList SYSTEM "ActionList.dtd">
<ActionList sessionID="1">
<SetResult caseID="1">
<Verdict>
failed
</Verdict>
<Annotation> result annotation </Annotation>
<AddAttachment file="path/attachment1.gif" name="attachmentName">
<Description> attachment description </Description>
</AddAttachment>
<AddRelatedItem id="12"/>
<AddRelatedItem id="13"/>
<SetStepResult stepID="1">
<Verdict>
passed
</Verdict>
<Annotation>
this is step annotation
</Annotation>
</SetStepResult>
<SetStepResult stepID="2">
<Verdict>
failed
</Verdict>
<Annotation>
this is step annotation
</Annotation>
</SetStepResult>
</SetResult>

<SetResult caseID="2">
<Verdict>
passed
</Verdict>
<Annotation> annotation </Annotation>
<SetStepResult stepID="4">
<Verdict>
passed
</Verdict>
<Annotation>
```

```
This is step annotation
</Annotation>
</SetStepResult>
<AddAttachment file="path/attachment2.gif" name="attachment2Name">
<Description> attachment description </Description>
</AddAttachment>
<RemoveAttachment name="screenshot2.jpeg"/>
<AddRelatedItem id="14"/>
<RemoveRelatedItem id="15"/>
</SetResult>
</ActionList>
```

- `--actionDefinitionFile=`*value*

  the path to the action definition file. The action definition file can be an XML file only, or a zip file containing an XML file. If it is contained in a zip file, this command will automatically unzip the file in a temporary folder, look in the unzipped folder for the file with the same name as the zip file but with an `xml` extension, and use that as the action definition file. See the `--actionDefinitionFile` option for the correct format for the file.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm results, tm viewuntested
- Miscellaneous: options

tm stepresults

displays test results for test case steps

Synopsis

```
tm stepresults [--fields=field1[:width1],field2[:width2]...][--fieldsDelim=value] [--sessionID=value] [--height=value][--
width=value] [--x=value] [--y=value] [--hostname=value] [--port=value][--password=value] [--user=value] [(-?|--usage)] [(-g|--gui)]
[(-F value|--selectionFile=value)] [--quiet] [--settingsUI=[gui|default]][--status=[none|gui|default]] [(-N|--no)] [(-Y|--yes)] [--
[no]batch][--cwd=value] [--forceConfirm=[yes|no]]caseID|sessionID:caseID|sessionID:caseID:stepID...
```

Description

This command displays test results for test case steps in a list format. You can specify which test result field values display.

For example,

```
tm stepresults --fields=stepID,verdict,annotation --sessionID=8071
```

displays the test step ID, verdict and annotation for the results for all test steps in all test cases in test session 8071.

Note

The test session item type, test case item type, and the test step item type must be visible to you in order for you to view the test step results.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  the test step result fields, and their respective widths, to be displayed. Fields can be any of `caseID`, `stepID`, `sessionID`, `verdict`, `annotation`, or `verdictType`. Use commas to specify more than one field.
- `--fieldsDelim=value`

  the string to be used as a delimiter between the fields in the display.
- `--sessionID=value`

  displays all test results in the specified test session. If you use this option, do not specify test results using the `--suiteID`, `--caseID` or `relatedToItemID` options, or `caseID|sessionID:caseID`.
- `caseID|sessionID:caseID|sessionID:caseID:stepID...`
- specifies the test case IDs to view test step results for. The test cases must all belong to the session specified in the `--sessionID` option. If the session is not specified in the `--sessionID` option, you must specify the session ID and case ID for each test case you want to view test step results for. You can also specify individual test steps in a test case to view results for. If you do not specify this option, you must specify the `--sessionID` option, and results are displayed for all steps in all test cases for that session.

  Note

  The test session item type, test case item type, and test step item type must be visible to you in order for you to view the test step results.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm results, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm testcases

displays test cases for a test session

Synopsis

`tm testcases [--fields=field1[:width1],field2[:width2]...][--fieldsDelim=value] [--[no]substituteParams][--queryDefinition=query] [--height=value] [--width=value] [--x=value][--y=value] [--hostname=value] [--port=value] [--password=value][--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)][--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value][--forceConfirm=[yes|no]] caseID|sessionID:caseID...`

Description

This command allows you to to view the test cases for one or more test sessions. Test cases are displayed in the order they appear in the test document(s) referenced by the test session. Attachments and related items display as a comma separated list of attachment names and related item IDs.

Note

The test cases that display for test documents are based on the date in the test session's Tests As Of Date field. Test cases added to test suites after this date are not displayed.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--fields=field1[:width1],field2[:width2]...`

  the test case fields, and their respective widths, to be displayed. Use commas to specify more than one field.
- `--fieldsDelim=value`

  the string to be used as a delimiter between the fields in the display.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `--queryDefinition=query`

  defines the query that determines what test cases display. For details of the query format, see the im createquery reference page.

  Note

  Do not use complex query definitions. For example, any form of query that uses a short or long text field.

- *sessionID...*

  specifies one or more test sessions that you want to view test cases for.

See Also

- Commands: tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm results, tm setresults, tm viewuntested
- Miscellaneous: options

tm verdicts

displays the list of test verdicts

Synopsis

```
tm verdicts [--fields=field1[:width1],field2[:width2]...] [--fieldsDelim=value] [--height=value] [--width=value] [-x value] [-y value] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [(-F file|--selectionFile=file)] [-g|--gui] [--settingsUI=[gui|default]] [--quiet] [--status=[none|gui|default]] verdict...
```

Description

`tm verdicts` displays a list of test verdicts. By default, all verdicts are displayed.

Options

This command takes the universal options available to all CLI commands, as well as some general options. See the options reference page for descriptions.

- `--fields=`*field1[:width1],field2[:width2]...*

  where field*n* can be any of the following:

  - `name`

    displays the name of the verdict. By default, only the name is shown.
  - `id`

    displays the database ID of the verdict. This is for PTC - Integrity Support only.
  - `displayName`

    displays the name assigned as the display name of the verdict.
  - `description`

    displays a description of the verdict.
  - `image`

    displays whether or not there is an image for the verdict.
  - `isActive`

    displays whether or not the verdict is active.
  - `verdictType`

    displays the verdict type.
  - `position`

    displays the sequential position of the type in the list of types.

  Note

  Field names are case sensitive.

- `--fieldsDelim=`*value*

  specifies the string to be used as a delimiter between fields displayed in the CLI.
- *verdict...*

  specifies names of the verdicts to display.

See Also

- Commands: tm createverdict, tm deleteverdict, tm viewverdict, tm editverdict
- Miscellaneous: options

tm viewprefs

displays preferences for test management commands

Synopsis

`tm viewprefs [--[no]global] [--command=value] [--[no]showValidValues] [--[no]ask] [--ui=[unspecified|gui|cli|api]] [(-?|--usage)] [(-N|--no)] [(-Y|--yes)] [(-F value|--selectionFile=value)] [--[no]batch] [--cwd=value] [--forceConfirm=[yes|no]] [(-g|--gui)] [--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]]`

Description

`tm viewprefs` displays preferences and configuration options for Integrity test management commands. These settings are used to determine default behaviors for other commands. You can only view one set of preferences at a time.

Options

This command takes the universal options available to all Integrity commands, as well as some general options. See the `options` reference page for descriptions. For an easy way to see a list of commands and values that may be set, type the `tm viewprefs` command, either piped through `|more` or redirected to a file, for example:

```
tm viewprefs --global --showValidValues >prefs.txt
```

Alternatively, the `--gui` option presents a dialog box that lets you view and configure the preferences.

- `--[no]global`

  specifies whether to show all preferences.
- `--command=value`

  identifies the command that preferences are to be viewed for.
- `--[no]showValidValues`

  specifies whether to display a list of valid values for the preferences.
- `--[no]ask`

  specifies whether to show the ask preference. Each preference option may be set to either `--ask` or `--noask`. When the command itself is run, any option set to `--ask` and that is not explicitly set with command line options will be queried. If this `--ask` option is set, then you do not specify a value for the preference at the same time, but instead the `pref=value` must supply one of the following four valid `ask` values:
  - `once`

    asks the user the first time only, and then uses the provided value every time after.
  - `never`

    never asks the user for a response, but uses the current setting (which may be specified by a preference).
  - `element-last`

    asks the user for each element of the selection, providing the most recently used value as the default.
  - `element-pref`

    asks the user for each element of the selection, resetting the default to the value specified by the preference.
- `--ui=[unspecified|gui|cli|api`

  controls whether to view the preference for the graphical user interface, the command line interface, or an unspecified interface. By default, `--ui=cli` is implied when using `tm viewprefs`. To view preferences for GUI behavior, however, you should specify `--ui=gui`. For example, to view the preference for the tm resulteditor command, type:

  ```
  tm viewprefs --command=resulteditor --ui=gui
  ```

  These correlate to settings in the`IntegrityClient.rc` file that have the `gui.tm.` or `cli.tm.` prefix, or the `tm.` prefix when it is unspecified.

See Also

- Commands: tm setprefs. tm createresult, tm editresult, tm viewresult, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm results, tm viewuntested
- Miscellaneous: ACL, options

tm viewresult

displays test result details for the specified test cases

Synopsis

```
tm viewresult [--sessionID=value] [--[no]showSteps[--[no]substituteParams] [--height=value] [--width=value] [--x=value][--y=value] [--hostname=value] [--port=value] [--password=value][--user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)][--quiet] [--settingsUI=[gui|default]] [--status=[none|gui|default]][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value][--forceConfirm=[yes|no]] caseID|sessionID:caseID...
```

Description

This command allows you to to view the test result details for one or more test cases. Attachments and related items display as a comma separated list of attachment names and related item IDs.

For example,

```
tm viewresult --showSteps --sessionID=8071 9023
```

displays the test result details, including test steps, for test case 9023 in test session 8071.

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=`*value*

  the ID of the test session item that you want to view result details for.
- `--[no]showSteps`

  specifies whether to show test steps for test case results. The default is to show the test steps.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- *caseID*|*sessionID:caseID...*

  specifies the test case IDs to view test result details for. The test cases must all belong to the session specified in the `--sessionID` option. If the session is not specified in the `--sessionID` option, you must specify the session ID and case ID for each test case. If you do not specify this option, you must specify the `--sessionID` option, and result detailss are displayed for all test cases in that session.

Note

The test session item type and the test case item type must be visible to you in order for you to view the test result details.

See Also

- Commands: tm createresult, tm editresult, tm results, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewuntested
- Miscellaneous: options

tm viewuntested

displays untested test cases for the specified item and test session

Synopsis

```
tm viewuntested [--sessionID=value] [--[no]substituteParams][--fields=field[:width[:rich|plain]],field[:width[:rich|plain]],...][--
fieldsDelim=value] [--height=value] [--width=value] [--x=value][--y=value] [--hostname=value] [--port=value] [--password=value][--
user=value] [(-?|--usage)] [(-g|--gui)] [(-F value|--selectionFile=value)][--quiet] [--settingsUI=[gui|default]] [--status=
[none|gui|default]][(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=value][--forceConfirm=[yes|no]] itemID...
```

Description

This command allows you to to view the untested test cases for one or more items in one or more test sessions. For example:.

To find all untested test cases in test session 2:

```
tm viewuntested 2
```

or

```
tm viewuntested --sessionID=2 2
```

To find all untested test cases on test objective 3, in test sessions 5, 6, and 7:

```
tm viewuntested --sessionID=5 --sessionID=6 --sessionID=7 3
```

To find all untested test cases on item 10 that has the Tests field visible in test sessions 2 and 3:

```
tm viewuntested --sessionID=2 --sessionID=3 10
```

Options

This command takes the universal options available to Integrity commands, as well as some general options. See the options reference page for descriptions.

- `--sessionID=`*value*

  the test session(s) that you want to view untested test cases for. If a test case does not have a verdict in the test session, it is considered to be untested for that session.

  You must specify at least one test session for this command. If the *itemID* is a test session, you do not need to specify a test session using this option.
- `--[no]substituteParams`

  specifies whether to replace parameter references in text fields with a parameter value. For more information on how parameter values are determined, see the *PTC Integrity User Guide*.
- `--fields=`*field[:width[:rich|plain]],field[:width[:rich|plain]],...*

  specifies the test case fields, and their respective widths, to be displayed. Your administrator defines the fields for the test case. Use commas to specify more than one field. The default fields displayed in the CLI are independent of the default fields (columns) specified in the GUI and Web interface.

  For rich content fields, you can specify display patterns to display field values as rich content (`:rich`) or plain text (`:plain`). The rich content display pattern displays the underlying HTML elements and attributes in the rich content field. For example, `--fields=Description::rich` displays the Description field value with HTML elements and attributes. By default, rich content fields display plain text.
- `--fieldsDelim=`*value*

  specifies the string to be used as a delimiter between the fields in the display.
- *itemID...*

  specifies the item IDs to view untested test cases for. Any test cases related to the items through the Tests field that do not have a test verdict for the test sessions specified in the `--sessionID` option are considered to be untested.

Note

An item ID is required and the item type must contain a visible Tests field. If the item ID you specify is an item type that performs the test session role, the `--sessionID` option is not required. If the item ID you specify is an item type that is not a test session, you must specify at least one test session using the `--sessionID` option.

See Also

- Commands: tm createresult, tm editresult, tm results, tm extractattachments, tm stepresults, tm resulteditor, tm deleteresult, tm editresult, tm testcases, tm setresults, tm viewresult
- Miscellaneous: options

tm viewverdict

displays the attributes of an test verdict

Synopsis

`tm viewverdict [--height=value] [--width=value] [-x value] [-y value] [--[no]showHistory] [--quiet] [--user=name] [--hostname=server] [--password=password] [--port=number] [(-?|--usage)] [(-F file|--selectionFile=file)] [(-N|--no)] [(-Y|--yes)] [--[no]batch] [--cwd=directory] [--forceConfirm=[yes|no]] [-g|--gui] [--settingsUI=[gui|default]] [--status=[none|gui|default]] test verdict...`

Description

`tm viewverdict` lists detailed information about specified test verdicts.

Options

This command takes the universal options available to all CLI commands, as well as some general options. See the options reference page for descriptions.

- `--[no]showHistory`

  specifies whether to display a read-only log of all changes to the test verdict. The default is to not show the history. If you display the history, the information displays in chronological order (the most recent changes appear at the bottom).
- *`test verdict`*

  specifies the name of the verdict you want to view. Verdict names are case sensitive.

See Also

- Commands: tm createverdict, tm deleteverdict, tm editverdict, tm verdicts
- Miscellaneous: options