PTC®

PTC Integrity™ 10.6

CLI Reference for Server Administration

PTC Integrity 10.6
CLI Reference for Server Administration

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Integrity CLI Reference for Server Administration

PTC provides a command line interface (CLI) to manage Integrity Server administration tasks.

Note: For more information on using Authorization Administration and ACLs, refer to the *PTC Integrity Server Installation and Configuration Guide*.

IMPORTANT: The terms item and issue refer to the same Integrity 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. Use the term that is documented for each command.

Each command allows a limited set of options. Single letter options must always be preceded by a single dash (`-`), while longer option strings must be preceded by a double dash (`--`). The long strings are not case sensitive, but are shown in mixed case to facilitate readability.

To view a list of options available to a particular command, simply append `-?` or `--usage` to the command.

In options, square brackets indicate optional strings, for example, the `no` is an optional prefix in `--[no]batch`. The two ways to use this option would be `--nobatch` or `--batch`.

Introduction

Authorization Administration Commands

Workflow and Document Management Commands

- im createproject
- im createstate
- im createtrigger
- im createtype
- im createuser
- im deletedynamicgroup
- im deletefield
- im deletegroup
- im deleteissue
- im deleteproject
- im deletestate
- im deletetrigger
- im deletetype
- im deleteuser
- im dynamicgroups
- im echo
- im editdynamicgroup
- im editfield
- im editgroup
- im editproject
- im editstate
- im edittrigger
- im edittype
- im edituser
- im extractwordtemplates
- im fields
- im getdbfile
- im groups
- im importgroup
- im importissue
- im importuser
- im installsolution
- im logging
- im obtainadminlock
- im projects
- im purgeauditlog
- im putdbfile
- im releaseadminlock
- im runtrigger
- im setnotification
- im setproperty
- im states
- im triggers
- im types
- im users
- im viewadminlock
- im viewauditlog
- im viewdynamicgroup
- im viewfield
- im viewgroup
- im viewproject
- im viewstate
- im viewtrigger
- im viewtype
- im viewuser

Test Management Commands

- tm createverdict
- tm deleteverdict
- tm editverdict

- tm purgeresults
- tm resultfields
- tm verdicts
- tm viewverdict

Integrity Server Commands

- integrity about
- integrity admin
- integrity admingui
- integrity changemksdomainuserpassword
- 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 loadrc
- integrity mksdomaingroups
- integrity mksdomainusers
- 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

Configuration Management Commands

- si admingui
- si copypolicysection
- si deletearchive
- si deletepolicysection
- si deleteproject
- si echo
- si getdbfile
- si logging
- si primeproject
- si purgeauditlog
- si putdbfile
- si setpolicysection
- si setproperty
- si viewauditlog
- si viewpolicysection
- si viewpolicysections
- si migrate

Miscellaneous Information

- acl
- diagnostics
- options
- preferences

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 command through the CLI **man** command.

The possible prefixes for CLI commands are `integrity`, `aa`, `si`, and `im`. Using workflow and document functionality to illustrate, first, you may simply type the `im` prefix and the command together as one word. Second, you may 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.

This reference page describes the parts of a reference page with examples taken from real reference pages for workflow and document management.

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` [**`--addAttachment`**=*value*] [**`--addRelationships`**=*value*] [**`--type`**=*value*] [**`--field`**=*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]*]

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 management 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 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 issues; here is the command's synopsis:

`im editissue` [**`--addAttachment`**=*value*] [**`--addRelationships`**=*value*] [**`--[no]batch`**] [**`--query`**=*query*] [**`--removeAttachment`**=*value*] [**`--removeRelationships`**=*value*] [**`--field`**=*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]*] *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 CLI 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 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 Arial*** 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.

man

display online reference pages

SYNOPSIS

man [**-wx**] [**-M** *path*] [*type*] *entry* ...

man [**-wx**] [**-T** *txt_indexes*] [*type*] *entry* ...

man -h [**-wx**] [**-C** *chm_indexes*] [*type*] *entry* ...

man -H [**-wx**] [**-P** *html_paths*] [*type*] *entry* ...

man -k [**-M** *path*] *keyword* ...

DESCRIPTION

The **man** command either displays online reference pages or searches for reference pages that have specified keywords associated with them.

Normally, **man** displays the reference page for each specified *entry*. To display only a reference page of a given type, specify *type* on the command line. *type* is a number representing which type of reference pages to search. Reference pages come in the following types:

1	Commands and Utilities
3	Functions
4	File Formats
5	Miscellaneous

To indicate an operating system specific version of the entry (if one exists) or to indicate an command specific to a given set of commands and/or functions, append one of the following letters to the specified *type*:

n	for Windows NT/2000/XP/2003/Vista/7/2008
t	for Tcl

When output is sent to the terminal, **man** invokes a pager command to filter and display the reference pages. If ***MANPAGER*** is defined, it is used. If not, and if ***PAGER*** is defined, it is used. If neither is defined, **man** defaults to using the command **more -A -s**.

Options

-C *filelist*
: specifies a list of `.idx` files (corresponding to `.chm` files) to search instead of searching the files listed in ***MAN_CHM_INDEX***.

-H
: launches the default web browser and displays the HTML version of the reference page. The reference page is found by searching each path listed in the ***MAN_HTM_PATHS*** environment variable (or indicated by the **-P** option) for an entry matching *entry* and *type* that indicates which page in the corresponding `.chm` file to display.

-h
: launches the HTML Help viewer and displays the HTML Help version of the reference page. The reference page is found by searching each `.idx` listed in the ***MAN_CHM_INDEX*** environment variable (or indicated by the **-C** option) for an entry matching *entry* and *type* that indicates which page in the corresponding `.chm` file to display.

-k
: searches a precomputed database of synopsis lines for information on *keyword*s.

-M *path*
: searches the directories indicated by *path* for reference pages. If **-M** is not specified, **man** uses the path specified in the ***MANPATH*** environment variable if it is set; otherwise **man** searches `$ROOTDIR/etc`. All reference pages are found by searching similarly structured file trees rooted at one or more places. See the *FILES* section for a description of the files and directories **man** should find in each directory that it searches.

-P *html_paths*
: specifies a list of HTML paths to search instead of searching the paths listed in ***MAN_HTM_PATHS*** when looking for a HTML version of a reference page.

  If this option is not specified and ***MAN_HTM_PATHS*** is not set, **man** searches `$ROOTDIR/etc/htm` by default. All reference pages are found by searching similarly structured file trees rooted at one or more places. See the *FILES* section for a description of the files and directories **man** should find in each directory that it searches.

-T *filelist*

specifies a list of `.idx` files to search instead of searching the files listed in ***MAN_TXT_INDEX*** when looking for a text version of a reference page.

-w

displays only the file name of the file containing the specified entry.

-x

displays the files that **man** is searching as it tries to find the entry.

Search Rules

To find a given entry, **man** follows a set of search rules. When you specify a *type*, **man** searches for the appropriate page amongst pages of that type; otherwise, **man** looks for the first page named *entry* regardless of the type.

When the **-h** option is specified, **man** searches the `.idx` files listed in the ***MAN_CHM_INDEX*** environment variable for an entry matching the specified *entry* which indicates the HTML Help page in corresponding `.chm` to display. The HTML Help viewer is launched, displaying the page. Once you exit, the view, the **man** command exits.

When the **-H** is specified, **man** takes the following steps to find the entry. Once found, **man** launches the default web browser to display the reference page.

- For each possible type (that is, *type* if you specified it, or all types in order from 1 through 9, then 0 if you did not),

  man checks each directory in ***MAN_HTM_PATHS*** (or specified by **-P**) for a file named `htm`*n*`/`*entry*`.`*n*`[`*l*`].html` (or if that file is not found, `htm`*n*`/`*entry*`.`*n*`[`*l*`].htm`) where *n* is the type number, and *l* is the optional letter code.

- If still not found, **man** checks each directory in ***MAN_HTM_PATHS*** (or specified by **-P**) for a file named *entry*`.html` (or if that file is not found, *entry*`.htm`).

When neither **-H** nor **-h** is specified, **man** takes the following steps to find the entry. Once a step results in finding the entry, **man** displays the reference page and exits.

- For each possible type (that is, *type* if you specified it, or all types in order from 1 through 9, then 0 if you did not):

  - **man** checks each directory in ***MANPATH*** for a file named `cat`*n*`/`*entry*`.`*n*`[`*l*`]` where *n* is the type number, and *l* is the optional letter code. If it exists, **man** checks to see if it was compressed with **pack**, **compress** or **mkszip**, and uncompresses it (calling **pcat** if the file was **pack**ed).

  - **man** checks each directory in ***MANPATH*** for a file named `man`*n*`/`*entry*`.`*n*`[`*l*`]`.

- **man** checks each directory in ***MANPATH*** for a file named `man.dbz`. If it exists, **man** looks for the requested *entry* in its index (see *man.dbz File Format*).

- **man** searches the `.idx` files listed in the ***MAN_TXT_INDEX*** environment variable for an entry matching the request *entry* which indicates the text (`.txt`) reference page to display.

man.dbz File Format

Sometimes, the reference pages are kept in a single large file, called `man.dbz`. The file starts with a magic text string:

```
!<man database compressed>\n
```

and continues with the index:

```
14 bytes formatted reference page name
9 bytes seek pointer
9 bytes length
```

The name is simply the page name, followed by a dot and the type number. For example, this reference page would be named `man.1`. When **man** finds a matching entry, it seeks to the point in the file specified by the given seek pointer, and uncompresses for length bytes. Each reference page is compressed separately.

EXAMPLES

To find the utilities that do comparisons, type:

```
man -k compar
```

ENVIRONMENT VARIABLES

MAN_CHM_INDEX
contains a semicolon separated list of `.idx` files to search for *entry* when the **-h** is specified.

MAN_TXT_INDEX
contains a semicolon separated list of `.idx` files to search for *entry* when the **-h** is not specified.

MAN_HTM_PATHS
contains a semicolon separated list of `.idx` paths to search for *entry* when the **-H** is specified.

MANPATH
contains a semicolon separated list of paths to search for reference pages.

MANPAGER, ***PAGER***
contains an output filtering command for use when displaying reference pages on a terminal.

TMPDIR
identifies the directory where temporary files reside.

FILES

`$ROOTDIR/etc`
is the default directory for the online reference pages. The rest of the files listed here reside in this directory.

`cat[0-9]/*.[0-9]`
pre-formatted reference pages in normal, compressed, or packed form.

`htm/htm[0-9]/*.[0-9]`
HTML reference pages.

`man[0-9]/*.[0-9]`
unformatted reference pages.

Note:
Unformated reference pages are not currently supported. Reference pages stored in these directories are treated as pre-formatted pages.

`whatis`
is a database used by **-k** option.

`*.chm`
`chm/*.chm`
HTML Help files containing collections of reference pages complete with index, table of contents, and full text search.

`*.idx`
`chm/*.idx`
index files that **man** how to find HTML Help and text versions of individual reference files. The `.idx` files to search are indicated by the ***MAN_CHM_INDEX*** and ***MAN_TXT_INDEX*** environment variables.

`man.dbz`
is a master file containing all reference pages.

The `etc` directory is found using the ***ROOTDIR*** environment variable.

DIAGNOSTICS

Possible exit status values are:

`0`
Successful completion.

`1`
Failure due to any of the following:

— unknown command line option
— missing *path* after an **-M** option
— no information available on the desired subject

— unable to create a child process to format reference page
— child process returned with non-zero exit status

PORTABILITY

POSIX.2. *x*/OPEN Portability Guide 4.0. All UNIX systems. Windows 2000. Windows XP. Windows Server 2003. Windows Vista. Windows 7. Windows Server 2008.

The **-C**, **-H**, **-h**, **-M**, **-P**, **-T**, **-w**, and **-x** options, the ***MANPAGER***, ***MAN_HTM_PATHS***, ***MAN_CHM_INDEX***, and ***MAN_TXT_INDEX*** environment variables, the default pager, the ability to specify *type* on the command line, and the ability to display reference pages in HTML Help format are all extensions to the POSIX and XPG standards.

AVAILABILITY

MKS Toolkit for Power Users
MKS Toolkit for System Administrators
MKS Toolkit for Developers
MKS Toolkit for Interoperability
MKS Toolkit for Professional Developers
MKS Toolkit for Enterprise Developers
MKS Toolkit for Enterprise Developers 64-Bit Edition
Integrity

SEE ALSO

Commands:

help, **manstrip**, **more**

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 addaclentry

creates ACL entries on the Integrity Server

SYNOPSIS

aa addaclentry [**--acl**=*name*] [**--[no|confirm]allowNonexistentPrincipal**] [**--[no|confirm]createAcl**] [**--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.

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; set up Deploy; 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; and Deploy). 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.

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 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**] [(**-F***file*|**--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 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 ***aa*** commands, as well as some general options. See the **options** reference page for descriptions.

SEE ALSO

Commands:

aa viewserveralerts, **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**] [**-F***file*|**--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, configuration management, and Deploy. 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 Installation and Configuration Guide*.

Note the following:

- 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, an administrator with the Deploy `AdminServer` permission can edit Deploy 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 admin**, **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*] [**-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]*] [*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]*]

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.

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**] [(**-F***file*|**--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**

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**)] [**-x***value*] [**-y***value*] [**--[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, set up Deploy, 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; and Deploy). 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(“Financial Toolkit Defects: In Dev”,
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=‘”Defect Count”’ –-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 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 **--im createquery** in the *Integrity CLI Reference for Workflows and Documents*.

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 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]*] [**--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.

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.

`Important`: Duplicate detection is not supported for IBM DB2 or DB2 for i databases. If you attempt to enable duplicate detection when using DB2 or DB2 for i, an error message is displayed.

--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.

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 for items of this type. This setting overrides `--copyCommonFields` if set.

--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 the following:

- 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*.

> 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 the following:

- 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 the following:

- 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.

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 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**)] [(**-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 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 the following:

- 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 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*] [(**-F***file*|**--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.

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 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]*] [**--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,...*] [**--displayLocation**=*value*] [**--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**] [**--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*]

DESCRIPTION

im createfield creates a field for workflows and documents. For example:

```
im createfield --hostname=abcFinancial --user=jriley --type=integer  --
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.

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*
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 the following:

- 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.
- 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 the following:

- 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 Installation and Configuration 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 following:

- 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.

--default=*value*
specifies the default value for the field.

Note the following:

- 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, Query Backed Relationship. This option sets default columns 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.

`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 following:

- 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:

- 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** in the *Integrity CLI Reference for Workflows and Documents*.

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 contains” 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 *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.

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.

`--displayLocation`=*value*
specify whether the query backed relationship field is displayed under `Fields` or `Relationships` when you view the issue details. Acceptable values are *relationship* or *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.

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 the following:

- 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 the following:

- 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 the following:

- 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.

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.

Important: Once you include the time and save the date field, you cannot change the date field to display the date only.

--[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.

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**] [**-F***file*|**--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 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 the following:

- 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 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**)] [(**-F***file*|**--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 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 the following:

- 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*] [**--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]*] [**--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.

`Important`: Duplicate detection is not supported for IBM DB2 or DB2 for i databases. If you attempt to enable duplicate detection when using DB2 or DB2 for i, an error message is displayed.

--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 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.

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 for items of this type. This setting overrides `--copyCommonFields` if set.

--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 the following:

- 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*.

> `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 the following:

- 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 the following:

- 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.

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**] [**-F***file*|**--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 the following:

- 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 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**)] [(**-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]*] *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.

IMPORTANT: 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 the following:

- 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 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 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 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`.

`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 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 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 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 *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*]
[**--[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,...*] [**--displayLocation**=*value*] [**--displayRows**=*value*]
[**--displayStyle**=*value*] [**--enableLinkFlags**=*value*] [**--disableLinkFlags**=*value*] [**--displayName**=*value*] [**--[no]trace**]
[**--displayPattern**=*value*] [**--showDateTime**] [**--[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]*] *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  --storeToHistoryFrequency=weekly
"Days_In_Phase"
```

changes the *Days_In_Phases* field to calculate on a weekly basis.

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 specifed 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.

--removeOverride=*description,relevanceRule,editabilityRule,ranges,displayPattern,phases,defaultColumns,default,substituteParams,displayAs*
: 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.

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 the following:

- 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 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.
- 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, 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.

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 following:

- 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:

- 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** in the *PTC Integrity CLI Reference for Workflows and Documents*.

`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" comparision.

--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.

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.

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.
- 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. See **--addEntry**=*value* for the format of *value*.

--deleteEntry=*value*
deletes an existing phase or range field value. See **--addEntry**=*value* for the format of *value*.

--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 the following:

- 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 the following:

- 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 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 the following:

- 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.

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.

--displayLocation=*value*
specify whether the query backed relationship field is displayed under Fields or Relationships when you view the issue details. Acceptable values are *relationship* or *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.

--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.

Important: Once you include the time and save the date field, you cannot change the date field to display the date only.

--[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.

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*.

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 the following

- 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 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**)] [(**-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

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 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*] [**--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=capabilities Approved
```

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*.

Important: 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 specifed 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 the following:

- 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*] [**--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]*]
[**--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.

`Important`: Duplicate detection is not supported for IBM DB2 or DB2 for i databases. If you attempt to enable duplicate detection when using DB2 or DB2 for i, an error message is displayed.

--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 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.

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 for items of this type. This setting overrides `--copyCommonFields` if set.

--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 the following:

- 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*.

`Important`: 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 the following:

- 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 the following:

- 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.

type
specifies the name of the type you want to edit.

SEE ALSO

Commands:

im createtype, **im viewtype**, **im deletetype**, **im types**, **im branch** **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 the following:

- 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 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 occuring 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.

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.

--[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 your *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 delimeter 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 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*] [**--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 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, 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.

value

--addRelationships=
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 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 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=every
--
userMap=administrator=guest,analyst=guest,developer=guest,integration=guest,manager=guest,tester=guest
--siProjectDir=C:/RM_repo7
```

Important: 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 logging

modifies logging levels on the Integrity Server

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 logging levels for the Integrity Server. For example:

```
im logging --target=client --category=DEBUG --level=5
```

changes *debug* logging on the client to level *5*.

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 in the table next). You can run this command from a client or server machine. Any category included in the `logger.properties` file can be used.

Note the following:

- 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. 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, 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:si. Resolved ACL: mks:si
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 obtainadminlock

set an administrative lock on the Integrity Server

SYNOPSIS

im obtainadminlock [**--[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 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 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 delimeter 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 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 Installation and Configuration Guide*.

For example, the following command purges all audit log records prior to 10:00 AM 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 10:00 AM 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 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 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 setnotification

sets e-mail notification for a user or group

SYNOPSIS

im setnotification [**--email**=*value*] [**--notificationRule**=*value*] [**--notificationRuleFile**=*value*] [(**-F***file*|**--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 the following:

- 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 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**] [**-F***file*|**--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, configuration management, and Deploy. 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 Installation and Configuration Guide*.

Note the following:

- 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, an administrator with the Deploy `AdminServer` permission can edit Deploy 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 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 delimeter 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 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 delimeter 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 for items of this type.

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.

allowAdditonalLockFields
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 delimeter 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 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 delimeter 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 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 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 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*] [**-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]*] *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**

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 --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 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 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 edit. Verdict names are case sensitive.

SEE ALSO

Commands:
tm createverdict, **tm deleteverdict**, **tm viewverdict**, **tm verdicts**

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 Jnauary 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 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 occuring 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 your *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 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 delimeter 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 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**

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 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; and Deploy). 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, set up Deploy, 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; and Deploy). 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**

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]*] [(**-F***file*|**--selectionFile**=*file*)] [(**-g**|**--gui**)] [**--quiet**] [**--settingsUI**=*[gui|default]*] [**--status**=*[none|gui|default]*]

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 the following:

- 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.

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.

  IMPORTANT: 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.

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

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 Getting Started 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 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 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.

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**

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**] [(**-F***file*|**--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 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 ***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*] [**-x***value*] [**-y***value*] [**-?**|**--usage**] [**-N**|**--no**] [**-Y**|**--yes**] [**--[no]batch**] [**--cwd**=*directory*] [**--forceConfirm**=*[yes|no]*] [**-g**|**--gui**] [**-F***file*|**--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 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 when using **integrity connect**, 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, configuration management, and Deploy. 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 Installation and Configuration Guide*.

Note the following:

- 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, an administrator with the Deploy `AdminServer` permission can edit Deploy 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**] [(**-F***file*|**--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 the following:

- 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]*] [(**-F***file*|**--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 the following:

- 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 the Integrity 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 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

displays Integrity 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**] [(**-F***file*|**--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]*] [(**-F***file*|**--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**

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, set up Deploy, 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; and Deploy). 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 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 the following:

- 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 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.

   IMPORTANT: 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.

IMPORTANT: 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` 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.

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 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 following:

- 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.

   IMPORTANT: 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 (“dumping” 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.

IMPORTANT: 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 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 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 logging

modifies logging levels on the Integrity Server

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 logging levels for the Integrity Server.

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:

```
im logging --target=client --category=CACHE --level=5
```

changes *cache* logging on the client to level *5*.

Note the following:

- 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 (0 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`**

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 Installation and Configuration 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 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]*] [(**-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

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 10:00 AM 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 10:00 AM 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 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 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 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 following:

- 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 file system rather than its relative name in the project hierarchy. This enhances the portability of change packages across different projects. Therefore, the actual file system name of the subproject is used to set a subproject policy.

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.

`Important`: 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 Installation and Configuration 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`
- `TextWorkingFileSizeGovernor=0`

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*

Lock Policies:

- `TextArchiveLockingPolicy=nonexclusive|exclusive|none`
- `BinaryArchiveLockingPolicy=nonexclusive|exclusive|none`

States Policy

`States=`*SpaceDelimitedList*

Keywords Policy

`Keywords=`*keyword*

Line Terminator Policies:

- `PreserveLineTerminatorsInTextArchives=true|false`
- `si.lineterminators=crlf|lf|native`

Integrity Client Default Policies:

- `commands=`*command*
- `views=`view

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 the following:

- 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 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**] [**-F***file*|**--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, configuration management, and Deploy. 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 Installation and Configuration Guide*.

Note the following:

- 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, an administrator with the Deploy `AdminServer` permission can edit Deploy 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 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*] [(**-F***file*|**--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 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`.
- `;` *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**

Miscellaneous:

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.

IMPORTANT:

- 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**

ACL (Access Control List)

permissions for Integrity Server ACL interaction

DESCRIPTION

The ACL (Access Control List) permissions control user access to Integrity Server functions (configuration management; workflows and documents; and Deploy) by associating development objects and operations with specific permissions. For example, whenever a user initiates an operation such as checking a file in or out, Integrity queries the ACL database to determine whether the user has permission to perform the operation. This reference page is provided as a guide to the ACL permissions.

By default, the following server-level ACLs are included:

- *mks*--controls root level access to Integrity Server operations. This allows you to set administrative permissions for the Integrity Server in one place; however, you can override the root level control by setting administrative permissions for the workflow and document, and configuration management components. For example, if the `AdminServer` permission is denied for the `mks` ACL, you can enable configuration management administration of the Integrity Server by allowing the `AdminServer` permission for the `mks:si` ACL.
- *mks:aa*--controls the Login access to the AA application for managing the ACLs.
- *mks:aa:mks*--controls Read and Update access to the ACLs.
- *mks:im* controls access to workflow and document operations.
- *mks:si*--controls access to configuration management operations.
- *mks:sd*--controls access to Deploy operations.
- *mks:patch* controls the Download permission required for service pack management.
- *mks:system:viewsets*--controls access to publishing ViewSets.
- *mks:system:mksdomain*--controls access to administering the MKS domain.

For the most part, you will 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 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>*.
- Archive ACLs simply specify the archive name in the ACL name, such as *mks:si:archive:<archive path>*.

ACL PERMISSIONS

You must have the appropriate ACL permissions before you can perform configuration management, and workflow and document operations. For details on configuring ACLs, see the *PTC Integrity Server Installation and Configuration Guide*.

Integrity Server Permissions

The following summarizes the Integrity Server permissions available under `mks`:

AdminProxy

For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

DebugProxy

For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the proxy.

Prerequisites: none.

DebugServer

For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the server.

Prerequisites: none.

Login

Allows a user to log in to Integrity.

Prerequisites: none.

Configuration Management Server Permissions

The following summarizes the configuration management server-related permissions available under `mks:si`:

AdminProxy

For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

DebugProxy

For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the proxy.

Prerequisites: none.

DebugServer

For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the server.

Prerequisites: none.

EditPolicy

Allows a user to modify and create configuration management policies on the Integrity Server. The Edit Policy permission should be restricted to administrators and configuration management project managers

Prerequisites: Login.

Login

Allows a user to log in to Integrity.

Prerequisites: none.

StagingSystemAdmin

Required to administer a staging system. Allows a user to perform any Deploy operation in the staging system. Users are automatically granted this permission at the project level when they create a staging system. This permission overrides any global permissions. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: none.

ViewPolicy

Allows a user to view configuration management policies on the Integrity Server. The View Policy permission should be restricted to administrators and configuration management project managers.

Prerequisites: Login.

Configuration Management Member Permissions

The following summarizes the configuration management member-related permissions available under `mks:si`:

ApplyLabel
Allows a user to add labels to revisions or move labels between revisions.

Prerequisites: ***Login***, ***OpenProject***.

CheckIn
Allows a user to check in working files as new revisions of members.

Prerequisites: ***Login***, ***OpenProject***, ***ApplyLabel***, ***Lock***, ***ModifyAuthor***, ***ModifyMemberRev***, ***ModifyMemberAttribute***.

DeleteLabel
Allows users to delete a label from a revision.

Prerequisites: ***Login***, ***OpenProject***.

DeleteRevision
Allows a user to delete revisions from the member history.

Note:
This permission allows users to irrevocably delete revisions from the member history. Administrators should assign this permission carefully.

Prerequisites: ***Login***, ***OpenProject***.

Demote
This permission allows a user to change the promotion state of revisions from a higher setting to a lower one, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

DowngradeOtherUserLock
Allows a user to downgrade exclusive locks held by other users to non-exclusive locks.

Prerequisites: ***Login***, ***OpenProject***.

FetchRevision
Allows a user check out member revisions.

Prerequisites: ***Login***, ***OpenProject***, ***Lock***, ***ModifyMemberRev***.

Freeze
Allows a user to freeze members. When a member is frozen, all configuration management operations are run on the frozen member revision.

Prerequisites: ***Login***, ***OpenProject***.

Lock
Allows a user to lock revisions.

Prerequisites: ***Login***, ***OpenProject***.

ModifyAuthor
Allows a user to change the author name associated with a revision.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberAttribute
Allows a user to set an attribute for a member that can be used later in a search.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberRule
Allows a user to configure a member revision rule that can be applied to one or more members.

Prerequisites: ***Login***, ***OpenProject***.

MoveLabel

Allows a user to move a member label to another revision within the member history.

Prerequisites: ***Login***, ***OpenProject, ApplyLabel***.

Promote

This permission specifies that a user may promote revisions from the current promotion state to a higher state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

ShareArchive

Allows sharing of member archives between two or more members.

Note:

Archive sharing is not recommended. Instead, creating variant Sandboxes is considered a better practice.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint***, ***CheckIn***, ***Lock***.

Thaw

Allows a user to thaw frozen members.

Prerequisites: ***Login***, ***OpenProject***.

Configuration Management Change Package Permissions

The following summarizes the configuration management change package-related permissions available under `mks:si`:

BypassChangePackageMandatory

Allows the user to bypass the Change Packages Mandatory policy, permitting the user to perform configuration management operations without change packages.

Prerequisites: none.

ChangePackageAdmin

Allows a user to edit, discard, close, and submit change packages; as well as move or discard change package unties, regardless of any documented user restrictions.

Prerequisites: none.

CreateChangePackage

If you are using configuration management functionality only, this permission allows a user to create change packages.

If you are using configuration management, and workflow and document functionality, this permission allows a user to create change packages based on the Change Package Creation Policy for the type that they want to create change packages for.

Prerequisites: ***Login***, ***OpenProject***.

PromoteCP

Required to promote a change package in the staging system corresponding to the ACL. Used to authorize use of the **`sd promotecp`** command. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: none.

SelfReview

Allows user to accept change packages under review that were created by that user.

SuperReview

Allows a user to accept or reject a change package under review regardless of the reviewer rules. Note: This permission supersedes the SelfReview permission.

Configuration Management Project Permissions

The following summarizes the configuration management project-related permissions available under `mks:si`:

AddMember

Allows a user to add new members to projects through a Sandbox.

Prerequisites: ***Login***, ***OpenProject***, ***Lock***, ***ShareArchive***, ***ModifyAuthor***.

AddProject

Allows a user to re-add a dropped project.

Prerequisites: ***Login***, ***OpenProject***.

AddSubproject

Allows a user to re-add dropped subprojects to a project.

Prerequisites: ***Login***, ***OpenProject***.

ApplyProjectLabel

Allows a user to add labels to projects or move labels between revisions of the project.

Prerequisites: ***Login***, ***OpenProject***.

CheckPoint

Allows a user to check in a new revision of a project (that is, checkpoint the project).

Prerequisites: ***Login***, ***OpenProject***, ***ApplyLabel***, ***Promote***, ***PromoteProject***, ***ApplyProjectLabel***.

ConfigureSubproject

Allows a user to configure a subproject's type. A subproject can be configured as a Normal, Variant, or Build subproject.

Prerequisites: ***Login***, ***OpenProject.***

CreateDevPath

Allows a user to create new development paths for variants of a project.

Prerequisites: ***Login***, ***OpenProject***.

CreateProject

Allows a user to create new projects.

Prerequisites: ***Login***, ***OpenProject***.

CreateSubproject

Allows a user to create new subprojects below existing projects.

Prerequisites: ***Login***, ***OpenProject***.

DeleteProjectLabel

Allows a user to delete a label from a project checkpoint.

Prerequisites: ***Login***, ***OpenProject***.

DemoteProject

This permission specifies that a user may demote projects from a higher promotion state to a lower state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

Deploy

Required to perform deploy request operations and deploy change packages in the staging system corresponding to the ACL. Used to authorize use of `sd startdeployrequest`, `sd stopdeployrequest`, `sd canceldeployrequest`, `sd editdeployrequest`, and **`sd deploycp`** commands. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

DropDevPath

Allows a user to drop a development path, also known as "dropping variants" from a project.

Prerequisites: ***Login***, ***OpenProject***.

DropMember

Allows a user to remove members from projects. The member archive remains, but the member is no longer treated as part of the project.

Prerequisites: ***Login***, ***OpenProject***.

DropProject

Allows a user to drop one or more top-level, registered projects from the server. The projects then become unregistered projects.

Prerequisites: ***Login***, ***OpenProject***.

DropSubProject

Allows a user to drop one or more subprojects from the server. The projects then become unregistered projects.

Prerequisites: ***Login***, ***OpenProject***.

ImportProject

This permission is for legacy functionality that no longer exists in the product.

Metrics

Allows metrics to be tracked for a project. Allows a user to define metrics to be tracked for projects.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberRev

Allows a user to make changes to the member revision of members.

Prerequisites: ***Login***, ***OpenProject***.

ModifyProjectAttribute

Allows a user to set an attribute for a project, which can be used later in a filter or search.

Prerequisites: ***Login***, ***OpenProject***.

MoveProjectLabel

Allows a user to move a project label to another project checkpoint within the project history.

Prerequisites: ***Login***, ***OpenProject***, ***ApplyProjectLabel***.

OpenProject

Allows a user to open existing registered projects. This is required for most actions.

Note:

When OpenProject is granted or denied on a project, clients accessing the project must disconnect and then reconnect in order to get the new permission set. If you do not disconnect and reconnect your client, you may see unexpected behavior due to out-of-date permissions.

Prerequisites: ***Login***.

PromoteProject

This permission specifies that a user may promote projects from the current promotion state to a higher state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

RestoreProject

Allows a user to restore a project to a particular checkpointed version.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint***.

SnapshotSandbox

Snapshot creates and records the state of the user's Sandbox as as a project checkpoint that you can create a build Sandbox or a development path from.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint, AddMember, DropMember***.

StagingSystemAdmin

Required to administer a staging system. Allows a user to perform any Deploy operation in the staging system. Users are automatically granted this permission at the project level when they create a staging system. This permission overrides any global permissions. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

ViewDeployRequest

Required to view a deploy request in the staging system corresponding to the ACL. Used to authorize use of the **`sd viewdeployrequest`** command. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

Workflow and Document Permissions

The following summarizes the workflow and document permissions available under `mks:im`:

Admin

Allows access to administrative functions related to workflows and documents. For super administrator whose tasks include managing users, groups, projects, states, types, and fields. Assign project administrators and type administrators. Customize permissions for change package types. Close change package initiated by another user. Create admin reports, dashboards, charts, and queries. Share reports, dashboards, charts, and queries created by another user, if shared to you. If the object is shared to you, you can also delete it. Create and clear Integrity Server alert messages.

Prerequisites: ***Login***.

AdminProxy

For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

CreateChart

Allows the assigned user or group to create a chart. Denying permission restricts the user or group to using only those charts already on system. For information on charts, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateCPType

Allows the assigned user or group to create a custom change package type. For information on custom change package types, contact PTC - Integrity Support.

Prerequisites: ***Login***.

CreateDashboard

Allows the assigned user or group to create a dashboard. Denying permission restricts the user or group to using only those dashboards already on system. For information on dashboards, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateProject

Allows the assigned user or group to create a new top level project for workflows and documents, and assign another Project Administrator. This permission can be used to extend the capability of the Project Administrator. Denying this permission means the user cannot create a new top level project or assign another Project Administrator.

Prerequisites: ***Login***.

CreateQuery

Allows a user to create a new query. Denying this permission restricts the user to using only those queries that already exist on the system.

Prerequisites: ***Login***.

CreateReport

Allows the assigned user or group to create a report. Denying permission restricts the user or group to using only those reports already on system. For information on reports, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateSharedAdmin

Allows a user to specify if a query, dashboard, report, or chart is a system provided object.

Prerequisites: ***Login***.

CreateType

Allows the assigned user or group to create a new type or assign another Type Administrator. This permission can be used to extend the capability of the Type Administrator. Denying this permission means the user cannot create any new types or assign another Type Administrator.

Prerequisites: ***Login***.

DeleteItem

Delete items of any type.

Prerequisites: Login.

Login

Allows a user to login to Integrity.

Prerequisites: none.

ModifyDeleteItemRule

Allows a user to set a type rule that specifies which users and groups can delete items of that type.

Prerequisites: ***Login***, ***CreateType***.

ModifyMyNotification

Allows a user to modify personal e-mail notification preferences.

Prerequisites: ***Login***, ***ViewMyNotification***.

PurgeTestResult

Allows a user to purge test results for the test cases in a test session.

Prerequisites: ***Login***.

ShareToEveryone

Allows a user to share queries, charts, and reports to the `Everyone` group.

Prerequisites: ***Login***.

TimeTrackingAdmin

Allows a user to create, edit, and delete time entries on behalf of other users. The ability to create, edit, and delete time entries is governed by normal issue permissions.

Prerequisites: ***Login***.

ViewAdmin

Allows a user to view administrative information related to workflows and documents.

Prerequisites: ***Login***.

ViewChangePackage

By controlling the user's ability to view change packages when working in Integrity, the `ViewChangePackage` permission provides an additional level of control for accessing information in projects.

ViewMyNotification

Allows a user to view personal e-mail notification preferences.

Prerequisites: ***Login***.

Available Deploy Permissions

The following summarizes the Deploy permissions available under `mks:sd` to perform specific commands:

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

CreateStagingSystem

A global permission for creating staging systems used to authorize use of the **`sd createstagingsystem`** and **`sd copystagingsystem`** commands. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: `Login`, `OpenProject`.

Available MKS Domain Permissions

The following summarizes the MKS Domain permissions available under `mks:system:mksdomain` to perform specific commands:

AdminServer

Allows a user to administer the MKS Domain.

Prerequisites: none.

Available ViewSets Permissions

The following summarizes the ViewSets permissions available under `mks:system:viewsets` to perform specific commands:

PublishNewViewSet

Allows a user to publish ViewSets to the Integrity Server.

Prerequisites: `Login`.

SEE ALSO

Commands:

`aa acls`, **`aa addaclentry`**, **`aa availablepermissions`**, **`aa deleteacl`**, **`aa deleteaclentry`**, **`aa groups`**, **`aa users`**, **`aa viewacl`**

Miscellaneous:

`diagnostics`

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**).

`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**

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

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).

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`

Summmary: 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 C0-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:

archiveshared
selects 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).

#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*.

#n=
#normal=
specifies that the subproject is a normal subproject. Do not enter a value.

#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.

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.

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 configuration management project to work with all the time.

projectRevision
: specifies a particular 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 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**

ACL (Access Control List)

permissions for Integrity Server ACL interaction

DESCRIPTION

The ACL (Access Control List) permissions control user access to Integrity Server functions (configuration management; workflows and documents; and Deploy) by associating development objects and operations with specific permissions. For example, whenever a user initiates an operation such as checking a file in or out, Integrity queries the ACL database to determine whether the user has permission to perform the operation. This reference page is provided as a guide to the ACL permissions.

By default, the following server-level ACLs are included:

- *mks*--controls root level access to Integrity Server operations. This allows you to set administrative permissions for the Integrity Server in one place; however, you can override the root level control by setting administrative permissions for the workflow and document, and configuration management components. For example, if the `AdminServer` permission is denied for the `mks` ACL, you can enable configuration management administration of the Integrity Server by allowing the `AdminServer` permission for the `mks:si` ACL.
- *mks:aa*--controls the Login access to the AA application for managing the ACLs.
- *mks:aa:mks*--controls Read and Update access to the ACLs.
- *mks:im* controls access to workflow and document operations.
- *mks:si*--controls access to configuration management operations.
- *mks:sd*--controls access to Deploy operations.
- *mks:patch* controls the Download permission required for service pack management.
- *mks:system:viewsets*--controls access to publishing ViewSets.
- *mks:system:mksdomain*--controls access to administering the MKS domain.

For the most part, you will 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 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>*.
- Archive ACLs simply specify the archive name in the ACL name, such as *mks:si:archive:<archive path>*.

ACL PERMISSIONS

You must have the appropriate ACL permissions before you can perform configuration management, and workflow and document operations. For details on configuring ACLs, see the *PTC Integrity Server Installation and Configuration Guide*.

Integrity Server Permissions

The following summarizes the Integrity Server permissions available under `mks`:

AdminProxy

For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

DebugProxy
For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the proxy.

Prerequisites: none.

DebugServer
For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the server.

Prerequisites: none.

Login
Allows a user to log in to Integrity.

Prerequisites: none.

Configuration Management Server Permissions

The following summarizes the configuration management server-related permissions available under `mks:si`:

AdminProxy
For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer
For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

DebugProxy
For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the proxy.

Prerequisites: none.

DebugServer
For PTC - Integrity Support only. Allows a user to perform diagnostic functions on the server.

Prerequisites: none.

EditPolicy
Allows a user to modify and create configuration management policies on the Integrity Server. The Edit Policy permission should be restricted to administrators and configuration management project managers

Prerequisites: Login.

Login
Allows a user to log in to Integrity.

Prerequisites: none.

StagingSystemAdmin
Required to administer a staging system. Allows a user to perform any Deploy operation in the staging system. Users are automatically granted this permission at the project level when they create a staging system. This permission overrides any global permissions. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: none.

ViewPolicy
Allows a user to view configuration management policies on the Integrity Server. The View Policy permission should be restricted to administrators and configuration management project managers.

Prerequisites: Login.

Configuration Management Member Permissions

The following summarizes the configuration management member-related permissions available under `mks:si`:

ApplyLabel

Allows a user to add labels to revisions or move labels between revisions.

Prerequisites: ***Login***, ***OpenProject***.

CheckIn

Allows a user to check in working files as new revisions of members.

Prerequisites: ***Login***, ***OpenProject***, ***ApplyLabel***, ***Lock***, ***ModifyAuthor***, ***ModifyMemberRev***, ***ModifyMemberAttribute***.

DeleteLabel

Allows users to delete a label from a revision.

Prerequisites: ***Login***, ***OpenProject***.

DeleteRevision

Allows a user to delete revisions from the member history.

Note:

This permission allows users to irrevocably delete revisions from the member history. Administrators should assign this permission carefully.

Prerequisites: ***Login***, ***OpenProject***.

Demote

This permission allows a user to change the promotion state of revisions from a higher setting to a lower one, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

DowngradeOtherUserLock

Allows a user to downgrade exclusive locks held by other users to non-exclusive locks.

Prerequisites: ***Login***, ***OpenProject***.

FetchRevision

Allows a user check out member revisions.

Prerequisites: ***Login***, ***OpenProject***, ***Lock***, ***ModifyMemberRev***.

Freeze

Allows a user to freeze members. When a member is frozen, all configuration management operations are run on the frozen member revision.

Prerequisites: ***Login***, ***OpenProject***.

Lock

Allows a user to lock revisions.

Prerequisites: ***Login***, ***OpenProject***.

ModifyAuthor

Allows a user to change the author name associated with a revision.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberAttribute

Allows a user to set an attribute for a member that can be used later in a search.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberRule

Allows a user to configure a member revision rule that can be applied to one or more members.

Prerequisites: ***Login***, ***OpenProject***.

MoveLabel

Allows a user to move a member label to another revision within the member history.

Prerequisites: ***Login***, ***OpenProject, ApplyLabel***.

Promote

This permission specifies that a user may promote revisions from the current promotion state to a higher state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

ShareArchive

Allows sharing of member archives between two or more members.

Note:

Archive sharing is not recommended. Instead, creating variant Sandboxes is considered a better practice.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint***, ***CheckIn***, ***Lock***.

Thaw

Allows a user to thaw frozen members.

Prerequisites: ***Login***, ***OpenProject***.

Configuration Management Change Package Permissions

The following summarizes the configuration management change package-related permissions available under `mks:si`:

BypassChangePackageMandatory

Allows the user to bypass the Change Packages Mandatory policy, permitting the user to perform configuration management operations without change packages.

Prerequisites: none.

ChangePackageAdmin

Allows a user to edit, discard, close, and submit change packages; as well as move or discard change package unties, regardless of any documented user restrictions.

Prerequisites: none.

CreateChangePackage

If you are using configuration management functionality only, this permission allows a user to create change packages.

If you are using configuration management, and workflow and document functionality, this permission allows a user to create change packages based on the Change Package Creation Policy for the type that they want to create change packages for.

Prerequisites: ***Login***, ***OpenProject***.

PromoteCP

Required to promote a change package in the staging system corresponding to the ACL. Used to authorize use of the **`sd promotecp`** command. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: none.

SelfReview

Allows user to accept change packages under review that were created by that user.

SuperReview

Allows a user to accept or reject a change package under review regardless of the reviewer rules. Note: This permission supersedes the SelfReview permission.

Configuration Management Project Permissions

The following summarizes the configuration management project-related permissions available under `mks:si`:

AddMember
Allows a user to add new members to projects through a Sandbox.

Prerequisites: ***Login***, ***OpenProject***, ***Lock***, ***ShareArchive***, ***ModifyAuthor***.

AddProject
Allows a user to re-add a dropped project.

Prerequisites: ***Login***, ***OpenProject***.

AddSubproject
Allows a user to re-add dropped subprojects to a project.

Prerequisites: ***Login***, ***OpenProject***.

ApplyProjectLabel
Allows a user to add labels to projects or move labels between revisions of the project.

Prerequisites: ***Login***, ***OpenProject***.

CheckPoint
Allows a user to check in a new revision of a project (that is, checkpoint the project).

Prerequisites: ***Login***, ***OpenProject***, ***ApplyLabel***, ***Promote***, ***PromoteProject***, ***ApplyProjectLabel***.

ConfigureSubproject
Allows a user to configure a subproject's type. A subproject can be configured as a Normal, Variant, or Build subproject.

Prerequisites: ***Login***, ***OpenProject.***

CreateDevPath
Allows a user to create new development paths for variants of a project.

Prerequisites: ***Login***, ***OpenProject***.

CreateProject
Allows a user to create new projects.

Prerequisites: ***Login***, ***OpenProject***.

CreateSubproject
Allows a user to create new subprojects below existing projects.

Prerequisites: ***Login***, ***OpenProject***.

DeleteProjectLabel
Allows a user to delete a label from a project checkpoint.

Prerequisites: ***Login***, ***OpenProject***.

DemoteProject
This permission specifies that a user may demote projects from a higher promotion state to a lower state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

Deploy
Required to perform deploy request operations and deploy change packages in the staging system corresponding to the ACL. Used to authorize use of `sd startdeployrequest`, `sd stopdeployrequest`, `sd canceldeployrequest`, `sd editdeployrequest`, and **`sd deploycp`** commands. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

DropDevPath

Allows a user to drop a development path, also known as "dropping variants" from a project.

Prerequisites: ***Login***, ***OpenProject***.

DropMember

Allows a user to remove members from projects. The member archive remains, but the member is no longer treated as part of the project.

Prerequisites: ***Login***, ***OpenProject***.

DropProject

Allows a user to drop one or more top-level, registered projects from the server. The projects then become unregistered projects.

Prerequisites: ***Login***, ***OpenProject***.

DropSubProject

Allows a user to drop one or more subprojects from the server. The projects then become unregistered projects.

Prerequisites: ***Login***, ***OpenProject***.

ImportProject

This permission is for legacy functionality that no longer exists in the product.

Metrics

Allows metrics to be tracked for a project. Allows a user to define metrics to be tracked for projects.

Prerequisites: ***Login***, ***OpenProject***.

ModifyMemberRev

Allows a user to make changes to the member revision of members.

Prerequisites: ***Login***, ***OpenProject***.

ModifyProjectAttribute

Allows a user to set an attribute for a project, which can be used later in a filter or search.

Prerequisites: ***Login***, ***OpenProject***.

MoveProjectLabel

Allows a user to move a project label to another project checkpoint within the project history.

Prerequisites: ***Login***, ***OpenProject***, ***ApplyProjectLabel***.

OpenProject

Allows a user to open existing registered projects. This is required for most actions.

Note:

When OpenProject is granted or denied on a project, clients accessing the project must disconnect and then reconnect in order to get the new permission set. If you do not disconnect and reconnect your client, you may see unexpected behavior due to out-of-date permissions.

Prerequisites: ***Login***.

PromoteProject

This permission specifies that a user may promote projects from the current promotion state to a higher state, when the ***States=*** configuration option defines a sequence of promotion states. For details, see the *PTC Integrity Server Installation and Configuration Guide*.

Prerequisites: ***Login***, ***OpenProject***.

RestoreProject

Allows a user to restore a project to a particular checkpointed version.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint***.

SnapshotSandbox

Snapshot creates and records the state of the user's Sandbox as as a project checkpoint that you can create a build Sandbox or a development path from.

Prerequisites: ***Login***, ***OpenProject***, ***Checkpoint, AddMember, DropMember***.

StagingSystemAdmin

Required to administer a staging system. Allows a user to perform any Deploy operation in the staging system. Users are automatically granted this permission at the project level when they create a staging system. This permission overrides any global permissions. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

ViewDeployRequest

Required to view a deploy request in the staging system corresponding to the ACL. Used to authorize use of the **`sd viewdeployrequest`** command. To use this permission you must be licensed for Deploy. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: ***Login***, ***OpenProject***.

Workflow and Document Permissions

The following summarizes the workflow and document permissions available under `mks:im`:

Admin

Allows access to administrative functions related to workflows and documents. For super administrator whose tasks include managing users, groups, projects, states, types, and fields. Assign project administrators and type administrators. Customize permissions for change package types. Close change package initiated by another user. Create admin reports, dashboards, charts, and queries. Share reports, dashboards, charts, and queries created by another user, if shared to you. If the object is shared to you, you can also delete it. Create and clear Integrity Server alert messages.

Prerequisites: ***Login***.

AdminProxy

For PTC - Integrity Support only. Allows a user to perform administrative functions on the proxy

Prerequisites: none.

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

CreateChart

Allows the assigned user or group to create a chart. Denying permission restricts the user or group to using only those charts already on system. For information on charts, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateCPType

Allows the assigned user or group to create a custom change package type. For information on custom change package types, contact PTC - Integrity Support.

Prerequisites: ***Login***.

CreateDashboard

Allows the assigned user or group to create a dashboard. Denying permission restricts the user or group to using only those dashboards already on system. For information on dashboards, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateProject

Allows the assigned user or group to create a new top level project for workflows and documents, and assign another Project Administrator. This permission can be used to extend the capability of the Project Administrator. Denying this permission means the user cannot create a new top level project or assign another Project Administrator.

Prerequisites: ***Login***.

CreateQuery

Allows a user to create a new query. Denying this permission restricts the user to using only those queries that already exist on the system.

Prerequisites: ***Login***.

CreateReport

Allows the assigned user or group to create a report. Denying permission restricts the user or group to using only those reports already on system. For information on reports, see your *PTC Integrity User Guide*.

Prerequisites: ***Login***.

CreateSharedAdmin

Allows a user to specify if a query, dashboard, report, or chart is a system provided object.

Prerequisites: ***Login***.

CreateType

Allows the assigned user or group to create a new type or assign another Type Administrator. This permission can be used to extend the capability of the Type Administrator. Denying this permission means the user cannot create any new types or assign another Type Administrator.

Prerequisites: ***Login***.

DeleteItem

Delete items of any type.

Prerequisites: Login.

Login

Allows a user to login to Integrity.

Prerequisites: none.

ModifyDeleteItemRule

Allows a user to set a type rule that specifies which users and groups can delete items of that type.

Prerequisites: ***Login***, ***CreateType***.

ModifyMyNotification

Allows a user to modify personal e-mail notification preferences.

Prerequisites: ***Login***, ***ViewMyNotification***.

PurgeTestResult

Allows a user to purge test results for the test cases in a test session.

Prerequisites: ***Login***.

ShareToEveryone

Allows a user to share queries, charts, and reports to the `Everyone` group.

Prerequisites: ***Login***.

TimeTrackingAdmin

Allows a user to create, edit, and delete time entries on behalf of other users. The ability to create, edit, and delete time entries is governed by normal issue permissions.

Prerequisites: ***Login***.

ViewAdmin

Allows a user to view administrative information related to workflows and documents.

Prerequisites: ***Login***.

ViewChangePackage

By controlling the user's ability to view change packages when working in Integrity, the `ViewChangePackage` permission provides an additional level of control for accessing information in projects.

ViewMyNotification

Allows a user to view personal e-mail notification preferences.

Prerequisites: ***Login***.

Available Deploy Permissions

The following summarizes the Deploy permissions available under `mks:sd` to perform specific commands:

AdminServer

For PTC - Integrity Support only. Allows a user to perform administrative functions on the server.

Prerequisites: none.

CreateStagingSystem

A global permission for creating staging systems used to authorize use of the **`sd createstagingsystem`** and **`sd copystagingsystem`** commands. For more information, see the *PTC Integrity Administration Guide for Deploy*.

Prerequisites: `Login`, `OpenProject`.

Available MKS Domain Permissions

The following summarizes the MKS Domain permissions available under `mks:system:mksdomain` to perform specific commands:

AdminServer

Allows a user to administer the MKS Domain.

Prerequisites: none.

Available ViewSets Permissions

The following summarizes the ViewSets permissions available under `mks:system:viewsets` to perform specific commands:

PublishNewViewSet

Allows a user to publish ViewSets to the Integrity Server.

Prerequisites: `Login`.

SEE ALSO

Commands:

`aa acls`, **`aa addaclentry`**, **`aa availablepermissions`**, **`aa deleteacl`**, **`aa deleteaclentry`**, **`aa groups`**, **`aa users`**, **`aa viewacl`**

Miscellaneous:

`diagnostics`