

IceGrid Administrative Utilities

IceGrid provides two administrative clients: a command-line tool and a graphical application.

On this page:

- [IceGrid Command Line Utility](#)
 - [Usage](#)
 - [Application Commands](#)
 - [Node Commands](#)
 - [Registry Commands](#)
 - [Server Commands](#)
 - [Service Commands](#)
 - [Adapter Commands](#)
 - [Object Commands](#)
 - [Server Template](#)
 - [Service Template](#)
 - [Configuration](#)
- [IceGrid Graphical Client](#)

IceGrid Command Line Utility

The `icegridadmin` utility is a command-line tool for administering an IceGrid domain. Deploying an application with this utility requires an XML file that defines the descriptors.

Usage

The IceGrid administration tool supports the following command-line options:

```
Usage: icegridadmin [options]
Options:
-h, --help           Show this message.
-v, --version        Display the Ice version.
-e COMMANDS          Execute COMMANDS.
-d, --debug          Print debug messages.
-s, --server          Start icegridadmin as a server (to parse XML
                     files).
-u, --username        Login with the given username.
-p, --password        Login with the given password.
-S, --ssl            Authenticate through SSL.
-r, --replica NAME    Connect to the replica NAME.
```

The `-e` option causes the tool to execute the given commands and then exit without entering an interactive mode. The `-s` option starts `icegridadmin` in a server mode that supports the `IceGrid::FileParser` interface; a proxy for the object is printed to standard output. If neither `-e` nor `-s` is specified, the tool enters an interactive mode in which you issue commands at a prompt.

To communicate with the IceGrid registry, `icegridadmin` establishes an [administrative session](#). The tool uses SSL authentication if you specify the `-S` option or define its equivalent property `IceGridAdmin.AuthenticateUsingSSL`. Otherwise, `icegridadmin` uses password authentication and prompts you for the username and password if you do not specify them via command-line options or properties. If you want `icegridadmin` to establish its session using a [Glacier2 router](#), define `Ice.Default.Router` appropriately. See [IceGrid Administrative Client Properties](#) for more information on the tool's configuration properties.

Once the session is successfully established, `icegridadmin` displays its command prompt. The `help` command displays the following usage information:

- `help`
Print this message.
- `exit, quit`
Exit this program.
- `CATEGORY help`
Print the help section of the given CATEGORY

- `COMMAND help`
Print the help of the given `COMMAND`.

The tool's commands are organized by category. The supported command categories are shown below:

- `application`
- `node`
- `registry`
- `server`
- `service`
- `adapter`
- `object`
- `server template`
- `service template`

You can obtain more information about each category using the `help` command:

```
>>> application help
```

Application Commands

- `application add [-n | --no-patch] DESC [TARGET ...] [NAME=VALUE ...]`
Add applications described in the XML descriptor file `DESC`. If specified the optional `targets` are deployed. `Variables` are defined using the `NAME=VALUE` syntax. The application is automatically `patched` unless the `-n` or `--no-patch` option is used to disable it.
- `application remove NAME`
Remove the application named `NAME`.
- `application describe NAME`
Describe the application named `NAME`.
- `application diff DESC [TARGET ...] [NAME=VALUE ...]`
Print the differences between the application in the XML descriptor file `DESC` and the current deployment. `Variables` are defined using the `NAME=VALUE` syntax.
- `application update DESC [TARGET ...] [NAME=VALUE ...]`
Update the application in the XML descriptor file `DESC`. `Variables` are defined using the `NAME=VALUE` syntax.
- `application patch [-f | --force] NAME`
`Patch` the application named `NAME`. If `-f` or `--force` is specified, IceGrid will first shut down any servers that depend on the data to be patched.
- `application list`
List all deployed applications.

Node Commands

- `node list`
List all registered nodes.
- `node describe NAME`
Show information about node `NAME`.
- `node ping NAME`
Ping node `NAME`.
- `node load NAME`
Print the load of the node `NAME`.
- `node processors [NAME]`
Print the number of processor sockets for node `NAME`. If `NAME` is omitted, print the number of processor sockets for each node. (The `IceGrid.Node.ProcessorSocketCount` property allows you to explicitly set this value for systems where the number of sockets cannot be obtained programatically.)
- `node show [OPTIONS] NAME [stderr | stdout]`
Print the text from the node's standard error or standard output. The supported options are shown below:
 - `-f, --follow`
Wait for new text to be available.
 - `-t, --tail N`
Print the last `N` lines of text.

- `-h, --head N`
Print the first `N` lines of text
- `node shutdown NAME`
Shutdown node `NAME`.

Registry Commands

- `registry list`
List all registered registries.
- `registry describe NAME`
Show information about registry `NAME`.
- `registry ping NAME`
Ping registry `NAME`.
- `registry show [OPTIONS] NAME [stderr | stdout]`
Print the text from the registry's standard error or standard output. The supported options are shown below:
 - `-f, --follow`
Wait for new text to be available.
 - `-t, --tail N`
Print the last `N` lines of text.
 - `-h, --head N`
Print the first `N` lines of text.
- `registry shutdown NAME`
Shutdown registry `NAME`.

Server Commands

- `server list`
List all registered servers.
- `server remove ID`
Remove server `ID`.
- `server describe ID`
Describe server `ID`.
- `server properties ID`
Get the run-time properties of server `ID`.
- `server property ID NAME`
Get the run-time property `NAME` of server `ID`.
- `server state ID`
Get the state of server `ID`.
- `server pid ID`
Get the process ID of server `ID`.
- `server start ID`
Start server `ID`.
- `server stop ID`
Stop server `ID`.
- `server patch ID`
[Patch](#) server `ID`.
- `server signal ID SIGNAL`
Send `SIGNAL` (such as `SIGTERM` or `15`) to server `ID`.
- `server stdout ID MESSAGE`
Write `MESSAGE` on server `ID`'s standard output.
- `server stderr ID MESSAGE`
Write `MESSAGE` on server `ID`'s standard error.
- `server show [OPTIONS] ID [stderr | stdout | LOGFILE]`
Print the text from the server's standard error, standard output, or the log file `LOGFILE`. The supported options are shown below:

- `-f, --follow`
Wait for new text to be available.
- `-t, --tail N`
Print the last `N` lines of text.
- `-h, --head N`
Print the first `N` lines of text.
- `server enable ID`
Enable server `ID`.
- `server disable ID`
Disable server `ID` (a disabled server can't be started on demand or administratively).

Service Commands

- `service start ID NAME`
Starts service `NAME` in IceBox server `ID`.
- `service stop ID NAME`
Stops service `NAME` in IceBox server `ID`.
- `service describe ID NAME`
Describes service `NAME` in IceBox server `ID`.
- `service properties ID NAME`
Get the run-time properties of service `NAME` from IceBox server `ID`.
- `service property ID NAME PROPERTY`
Get the run-time property `PROPERTY` of service `NAME` from IceBox server `ID`.
- `service list ID`
List the services in IceBox server `ID`.

Adapter Commands

- `adapter list`
List all registered adapters.
- `adapter endpoints ID`
Show the endpoints of adapter or replica group `ID`.
- `adapter remove ID`
Remove adapter or replica group `ID`.

Object Commands

The `object` command operates on [well-known objects](#).

- `object add PROXY [TYPE]`
Add a well-known object to the registry, optionally specifying its type.
- `object remove IDENTITY`
Remove a well-known object from the registry.
- `object find TYPE`
Find all well-known objects with the type `TYPE`.
- `object describe EXPR`
Describe all well-known objects whose stringified identities match the expression `EXPR`. A trailing wildcard is supported in `EXPR`, for example `"object describe Ice*"`.
- `object list EXPR`
List all well-known objects whose stringified identities match the expression `EXPR`. A trailing wildcard is supported in `EXPR`, for example `"object list Ice*"`.

Server Template

- `server template instantiate APPLICATION NODE TEMPLATE [NAME=VALUE ...]`
Instantiate the requested [server template](#) defined in the given application on a node. [Variables](#) are defined using the `NAME=VALUE` syntax.

- `server template describe APPLICATION TEMPLATE`
Describe a [server template](#) TEMPLATE from the given application.

Service Template

- `service template describe APPLICATION TEMPLATE`
Describe a [service template](#) TEMPLATE from the given application.

Configuration

`icegridadmin` requires that the locator proxy be defined in the configuration property `Ice.Default.Locator`. If a configuration file already exists that defines this property, you can start `icegridadmin` using the configuration file as shown below:

```
$ icegridadmin --Ice.Config=<file>
```

Otherwise, you can define the property on the command line:

```
$ icegridadmin --Ice.Default.Locator=<proxy>
```

Refer to the discussion of our [ripper client](#) for more information on configuring the `Ice.Default.Locator` property for an IceGrid client.

IceGrid Graphical Client

The graphical administration tool, IceGrid Admin, allows you to perform anything that you can do from the command line via a GUI. Please refer to the instructions included with your Ice distribution for details on how to start the administration tool.

See Also

- [IceGrid Administrative Sessions](#)
- [Glacier2 Integration with IceGrid](#)
- [IceGrid XML Features](#)
- [Using Descriptor Variables and Parameters](#)
- [Application Distribution](#)
- [Getting Started with IceGrid](#)
- [IceGrid Administrative Client Properties](#)