## Contents

### Chapter 1: Overview
- Logging In ................................................................. 1-2
- Using cmtool .................................................................. 1-2
- Using runAgentCmd ...................................................... 1-2
- Global Arguments (Optional) .......................................... 1-2

### Chapter 2: Agent Management
- changeAgentsEnabled .................................................. 2-2
- createAgentComment ................................................... 2-2
- createResource ............................................................ 2-3
- createResourceComment ............................................... 2-3
- deleteAgentComment .................................................... 2-4
- deleteAgents ............................................................... 2-4
- deleteResource ............................................................. 2-5
- deleteResources ........................................................... 2-5
- deleteResourceComment ............................................... 2-5
- getAgentComments ...................................................... 2-6
- getAgentPerformance .................................................. 2-6
- getAgents ..................................................................... 2-7
- getAgentStatus ............................................................ 2-9
- getLsfInformation ......................................................... 2-10
- getLsfJobs ................................................................... 2-10
- getResource ................................................................. 2-11
- getResources .................................................................. 2-11
- getResourceComments .................................................. 2-12
- modifyAgentComment ................................................... 2-13
- modifyBuildClassComment ............................................ 2-13
- modifyResource ........................................................... 2-13
- modifyResourceComment .............................................. 2-14
- setAgentDebug ............................................................. 2-14

### Chapter 3: Build Management
- createBuildClass .......................................................... 3-2
- createBuildClassComment .............................................. 3-2
- createBuildComment .................................................... 3-3
- deleteBuild ................................................................. 3-3
- deleteBuildClass .......................................................... 3-3
- deleteBuildClasses ........................................................ 3-4
- deleteBuildClassComment ............................................. 3-4
- deleteBuildComment ..................................................... 3-4
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteBuilds</td>
<td>3-5</td>
</tr>
<tr>
<td>getBuild</td>
<td>3-5</td>
</tr>
<tr>
<td>getBuilds</td>
<td>3-6</td>
</tr>
<tr>
<td>getBuildComments</td>
<td>3-9</td>
</tr>
<tr>
<td>getBuildClass</td>
<td>3-10</td>
</tr>
<tr>
<td>getBuildClasses</td>
<td>3-10</td>
</tr>
<tr>
<td>getBuildClassComments</td>
<td>3-12</td>
</tr>
<tr>
<td>getBuildUserStats</td>
<td>3-12</td>
</tr>
<tr>
<td>modifyBuild</td>
<td>3-13</td>
</tr>
<tr>
<td>modifyBuildClass</td>
<td>3-14</td>
</tr>
<tr>
<td>modifyBuildContent</td>
<td>3-14</td>
</tr>
<tr>
<td>setDatabaseConfiguration</td>
<td>3-15</td>
</tr>
<tr>
<td>stopBuild</td>
<td>3-15</td>
</tr>
</tbody>
</table>

**Chapter 4: Cluster Management**

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>createServerComment</td>
<td>4-1</td>
</tr>
<tr>
<td>deleteLicense</td>
<td>4-2</td>
</tr>
<tr>
<td>deleteMessage</td>
<td>4-2</td>
</tr>
<tr>
<td>deleteMessages</td>
<td>4-3</td>
</tr>
<tr>
<td>deleteServerComment</td>
<td>4-3</td>
</tr>
<tr>
<td>exportData</td>
<td>4-3</td>
</tr>
<tr>
<td>getLicense</td>
<td>4-4</td>
</tr>
<tr>
<td>getLicenses</td>
<td>4-4</td>
</tr>
<tr>
<td>getMessage</td>
<td>4-4</td>
</tr>
<tr>
<td>getMessages</td>
<td>4-5</td>
</tr>
<tr>
<td>getResourceStats</td>
<td>4-6</td>
</tr>
<tr>
<td>getServer</td>
<td>4-8</td>
</tr>
<tr>
<td>getServerComments</td>
<td>4-9</td>
</tr>
<tr>
<td>getVersion</td>
<td>4-9</td>
</tr>
<tr>
<td>importData</td>
<td>4-10</td>
</tr>
<tr>
<td>importLicenseData</td>
<td>4-10</td>
</tr>
<tr>
<td>logMessage</td>
<td>4-11</td>
</tr>
<tr>
<td>modifyServer</td>
<td>4-11</td>
</tr>
<tr>
<td>modifyServerComment</td>
<td>4-12</td>
</tr>
<tr>
<td>shutdownServer</td>
<td>4-12</td>
</tr>
<tr>
<td>testAgents</td>
<td>4-12</td>
</tr>
</tbody>
</table>

**Chapter 5: Reporting**

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>createFilter</td>
<td>5-1</td>
</tr>
<tr>
<td>deleteFilter</td>
<td>5-2</td>
</tr>
<tr>
<td>getCurrentServerLoad</td>
<td>5-2</td>
</tr>
<tr>
<td>getFilter</td>
<td>5-3</td>
</tr>
<tr>
<td>getFilters</td>
<td>5-4</td>
</tr>
<tr>
<td>modifyFilter</td>
<td>5-4</td>
</tr>
</tbody>
</table>

**Chapter 6: User Management**

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>addGroupMember</td>
<td>6-1</td>
</tr>
<tr>
<td>changeOwnUser</td>
<td>6-2</td>
</tr>
<tr>
<td>createGroup</td>
<td>6-2</td>
</tr>
</tbody>
</table>
createUser ................................................................. 6-3
deleteGroup .............................................................. 6-3
deleteUser ................................................................. 6-3
getAccessEntries ....................................................... 6-4
getGroupMembers ...................................................... 6-4
getGroups ................................................................. 6-5
getEffectivePermissions .............................................. 6-5
getPermissions .......................................................... 6-6
getUser ..................................................................... 6-7
getUsers ................................................................. 6-7
getUserSettings .......................................................... 6-8
login ...................................................................... 6-8
logout ................................................................. 6-9
modifyGroup ............................................................ 6-9
modifyUser ............................................................... 6-9
removeGroupMember .................................................. 6-10
setBuildEndNotification ............................................. 6-10
setPermissions .......................................................... 6-10
setUserSettings .......................................................... 6-11
**Chapter 1: Overview**

**cmtool** is the ElectricAccelerator® command-line tool. **cmtool** provides access to the Cluster Manager through a command-line interface instead of using the web interface. With cmtool, you can write Perl scripts to access Cluster Manager information or manage builds. Almost all ElectricAccelerator operations and tasks can be implemented with cmtool—with the exception of a few reports that are generated only from the web interface.

**cmtool** is used primarily for build and agent management, including commands for build class management, agent testing, and adding comments automatically.

**Topics:**

- Logging In
- Using cmtool
- Using runAgentCmd
- Global Arguments (Optional)
Logging In

If you use cmtool outside of a job, you must invoke the **cmtool login** command to log in to the server. After logging in, cmtool saves information about the login session for use in future cmtool invocations. If you run cmtool as part of an ElectricAccelerator job, you do not need to log in because **cmtool** uses the login session (and credentials) for that job.

To Log In to cmtool:

```
    cmtool login <username> <password>
```

To specify a session file, use the **--sessionFile=<fileName>** option, so you can use the same session for subsequent cmtool invocations.

Using cmtool

An invocation of cmtool identifies the Cluster Manager to contact, using the **--server** command-line option, followed by a list of commands to execute. Certain commands may have optional or required arguments.

For example, the following invocation receives all build requests that ran fewer than 10 jobs and orders the list [that ran the build] by host name.

```
    cmtool --server easerver getBuilds-filter "job_count <10" --order host_name
```

General syntax for cmtool command usage:

```
    cmtool [optional global argument(s)] <command> <required arguments> [optional arguments]
```

Return Codes

- 0 = success (the command was correct; if no data meets the criteria, return is still 0)
- 1 = failure (command was invalid)

Using runAgentCmd

**IMPORTANT:** Exercise caution when using the **runAgentCmd** command. Electric Cloud recommends using this command for documented scenarios only or under the direction of Electric Cloud Technical Support.

The **runAgentCmd** command enables you to run agent commands against the cluster.

Use this format: **cmtool --cm=<cm> runAgentCmd "agent command to run"**

where **<cm>** is the IP address or name of your Cluster Manager.

Some of the possible reasons for using **runAgentCmd** include:

- Setting agent-side breakpoints (see the Using Breakpoints topic in online help)
- Configuring agent log rotation (see the Installation and Configuration Guide)
- Getting and setting agent and EFS debug levels (Knowledge Base article KBEA-00020)
- Configuring the stalled job killer (Knowledge Base article KBEA-00031)
- Troubleshooting builds that appear to hang (Knowledge Base article KBEA-00036)

Global Arguments (Optional)

Global arguments supply general information quickly, including cmtool online help.

**Note:** Global arguments support using the "=" sign character.
• **--help [command]** - Prints this message and exits. If a command is specified, prints the help text for that command.

• **--help-commands** - Prints the list of available commands with a short description.

• **--help-fields <command>** - Displays a list of fields for a command—requires the <command> argument.

• **--version** - Prints cmtool version number.

• **--server <hostname>** - ElectricAccelerator server address. Defaults to the ACCELERATOR_SERVER environment variable. If this variable does not exist, default is to the localhost.

• **--port <port>** - HTTP listener port on the ElectricAccelerator server. Defaults to port 8030.

• **--securePort <securePort>** - HTTPS listener port on the ElectricAccelerator server. Defaults to port 8031.

• **--secure** - Uses HTTPS to communicate with the ElectricAccelerator server.

• **--timeout <s>** - cmtool waits for a response from the server for a specified amount of time. Timeout for server communication defaults to 180 seconds (3 minutes) if no other time is specified. After the timeout, cmtool exits but the server will continue to process the command.

• **--output <style>** - Set output style—default is 'xml'. 'xml' for an XML document; 'csv' for comma separated values; 'simple' for no formatting; 'silent' for no output.

• **--fields <list>** - List is a comma separated list of fields to emit when using 'csv' or 'simple' output styles. Default is all fields.

• **--sessionFile <path>** - Overrides the location where session information will be stored.
Chapter 2: Agent Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

- changeAgentsEnabled
- createAgentComment
- createResource
- createResourceComment
- deleteAgentComment
- deleteAgents
- deleteResource
- deleteResources
- deleteResourceComment
- getAgentComments
- getAgentPerformance
- getAgents
- getAgentStatus
- getLsfInformation
- getLsfJobs
- getResource
- getResources
- getAgentComments
- modifyAgentComment
- modifyBuildClassComment
- modifyResource
- modifyResourceComment
- setAgentDebug
changeAgentsEnabled

Description
Changes the agent enabled status of one or more agents.

Required Arguments
- `<enabled>` - Possible values are `true` or `false`.

Optional Arguments
Note: If no agent name, agent ID, or filter is specified, all agents are changed.
- `--agentId <unique, internal number that can change, assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`
- `--filter <SQL query used to limit the result set>`. For a list of possible SQL values, see the `getAgents` command.

Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

Examples
- `cmtool changeAgentsEnabled false`
  Disables all agents in the cluster.
- `cmtool changeAgentsEnabled true --agentName linagent1`
  Enables the agent named “linagent1”.
- `cmtool changeAgentsEnabled true --filter "agent_name LIKE 'winbuild1-%'"
  Enables all agents with a name that begins with "winbuild1-".

createAgentComment

Description
Creates a new agent comment.

Required Arguments
Note: Either `agentId` or `agentName` must also be specified.
- `<text>` - The text of the item.

Optional Arguments
- `--agentId <unique, internal number that can change, assigned by the Cluster Manager>`
Chapter 2: Agent Management

- **--agentName** <name defined by the host where the agent resides [numbers and/or letters]>

**Example**
cmtool createAgentComment --agentName linagent "Agent has been running great"

Creates a comment for an agent named "linagent".

**createResource**

**Description**
Creates a new resource definition. After creating a resource, ensure the server is configured to support resource management. You can use the `modifyServer` command to enable resource management.

**Required Arguments**
- `<resourceName>` - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class. It is used in the `ea_resource` table and also matches the resource requirement string for eMake.
- `<hostMasks>` - This is a quote-enclosed, semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. "*" is the wildcard character.

**Optional Arguments**
- **--description** <a quote-enclosed text description for your reference only>

**Example**
cmtool createResource R29 "rs*; rt*" --description "rs or rt hosts"

Creates a new resource named R29 that only uses hosts whose names start with 'rs' or 'rt'.

**createResourceComment**

**Description**
Creates a new resource comment.

**Required Arguments**
- `<resourceId>` - A unique number that identifies each resource.
- `<text>` - The text of the item.

**Optional Arguments**
None

**Example**
cmtool createResourceComment 2 "This resource identifies production servers"

Creates a comment for resource 2.
deleteAgentComment

Description
Deletes an agent comment.

Required Arguments
Note: Either agentId or agentName must also be specified.

- <commentId> - The unique key that identifies a comment.
  Use getAgentComments to get a list of commentId numbers.

Optional Arguments

- --agentId <unique, internal number that can change, assigned by the Cluster Manager>
- --agentName <name defined by the host where the agent resides [numbers and/or letters]>

Example

cmtool deleteAgentComment 1008 --agentId 14

Deletes comment 1008 from agent 14 (14 is the Cluster Manager internal ID for the agent). To find out what the appropriate comment ID is, use the getAgentComments command, which will list the comments attached to a particular agent.

deleteAgents

Description
Deletes one or more agents, including all dependent records.

Required Arguments
None

Optional Arguments

- --agentId <unique, internal number that can change, assigned by the Cluster Manager>
- --agentName <name defined by the host where the agent resides [numbers and/or letters]>
- --filter <SQL query used to limit the result set> - For a list of possible SQL values, see the getAgents command.

Example

cmtool deleteAgents --agentName winbuild1

Deletes agent "winbuild1" and all associated comments.
deleteResource

Description
Deletes a resource definition.

Required Arguments
- `<resourceId>` - A unique number that identifies each resource.
  Use the `getResources` command to get a list of resource IDs.

Optional Arguments
None

Example
cmtool deleteResource 3
Deletes the resource definition for resource 3.

deleteResources

Description
Deletes multiple resource definitions.

Required Arguments
None

Optional Arguments
- `--filter <SQL query used to limit the result set>` - For a list of possible SQL values, see the `getResources` command.

Example
cmtool deleteResources
Deletes all resource definitions.

deleteResourceComment

Description
Deletes a resource comment.

Required Arguments
- `<resourceId>` - A unique number that identifies each resource.
- `<commentId>` - The unique key that identifies a comment.
  Use the `getResourceComments` command to get a list of comment IDs.

Optional Arguments
None
Example

cmtool deleteResourceComment 3 49

Deletes comment 49 from resource 3.

**getAgentComments**

*Description*

Retrieves a list of related agent comments, or a specific comment (by using the `--commentId` option).

*Required Arguments*

None

*Optional Arguments*

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`

*Result Tags*

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

cmtool getAgentComments --agentName ahost-3

Retrieves all comments for agent “ahost-3”.

**getAgentPerformance**

*Description*

Retrieves the performance log of one or more agents.

*Required Arguments*

None

*Optional Arguments*

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`
- `--agents <a list of agents whose performance you want to see>`
Chapter 2: Agent Management

- --buildId <this further restricts the returned agents to those running a specific build ID>
- --status <1|0>- Choose active or inactive agents.
- --enabled <1|0>- Choose enabled or disabled agents only.

**Result Tags**

- agentName - This is the name of the agent as it appears on the web page (product UI).
- result - This is the performance information of the agent.

**Example**

cmtool getAgentPerformance --agentName SOL1-1

Returns the performance log of the agent named “SOL1-1”.

**getAgents**

**Description**

Retrieves a list of agents.

**Required Arguments**

None

**Optional Arguments**

- --agentId <unique, internal number that can change; assigned by the Cluster Manager>
- --agentName <name defined by the host where the agent resides [numbers and/or letters]>
- --filter <SQL query used to limit the result set>- See possible values in the table below.
  
  **Note:** There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using --filter for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- --maxResults <maximum number of elements to run from a query>
- --firstResult <starting index for the query result set>
  
  **Note:** --firstResult takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.
- --order <SQL order by clause>- Used to specify ordering for the query result set.
- --profile <details|info>- This is the level of detail to return from a query; details gets all information and info gets a reduced information set.

**Result Tags and SQL Query Names**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
</table>

2-7
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>a2aPort</td>
<td>The agent to agent protocol communication port.</td>
<td>a2a_port</td>
</tr>
<tr>
<td>agentId</td>
<td>A unique, internal number assigned to each agent by the Cluster Manager; this number can change.</td>
<td>id</td>
</tr>
<tr>
<td>agentName</td>
<td>A name defined by the host where the agent resides [numbers and/or letters].</td>
<td>agent_name</td>
</tr>
<tr>
<td>agentVersion</td>
<td>The agent version string.</td>
<td>agent_version</td>
</tr>
<tr>
<td>availableResults</td>
<td>This is a count of ‘max’ or ‘first’ results if --maxResults or --firstResult is specified.</td>
<td>N/A</td>
</tr>
<tr>
<td>buildId</td>
<td>A unique number assigned by the Cluster Manager for each build.</td>
<td>current_build_id</td>
</tr>
<tr>
<td>buildName</td>
<td>The build name that is the expanded build class tag.</td>
<td>N/A</td>
</tr>
<tr>
<td>consolePort</td>
<td>The agent console port.</td>
<td>console_port</td>
</tr>
<tr>
<td>efsVersion</td>
<td>The EFS version string.</td>
<td>efs_version</td>
</tr>
<tr>
<td>enabled</td>
<td>The flag indicating if an agent is enabled or not.</td>
<td>enabled</td>
</tr>
<tr>
<td>errorCount</td>
<td>The number of internal agent errors.</td>
<td>error_count</td>
</tr>
<tr>
<td>hostName</td>
<td>The name of the machine where Electric Make was invoked.</td>
<td>host_name</td>
</tr>
<tr>
<td>inPenaltyBox</td>
<td>A flag indicating Electric Make had a recent problem with this agent.</td>
<td>N/A</td>
</tr>
<tr>
<td>ipAddress</td>
<td>The agent IP address.</td>
<td>ip_address</td>
</tr>
<tr>
<td>lastErrorTime</td>
<td>The last time the agent experienced an error.</td>
<td>last_error_time</td>
</tr>
<tr>
<td>lastPingTime</td>
<td>The last time the agent was pinged to determine its status.</td>
<td>last_ping_time</td>
</tr>
<tr>
<td>platform</td>
<td>The operating system being used or supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.</td>
<td>platform</td>
</tr>
</tbody>
</table>

ElectricAccelerator
## Chapter 2: Agent Management

### port

<table>
<thead>
<tr>
<th>port</th>
<th>The agent protocol communication port.</th>
<th>port</th>
</tr>
</thead>
</table>

### restartCount

<table>
<thead>
<tr>
<th>restartCount</th>
<th>The number of agent restarts.</th>
<th>restart_count</th>
</tr>
</thead>
</table>

### status

<table>
<thead>
<tr>
<th>status</th>
<th>The agent status. 1 = OK, but anything else is an error code.</th>
<th>status</th>
</tr>
</thead>
</table>

### statusDetail

<table>
<thead>
<tr>
<th>statusDetail</th>
<th>If the last status update resulted in an error, it contains the error string (or the &quot;OK&quot; string if no error occurred).</th>
<th>status_detail</th>
</tr>
</thead>
</table>

### webPort

<table>
<thead>
<tr>
<th>webPort</th>
<th>The agent web server port.</th>
<th>web_port</th>
</tr>
</thead>
</table>

### Example

```plaintext
example

```
cmtool getAgents --filter "agent_name like '%SOL%'"
```
Retrieves a list of all agents whose names start with “SOL”.

### getAgentStatus

#### Description

Retrieves the state of one or more agents. By default, only active agents are returned. Use `--status 0` to list inactive agents.

#### Required Arguments

None

#### Optional Arguments

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`
- `--agents <this can be a list of agents whose status you want to see>`
- `--buildId <this further restricts the returned agents to those running a specific build ID>`
- `--status <1|0> - Choose active or inactive agents.
- --enabled <1|0> - Choose enabled or disabled agents only.```

#### Result Tags

- `agentName` - This is the name of the agent as it appears on the web page (product UI).
- `result` - This is the text string that describes the current state of the agent.

### Example

```plaintext
example

cmtool getAgentStatus --agentName SOL1-1
```

Returns the status of the agent named “SOL1-1”.

---

2-9
getLsflInformation

Description
Retrieves current information about the LSF interface.

Note: LSF must be enabled to retrieve information.

Required Arguments
None

Optional Arguments
None

Result Tags
- clusterName - The name of the LSF grid cluster.
- lsfAvailable - "1" if LSF is available to the Cluster Manager.
- masterName - The LSF Master Host name.
- numPendingAgentJobs - The number of LSF jobs submitted by the Cluster Manager that are waiting to run.
- numRunningAgentJobs - The number of LSF jobs Cluster Manager submitted that are running now.
- statusMessage - A message.

Example

cmtool --output csv --fields getLsflInformation

Retrieves a Boolean value to indicate whether LSF is available or not.

getLsfJobs

Description
Retrieves information about all jobs submitted to LSF.

Required Arguments
None

Optional Arguments
None

Result Tags
- agentHostName - The machine name where the agent is running.
- jobNumber - The job number referencing a batch job submitted to LSF.
- jobStatus - The current status of an LSF job.
- resourceRequest - A request to the resource manager for a particular type of agent.
- submitTime - The time the job was submitted to LSF.
Example
cmtool getLsfJobs
Retrieves all LSF job information.

getResource

Description
Finds a resource with full detail by the resource ID number.

Required Arguments

- `<resourceId>` - A unique number that identifies each resource.
  Use `getResources` to retrieve a list of resource IDs.

Optional Arguments
None

Result Tags

- `hostMasks` - This is a semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. "*" is the wildcard character.
- `resourceId` - A unique number that identifies each resource.
- `resourceName` - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class.

Example
cmtool getResource 7
Retrieves resource 7.

getResources

Description
Retrieves a list of all resources.

Required Arguments
None

Optional Arguments

- `--filter <SQL query used to limit the result set>`
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- `--maxResults <maximum number of elements to run from a query>`
- **firstResult** `<starting index for the query result set>`
  
  **Note**: `--firstResult` takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.

- **order** `<SQL order by clause>` - Used to specify ordering for the query result set.

- **profile** `<details|info>` - This is the level of detail to return from a query; `details` gets all information and `info` gets a reduced information set.

### Result Tags and SQL Query Names

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
<tbody>
<tr>
<td>availableResults</td>
<td>This is a count of <code>max</code> or <code>first</code> results if <code>--maxResults</code> or <code>--firstResult</code> is specified.</td>
<td>N/A</td>
</tr>
<tr>
<td>hostMasks</td>
<td>This is a semi-colon delimited list of host name masks, used to identify the list of hosts that support a resource. <code>*</code> is the wildcard character.</td>
<td>host_masks</td>
</tr>
<tr>
<td>resourceId</td>
<td>A unique number that identifies each resource.</td>
<td>id</td>
</tr>
<tr>
<td>resourceName</td>
<td>This name is used on the eMake parameter: <code>--emake-resource</code>, and can be specified in a build class.</td>
<td>resource_name</td>
</tr>
</tbody>
</table>

### Example

cmtool getResources --order resource_name

Retrieves a list of resources ordered by the resource name.

**getResourcesComments**

**Description**

Retrieves resource comments.

**Required Arguments**

- `<resourceId>` - A unique number that identifies each resource.

**Optional Arguments**

- `<commentId>` - The unique key that identifies a comment.

### Example

cmtool getResourcesComments 29

Retrieves comments for resource 29.
modifyAgentComment

Description
Modifies an agent comment.

Required Arguments
Note: Either agentId or agentName must also be specified.

- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

Optional Arguments

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`

Example
cmtool modifyAgentComment 1037 "changed comment" --agentName SOL1-1
Changes comment number 1037 on agent SOL1-1 to "changed comment".

modifyBuildClassComment

Description
Modifies a build class comment.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class.
- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

Optional Arguments
None

Example
cmtool modifyBuildClassComment 1037 1129 "This is a low-priority class"

modifyResource

Description
Modifies a resource definition.

Required Arguments

- `<resourceId>` - A unique number that identifies each resource.
**Optional Arguments**

- **--hostMasks** `<a semi-colon delimited list of host name masks>` - This is used to identify the list of hosts that support a resource. "*" is the wildcard character.
- **--resourceName** `<the unique name of the resource>`
- **--description** `<a text description for your reference only>`

**Example**

cmtool modifyResource 27 --hostMasks "SOL*; SRL*"

Sets the host masks for resource 27 to "SOL*; SRL*".

**modifyResourceComment**

**Description**

Modifies a resource comment. Use getResources to retrieve a list of resource IDs.

**Required Arguments**

- `<resourceId>` - A unique number that identifies each resource.
- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

**Optional Arguments**

None

**Example**

cmtool modifyResourceComment 1 1015 "new xxx"

Changes comment 1015 for resource 1.

**setAgentDebug**

**Description**

Sets the agent debug level (see getAgentStatus). This command sends a message to the agent(s) in real time; therefore, the agents must be up and connected to the Cluster Manager to have any effect.

**Required Arguments**

- `<level>` - Possible values are:
all       registry
commands requests
environment state
fileinfo   test
log        usage
other      nothing
profile

**Optional Arguments**

- `--agentId <unique, internal number that can change, assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`
- `--status <1|0>` - Chooses active or inactive agents.
- `--buildId <a specific build ID>` - This further restricts the returned agents to those running a specific build ID.
- `--enabled <1|0>` - Choose enabled or disabled agents only.
- `--agents <host>[:<port>[:<agentKey>]]` - This specifies individual agents based on their host name and listening port.

**Result Tags**

- `agentName` - The name of the configured agent.
- `result` - The configuration result.

**Example**

cmtool setAgentDebug profile --agentName SOLAgent-1

Sets SOLAgent-1’s debug level to “profile”.

Chapter 3: Build Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

- createBuildClass
- createBuildClassComment
- createBuildComment
- deleteBuild
- deleteBuildClass
- deleteBuildClasses
- deleteBuildClassComment
- deleteBuildComment
- deleteBuilds
- getBuild
- getBuilds
- getBuildComments
- getBuildClass
- getBuildClasses
- getBuildClassComments
- getBuildUserStats
- modifyBuild
- modifyBuildClass
- modifyBuildContent
- setDatabaseConfiguration
- stopBuild
createBuildClass

Description
Creates a build class.

Required Arguments
- <buildClassName> - A name assigned by the user for the build class.

Optional Arguments
- --tagDefinition <a format string that defines the resulting build name>
- --annotationLevels <a comma-separated list of values> - Possible values are: basic, history, file, lookup, and waiting.
- --maxAgents <maximum number of agents to request for this build>
- --minAgents <minimum number of agents required for this build to run>
- --platform <operating system being used/supported> - Must be either Windows, Linux, or Solaris. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.
- --priority <priority value> - The default priority value is 120 (normal). 220 is high and 20 is low. Priority value can be adjusted up or down by 1-10 to “boost” the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- --annoUpload <Y|N> - If set to true, the annotation file is uploaded to Cluster Manager. Values can also be 1, 0, true, or false.
- --resourceRequest <value> - This is a request to the resource manager for a particular type of agent. Value is the name of a pre-existing resource.

Example
cmtool createBuildClass batch --minAgents 5 --maxAgents 12 --priority 30 --resourceRequest blades

Creates a build class named batch that requires a minimum of 5 agents and a maximum of 12 agents. The priority is relatively low and the requested resource is named blades.

createBuildClassComment

Description
Creates a new build class comment.

Required Arguments
- <buildClassId> - A unique number assigned by the Cluster Manager for each build class. Use getBuildClass to retrieve a list of build class IDs.
- <text> - The text of the item.

Optional Arguments
None
Example
cmtool createBuildClassComment 7 "This build class is for QA builds."

Creates a comment for build class 7.

createBuildComment

Description
Creates a new build comment.

Required Arguments

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.
  Use `getBuilds` to retrieve a list of build IDs.
- `<text>` - The text of the item.

Optional Arguments
None

Example
cmtool createBuildComment 1044 "This is our gold build for release 7.0"

Creates a comment for build 1044.

deleteBuild

Description
Deletes a build, including all dependent records.

Required Arguments

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.
  Use `getBuilds` to retrieve a list of build IDs.

Optional Arguments
None

Example
cmtool deleteBuild 1037

Deletes build 1037.

deleteBuildClass

Description
Deletes a build class, including all dependent records.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class.
  Use `getBuildClasses` to retrieve a list of build class IDs.
Optional Arguments
None

Example
cmtool deleteBuildClass 7
Deletes build class 7.

deleteBuildClasses

Description
Deletes a set of build classes, including all dependent records.

Required Arguments
None

Optional Arguments

Note: If no filter is provided, all build classes (except the default) will be deleted.

- --filter <SQL query used to limit the result set> - For a list of possible SQL values, see the getBuildClasses command.

Example
cmtool deleteBuildClasses --filter "max_agents >20"
Deletes all build classes with more than 20 maximum agents.

deleteBuildClassComment

Description
Deletes a build class comment.

Required Arguments

- <buildClassId> - A unique number assigned by the Cluster Manager for each build class.
- <commentId> - The unique key that identifies a comment.

Optional Arguments
None

Example
cmtool deleteBuildClassComment 6 1018
Deletes comment 1018 for build class 6.

deleteBuildComment

Description
Deletes a build comment.
**Required Arguments**

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.

- `<commentId>` - The unique key that identifies a comment. Use `getBuildComments` to retrieve a list of comment IDs.

**Optional Arguments**

None

**Example**

cmtool deleteBuildComment 1037 1019

Deletes build comment 1019 for build 1037.

**deleteBuilds**

**Description**

Deletes a set of builds, including all dependent records.

It is important to remove build logs periodically so they do not fill up the Cluster Manager’s available disk space. Uploaded annotation is also considered part of build logs, so remember to clean up build logs regularly if annotation is frequently uploaded to the Cluster Manager.

You can also manage build logs using the Cluster Manager web interface. Select the Builds tab, and then create and run a “Builds by Date” filter to display the set of builds that you want to remove. Click **Delete Filtered Builds** to remove the build logs from disk and from the database.

**Required Arguments**

If no argument is provided, all builds will be deleted.

**Optional Arguments**

- `--filter <SQL query used to limit the result set>` - For a list of possible SQL values, see the `getBuilds` command.

**Example**

cmtool deleteBuilds --filter "start_time <date_sub(curdate( ), interval 20 day)"

Deletes all builds more than 20 days old.

**Note**: This example is valid for MySQL only. If you use a different database, use syntax that is appropriate for your respective database.

**getBuild**

**Description**

Finds a build with full detail by the build’s ID number.

**Required Arguments**

- `<buildId>` - A unique number assigned by the Cluster Manager for each build. Use `getBuilds` to retrieve a list of build IDs.
Optional Arguments

None

Result Tags

See getBuilds for descriptions.

<table>
<thead>
<tr>
<th>Optional Argument</th>
<th>Result Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocatedAgents</td>
<td>ipAddress</td>
</tr>
<tr>
<td>buildClassId</td>
<td>jobCount</td>
</tr>
<tr>
<td>buildClassName</td>
<td>lastRequestTime</td>
</tr>
<tr>
<td>buildId</td>
<td>maxAgents</td>
</tr>
<tr>
<td>buildLogDir</td>
<td>minAgents</td>
</tr>
<tr>
<td>buildName</td>
<td>osUserName</td>
</tr>
<tr>
<td>commandLine</td>
<td>platform</td>
</tr>
<tr>
<td>conflicts</td>
<td>priority</td>
</tr>
<tr>
<td>cwd</td>
<td>resourceRequest</td>
</tr>
<tr>
<td>duration</td>
<td>result</td>
</tr>
<tr>
<td>effectiveAgentAlloc</td>
<td>requestedAgents</td>
</tr>
<tr>
<td>emakeVersion</td>
<td>startTime</td>
</tr>
<tr>
<td>historyExists</td>
<td>userLabel</td>
</tr>
<tr>
<td>historyFile</td>
<td>userName</td>
</tr>
<tr>
<td>hostName</td>
<td>waitTime</td>
</tr>
</tbody>
</table>

Example

cmtool getBuild 1000
Retrieves build 1000.

getBuilds

Description

Retrieves a list of builds.

Required Arguments

None

Optional Arguments

- --filter <SQL query used to limit the result set> - See the possible values below. There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using --filter for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

- --maxResults <maximum number of elements to run from a query>

- --firstResult <starting index for the query result set>
  Note: --firstResult takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.

- --order <SQL order by clause> - Used to specify ordering for the query result set.

- --profile <details|info> - This is the level of detail to return from a query; details gets all information and info gets a reduced information set.
  Note: You must set --profile details in order to print fields that are part of the details category.
### Result Tags and SQL Query Names

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocatedAgents</td>
<td>The number of currently assigned agents for this build.</td>
<td>N/A</td>
</tr>
<tr>
<td>availableResults</td>
<td>This is a count of 'max' or 'first' results if --maxResults or --firstResult is specified.</td>
<td>N/A</td>
</tr>
<tr>
<td>buildClassId</td>
<td>A unique number assigned by the Cluster Manager for each build class.</td>
<td>build_class_id</td>
</tr>
<tr>
<td>buildClassName</td>
<td>A name assigned by the user for the build class.</td>
<td>build_class_name</td>
</tr>
<tr>
<td>buildId</td>
<td>A unique number assigned by the Cluster Manager for each build.</td>
<td>id</td>
</tr>
<tr>
<td>buildLogDir</td>
<td>The directory containing uploaded build logs.</td>
<td>N/A</td>
</tr>
<tr>
<td>buildName</td>
<td>The build name that is the expanded build class tag.</td>
<td>build_name</td>
</tr>
<tr>
<td>commandLine</td>
<td>The original command-line invocation of Electric Make.</td>
<td>command_line</td>
</tr>
<tr>
<td>conflicts</td>
<td>The number of conflicts in the build.</td>
<td>conflicts</td>
</tr>
<tr>
<td>cwd</td>
<td>The current working directory where Electric Make was invoked.</td>
<td>cwd</td>
</tr>
<tr>
<td>duration</td>
<td>The number of milli-seconds the build has been running.</td>
<td>duration</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> duration for running builds is always 0.</td>
<td></td>
</tr>
<tr>
<td>effectiveAgentAlloc</td>
<td>The effective agent allocation percentage. 100% means eMake had all the hosts it needed all the time, while a lesser percentage means eMake had the hosts it needed for that percent of time.</td>
<td>effective_agent_alloc</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> effective_agent_alloc for running builds is always 0.</td>
<td></td>
</tr>
<tr>
<td>emakeVersion</td>
<td>The Electric Make version used for this build.</td>
<td>emake_version</td>
</tr>
<tr>
<td><strong>historyExists</strong></td>
<td>True means the history file existed and was used by the build.</td>
<td><strong>history_exists</strong></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><strong>historyFile</strong></td>
<td>The name of the Electric Make history file.</td>
<td><strong>history_file</strong></td>
</tr>
<tr>
<td><strong>hostName</strong></td>
<td>The name of the machine where Electric Make was invoked.</td>
<td><strong>host_name</strong></td>
</tr>
<tr>
<td><strong>ipAddress</strong></td>
<td>The IP address of the machine where Electric Make was invoked.</td>
<td><strong>ip_address</strong></td>
</tr>
<tr>
<td><strong>jobCount</strong></td>
<td>The total number of jobs that ran for the build.</td>
<td><strong>job_count</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> <em>job_count</em> for running builds is always 0.</td>
<td></td>
</tr>
<tr>
<td><strong>lastRequestTime</strong></td>
<td>The last time Electric Make requested agents for this build.</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>maxAgents</strong></td>
<td>The maximum number of agents to request for this build.</td>
<td><strong>max_agents</strong></td>
</tr>
<tr>
<td><strong>minAgents</strong></td>
<td>The minimum number of agents required for this build to run.</td>
<td><strong>min_agents</strong></td>
</tr>
<tr>
<td><strong>osUserName</strong></td>
<td>The OS-level name for the user who started Electric Make.</td>
<td><strong>os_user_name</strong></td>
</tr>
<tr>
<td><strong>platform</strong></td>
<td>The operating system being used/supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.</td>
<td><strong>platform</strong></td>
</tr>
<tr>
<td><strong>priority</strong></td>
<td>The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.</td>
<td><strong>priority</strong></td>
</tr>
<tr>
<td><strong>resourceRequest</strong></td>
<td>A request to the resource manager for a particular type of agent.</td>
<td><strong>resource_request</strong></td>
</tr>
<tr>
<td><strong>result</strong></td>
<td>The build result code. -1 means the build is still running, 0-254 are actual exit codes, 256 means the build timed out, and 257 means the build was stopped.</td>
<td><strong>result</strong></td>
</tr>
<tr>
<td>requestedAgents</td>
<td>The number of agents Electric Make requested.</td>
<td>N/A</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>startTime</td>
<td>The time the build was started.</td>
<td>start_time</td>
</tr>
<tr>
<td>userLabel</td>
<td>The user-supplied label (via the eMake command-line), attached to the build.</td>
<td>user_label</td>
</tr>
<tr>
<td>userName</td>
<td>The unique name of the user.</td>
<td>user_name</td>
</tr>
<tr>
<td>waitTime</td>
<td>The number of seconds Electric Make was stalled because it had to wait for agents. <strong>Note:</strong> wait_time for running builds is always 0.</td>
<td>wait_time</td>
</tr>
</tbody>
</table>

**Example**

cmtool --output simple --fields "startTime,buildName,userId,duration" getBuilds --filter "duration >10000"

Returns the start time, build name, userid, and duration of all builds that ran more than 10 seconds.

**getBuildComments**

**Description**
Retrieves a list of related build comments.

**Required Arguments**
- `<buildId>` - A unique number assigned by the Cluster Manager for each build.

**Optional Arguments**
- `--commentId <unique key that identifies a comment>`

**Result Tags**
- commentId - The unique key that identifies a comment.
- createTime - The time when the item was created.
- lastModifiedBy - The user who last modified the item.
- modifyTime - The time when the item was last modified.
- text - The text of the item.

**Example**

cmtool getBuildComments 1000 --commentId 1039

Retrieves comment 1039 for build 1000.
getBuildClass

Description
Finds a build class with full detail by its ID.

Required Arguments
- <buildClassId> - A unique number assigned by the Cluster Manager for each build class.
  Use getBuildClasses to retrieve a list of build class IDs.

Optional Arguments
None

Result Tags
See getBuildClasses for descriptions.

Example
```
cmtool getBuildClass 1
```
Retrieves build class 1.

getBuildClasses

Description
Retrieves a list of build classes with limited detail.

Required Arguments
None

Optional Arguments
- --filter <SQL query used to limit the result set>
  Note: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using --filter for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.
- --maxResults <maximum number of elements to run from a query>
Chapter 3: Build Management

- **--firstResult <starting index for the query result set>**
  Note: --firstResult takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.

- **--order <SQL order by clause>** - Used to specify ordering for the query result set.

- **--profile <details|info>** - This is the level of detail to return from a query; details gets all information and info gets a reduced information set.

### Result Tags and SQL Query Names

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotationLevels</td>
<td>Annotation choices to include in the annotation file. Possible values are basic, history, file, lookup, and waiting.</td>
<td>annotation_levels</td>
</tr>
<tr>
<td>annoUpload</td>
<td>If set to true, the annotation file is uploaded to Cluster Manager.</td>
<td>anno_upload</td>
</tr>
<tr>
<td>availableResults</td>
<td>This is a count of ‘max’ or ‘first’ results if --maxResults or --firstResult is specified.</td>
<td>N/A</td>
</tr>
<tr>
<td>buildClassId</td>
<td>A unique number assigned by the Cluster Manager for each build class.</td>
<td>id</td>
</tr>
<tr>
<td>buildClassName</td>
<td>A name assigned by the user for the build class.</td>
<td>build_class_name</td>
</tr>
<tr>
<td>defaultClass</td>
<td>If set, this is the default build class and cannot be deleted.</td>
<td>default_class</td>
</tr>
<tr>
<td>maxAgents</td>
<td>The maximum number of agents to request for this build.</td>
<td>max_agents</td>
</tr>
<tr>
<td>minAgents</td>
<td>The minimum number of agents required for this build to run.</td>
<td>min_agents</td>
</tr>
<tr>
<td>notifyOnBuildEnd</td>
<td>If set to true, the currently logged-in user will receive an email when the build is finished.</td>
<td>notify_on_build_end</td>
</tr>
<tr>
<td>platform</td>
<td>The operating system being used/supported. If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class.</td>
<td>platform</td>
</tr>
</tbody>
</table>
The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.

A request to the resource manager for a particular type of agent.

A format string that defines the resulting build name.

Example

```
cmtool getBuildClasses --filter "min_agents <5"
```

Retrieves a list of build classes that require less than 5 agents.

getBuildClassComments

**Description**

Retrieves a list of related build class comments.

**Required Arguments**

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class. Use `getBuildClasses` to retrieve a list of build class IDs.

**Optional Arguments**

- `--commentId <unique key that identifies a comment>`

**Result Tags**

- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

Example

```
cmtool getBuildClassComments 12
```

Retrieves all build class comments for build class 12.

getBuildUserStats

**Description**

Retrieves a list of user build statistics, grouped by user name, IP address, or host name.
Required Arguments

- `<groupBy>` - Possible values are `hostName`, `ipAddress`, and `userName`.

Optional Arguments

- `--filter <SQL query used to limit the result set>`
  
  **Note**: There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

- `--order <SQL order by clause>` - Used to specify ordering for the query result set.

Result Tags and SQL Query Names

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for <code>--filter</code> and <code>--order</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>duration</td>
<td>The total number of milli-seconds of all builds, filtered by the value specified in the <code>groupBy</code> argument.</td>
<td>duration</td>
</tr>
<tr>
<td>entryName</td>
<td>The value specified in the <code>groupBy</code> argument. If <code>groupBy</code> is &quot;userName&quot;, the entry name is the user name.</td>
<td>N/A</td>
</tr>
<tr>
<td>numOfBuilds</td>
<td>The number of builds.</td>
<td>N/A</td>
</tr>
<tr>
<td>waitTime</td>
<td>The number of seconds Electric Make was stalled because it had to wait for agents.</td>
<td>wait_time</td>
</tr>
<tr>
<td>workload</td>
<td>The total number of seconds used by the agents for all of the filtered builds.</td>
<td>workload</td>
</tr>
</tbody>
</table>

Example

`cmtool getBuildUserStats hostName --filter "duration >30000" --order "waitTime desc, entryName asc"`

Retrieves build user statistics for builds longer than 30 seconds, grouped by host name and ordered by wait time in a descending order and by entry name (in this case host name) in an ascending order.

**modifyBuild**

**Description**

Modifies a build.

**Required Arguments**

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.

- `<priority>` - The build priority level. Value can be `Low` or `Normal`, but not `High`.

**Optional Arguments**

None
Example
cmtool modifyBuild 1137 20

Changes build 1137 to priority 20.

modifyBuildClass

Description
Modifies a build class.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class.

Optional Arguments

- `--buildClassName <name assigned by the user for the build class>`
- `--tagDefinition <format string that defines the resulting build name>`
- `--annotationLevels <a comma-separated list of values>` - Possible values are: basic, history, file, lookup, and waiting.
- `--maxAgents <maximum number of agents to request for this build>`
- `--minAgents <minimum number of agents required for this build to run>`
- `--platform <the operating system being used/supported>` - If an OS is specified for a build class, builds from other operating systems cannot affiliate themselves with this class. The value can be Windows, Linux, or Solaris.
- `--priority <priority value>` - The default priority value is 120 (normal). 220 is high and 20 is low. Priority value can be adjusted up or down by 1-10 to "boost" the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `--annoUpload <Y|N>` - If set to true, the annotation file is uploaded to Cluster Manager. Values can also be 1, 0, true, or false.
- `--resourceRequest <value>` - This is a request to the resource manager for a particular type of agent. Value is the name of a pre-existing resource.

Example
cmtool modifyBuildClass 1 --annoupload true

Changes build class 1 to upload annotation files.

modifyBuildContent

Description
Modifies a build comment.

Required Arguments

- `<buildClassId>` - A unique number assigned by the Cluster Manager for each build class.
- `<commentId>` - The unique key that identifies a comment.
• `<text>` - The text of the item.

**Optional Arguments**
None

**Example**
cmtool modifyBuildComment 16975 1137 "This is not a usable build"

**setDatabaseConfiguration**

**Description**
Modifies database configuration settings.

**Required Arguments**

- `<databaseName>` - The database instance name.
- `<databaseType>` - The database type. Possible values are mariadb, mysql, oracle, and sqlserver.
- `<hostName>` - Machine name where the database is installed.
- `<port>` - Database port number.
- `<userName>` - Unique name of the user that is used to access the database.
- `<password>` - Secret value used to identify an account for a particular user.

**Optional Arguments**
None

**stopBuild**

**Description**
Stops a running build. (This command has no effect on completed builds.)

**Required Arguments**

*Note:* Use `getBuilds --filter "result <0"` to retrieve a list of running builds.

- `<buildId>` - A unique number assigned by the Cluster Manager for each build.

**Optional Arguments**
None

**Example**
cmtool stopBuild 16937
Chapter 4: Cluster Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

- createBuildClass
- deleteLicense
- deleteMessage
- deleteMessages
- deleteServerComment
- exportData
- getLicense
- getLicenses
- getMessage
- getMessages
- getResourceStats
- getServer
- getServerComments
- getVersion
- importData
- importLicenseData
- logMessage
- modifyServer
- modifyServerComment
- shutdownServer
- testAgents
createServerComment

Description
Creates a new server comment. Server comments are displayed on the Home page of the Cluster Manager machine.

Required Arguments
- `<text>` - The text of the item.

Optional Arguments
None

Example
cmtool createServerComment "cluster needs more servers to handle production builds"

deleteLicense

Description
Deletes a license.

Required Arguments
- `<productName>` - ElectricAccelerator, the name of the license.
- `<featureName>` - Feature name of the license, which is currently the server.

Optional Arguments
None

Example
cmtool deleteLicense ElectricAccelerator Server

Deletes the license stored in the server.

deleteMessage

Description
Deletes a specific message, including all dependent records. Messages are listed in the Cluster Manager interface Messages tab and generally are notifications about issues with agents or the Cluster Manager.

Required Arguments
- `<messageId>` - The numeric value that uniquely identifies each message.

Optional Arguments
None

Example
cmtool deleteMessage 501
**deleteMessages**

*Description*
Deletes a set of messages, including all dependent records.

*Required Arguments*
None

*Optional Arguments*
- `--filter <SQL query used to limit the result set>` - For a list of possible SQL values, see the `getMessages` command.

*Example*
cmtool deleteMessages --filter "create_time <date_sub(curdate( ), interval 200 day)"
Removes all messages more than 200 days old.

*Note:* This example is valid for MySQL only. If you use a different database, use syntax that is appropriate for your respective database.

**deleteServerComment**

*Description*
Deletes a server comment.

*Required Arguments*
- `<commentId>` - The unique key that identifies a comment.

*Optional Arguments*
None

*Example*
cmtool deleteServerComment 1396

**exportData**

*Description*
Exports Cluster Manager data to a file.

*Note:* This is a full database dump, which may take an extended period of time to complete depending on the size of the database.

*Required Arguments*
- `<fileName>` - The filename or path to export to. If you use a filename, the destination is the current working directory of the Java process, for example, `/opt/ecloud/i686_Linux` or `C:\ECloud\i686\win32`. If you use a path, the Cluster Manager Java user must have execute and write access to the destination path.
Optional Arguments
None

Example
cmtool exportData fileabc

getLicense

Description
Retrieves information for one license.

Required Arguments

- <productName> - ElectricAccelerator, the name of the license.
- <featureName> - Feature name of the license.

Optional Arguments
None

Example
cmtool getLicense ElectricAccelerator Server

getLicenses

Description
Retrieves all license data.

Required Arguments
None

Optional Arguments
None

Example
cmtool getLicenses

getMessage

Description
Retrieves a particular message.

Required Arguments

- <messageId> - The numeric value that uniquely identifies each message.

Optional Arguments
None
**Result Tags**

See `getMessages` for descriptions.

agentId
agentName
buildId
buildName
createTime
messageId
severity
text

**Example**

cmtool getMessage 47

**getMessages**

**Description**

Retrieves a list of messages

**Required Arguments**

None

**Optional Arguments**

- `--filter <SQL query used to limit the result set>` - See the possible values in the table below.
  
  **Note:** There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

- `--maxResults <maximum number of elements to run from a query>`

- `--firstResult <starting index for the query result set>`
  
  **Note:** `--firstResult` takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.

- `--order <SQL order by clause>` - Used to specify ordering for the query result set.

- `--profile <details|info>` - This is the level of detail to return from a query; details gets all information and info gets a reduced information set.

**Result Tags and SQL Query Names**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentId</td>
<td>A unique, internal number assigned to each agent by the Cluster Manager; this number can change.</td>
<td>N/A</td>
</tr>
</tbody>
</table>
### Example

```cmtool --output csv --fields buildId,severity,text getMessages --filter "text like '%I/O%'"
```

Lists all messages in the Cluster Manager that contain the string 'I/O'.

#### getResourceStats

**Description**

Retrieves resource usage statistics.

**Required Arguments**

None

**Optional Arguments**

- `--filter <SQL query used to limit the result set>` - See the possible values in the table below.

  **Note:** There is a syntax difference between MySQL and Oracle/MS SQL for enclosing criteria when using `--filter` for specific strings—for MySQL, use double quotes; for Oracle/MS SQL, use single quotes.

- `--maxResults <maximum number of elements to run from a query>`
- --firstResult \( <\text{starting index for the query result set}> \)
  
  **Note:** --firstResult takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.

- --order \( <\text{SQL order by clause}> \) - Used to specify ordering for the query result set.
- --profile \( <\text{details|info}> \) - This is the level of detail to return from a query; details gets all information and info gets a reduced information set.

### Result Tags and SQL Query Names

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>SQL Query Name for --filter and --order</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentClusterShortage</td>
<td>The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.</td>
<td>agent_cluster_shortage</td>
</tr>
<tr>
<td>agentDemand</td>
<td>The total maximum number of requests for agents by all running builds.</td>
<td>agent_demand</td>
</tr>
<tr>
<td>agentLicenseShortage</td>
<td>The difference between the maximum request for agents by all builds and the number of agents the license allows.</td>
<td>agent_license_shortage</td>
</tr>
<tr>
<td>agentsAvailable</td>
<td>The total number of active agents in the cluster.</td>
<td>agents_available</td>
</tr>
<tr>
<td>agentsInUse</td>
<td>The total number of agents assigned to builds.</td>
<td>agents_in_use</td>
</tr>
<tr>
<td>availableResults</td>
<td>This is a count of 'max' or 'first' results if --maxResults or --firstResult is specified.</td>
<td>N/A</td>
</tr>
<tr>
<td>buildsDuration</td>
<td>The average amount of time the current builds have been running.</td>
<td>builds_duration</td>
</tr>
<tr>
<td>buildsRunning</td>
<td>Average number of simultaneous builds running during a specific time period.</td>
<td>builds_running</td>
</tr>
<tr>
<td>createTime</td>
<td>The time when the item was created.</td>
<td>create_time</td>
</tr>
<tr>
<td>duration</td>
<td>The number of milliseconds the build has been running.</td>
<td>duration</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>resourceName</td>
<td>This name is used on the eMake parameter: --emake-resource, and can be specified in a build class. It is used in the ea_resource table and also matches the resource requirement string for eMake.</td>
<td>resource_name</td>
</tr>
<tr>
<td>resourceStatId</td>
<td>The resource ID number that uniquely identifies every resource.</td>
<td>id</td>
</tr>
</tbody>
</table>

**Example**

cmtool getResourceStats --maxResults 100 --order "id desc" --filter "resource_name='Cluster'"

Retrieves the 100 most current resource statistic records for the entire cluster.

**getServer**

*Description*

Retrieves server configuration.

*Required Arguments*

None

*Optional Arguments*

None

*Result Tags*

- agentAllocationPolicy - **Defined as either exclusive or shared.**
- agentLockTimerSec - **When jobs run beyond this number of seconds, the agent should be locked.**
- badAgents - **The number of enabled agents with a bad status.**
- disabledAgents - **The number of disabled agents.**
- emailInterval - **The number of minutes between email notifications.**
- emailItemLimit - **Maximum number of messages per email notification.**
- goodAgents - **The number of enabled agents with a good status.**
- logDaysToKeep - **The number of days to keep message log entries.**
- lsfAvailable - **True if LSF is available to the Cluster Manager.**
- mailFrom - **The value to use in the From header element.**
- mailPrefix - **The string used to prefix subject lines.**
- maxAgents - **The maximum number of agents to request for this build.**
- maxClockSkew - **The maximum clock skew (in seconds) allowed between the Electric Make client and agents in the cluster.**
- `minAgents` - The minimum number of agents required for this build to run.
- `preemptionPolicy` - The allocation preemption policy.
- `priority` - The build priority level. When assigning resources, an optional priority boost value can be selected to give a build class preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `resourceManagerType` - The type of resource manager that Cluster Manager should employ.
- `resourceStatInterval` - In minutes, the interval to collect stats on resource usage.
- `resourceStatKeep` - The number of minutes of resource usage statistics to keep.
- `runningBuilds` - The number of incomplete builds in the system.

**Example**

cmtool getServer

getServerComments

**Description**
Retrieves a list of related server comments.

**Required Arguments**
None

**Optional Arguments**
- `--commentId <unique key that identifies a comment>`

**Result Tags**
- `commentId` - The unique key that identifies a comment.
- `createTime` - The time when the item was created.
- `lastModifiedBy` - The user who last modified the item.
- `modifyTime` - The time when the item was last modified.
- `text` - The text of the item.

**Example**

cmtool getServerComments

Returns all comments related to the server.

getVersion

**Description**
Retrieves server version information.

**Required Arguments**
None
**Optional Arguments**

None

**Result Tags**

- label - The Electric Cloud build label for the server.
- protocolVersion - The server protocol version.
- schemaVersion - The server database schema version.
- version - The string identifying a component version.

**Example**

cmtool getVersion

**importData**

**Description**

Imports Cluster Manager data from a file.

**Note:** Because this command imports a full database dump, be advised of the following:

- The import may take an extended period of time to complete depending on the size of the database.
- You must manually delete any old/unused agents from the agents list.
- You must update the license file after import if it previously expired.

**Required Arguments**

- `<fileName>` - The name of the file to import. The file’s path is relative to the current working directory of the Java process, for example, `/opt/ecloud/i686/Linux` or `C:\ECloud\i686_win32`.

**Optional Arguments**

None

**Example**

cmtool importData fileabc

**importLicenseData**

**Description**

Imports one or more licenses.

**Required Arguments**

- `<licenseFile>` - Name of the file containing the license with the path.

**Optional Arguments**

None

**Example**

cmtool importLicenseData ./license.xml
**logMessage**

*Description*

Creates a custom message on the Cluster Manager Messages page.

**Required Arguments**

- “message text”

**Optional Arguments**

*Note:* If `--buildId` and `--agentName` are on the same line, the message is applied to the build and the agent name.

- `--severity <Debug|Info|Warning|Error>` - You can also use 0, 1, 2, or 3.
- `--buildId <buildId>` - The message applies to the specified build only.
- `--agentName <agentName>` - The message applies to the specified agent name only.

**Example**

cmtool logMessage “some text”

**modifyServer**

*Description*

Modifies the server configuration.

**Required Arguments**

None

**Optional Arguments**

- `--priority <priority value>` - The default priority value is 120 (normal). 220 is high and 20 is low. Priority value can be adjusted up or down by 1-10 to “boost” the priority to give certain build classes preference over other builds of the same priority level. Higher boost values correspond to greater preference.
- `--emailInterval <number of minutes between email notifications>`
- `--emailItemLimit <maximum number of messages per email notification>`
- `--agentAllocationPolicy <exclusive|shared>`
- `--preemptionPolicy <the allocation preemption policy>`
- `--maxClockSkew <the maximum clock skew (in seconds) allowed between the Electric Make client and agents in the cluster>`
- `--maxAgents <maximum number of agents to request for this build>`
- `--minAgents <minimum number of agents required for this build to run>`
- `--resourceManagerType <none|ea|lsf|cloud|prioritypool>` - Define which resource manager Cluster Manager should employ.
- `--mailFrom <value to use in the From header element>`
- `--mailPrefix <string used to prefix subject lines>`
- --logDaysToKeep <the number of days to keep message log entries>
- --resourceStatInterval
- --resourceStatKeep
- --wideDeepAllocationPolicy <deep|wide>  Deep means the agent allocation algorithm favors assigning more agents on the same host to a build. Wide means the algorithm favors assigning more agents from different hosts. If wide, be sure --agentAllocationPolicy is set to shared.

**Example**

```bash
cmtool modifyServer --mailFrom "cm@ourhost.com" --mailPrefix "cm message:"
```

Changes the mail "from" and mail prefix values used for mail notifications sent by the server.

**modifyServerComment**

**Description**
Modifies a server comment.

**Required Arguments**

- `<commentId>` - The unique key that identifies a comment.
- `<text>` - The text of the item.

**Optional Arguments**
None

**Example**

```bash
cmtool modifyServerComment 1178 "Server is fine"
```

**shutdownServer**

**Description**
Stops the server.

**IMPORTANT:** Use with caution.

**Required Arguments**
None

**Optional Arguments**

- --restart <true|false>

**Example**

```bash
cmtool shutdownServer
```

**testAgents**

**Description**
Instructs the Cluster Manager to contact each active agent and update its status.
**Required Arguments**

None

**Optional Arguments**

- `--agentId <unique, internal number that can change; assigned by the Cluster Manager>`
- `--agentName <name defined by the host where the agent resides [numbers and/or letters]>`
- `--filter <SQL query used to limit the result set> For a list of possible SQL values, see the `getAgents` command`

**Example**

cmtool testAgents --filter "agent_name like '%bl%'"

This command contacts all agents whose name contains 'bl' and updates their status.
Chapter 5: Reporting

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

- createFilter
- deleteFilter
- getCurrentServerLoad
- getFilter
- getFilters
- modifyFilter
createFilter

Description

Creates a named filter for a specific table.

Note: Non-global filters are stored by user ID; therefore, the same name can be used by more than one user.

Required Arguments

- <tableName> - A short string that uniquely identifies the table being filtered.
  Possible table names are: ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.
- <filterName> - A short string that uniquely identifies the filter.
- <filterQuery> - A SQL order by clause for the associated table.

Optional Arguments

- --global <true|false> - If true, this is a globally visible filter.
- --order <SQL order by clause> - Used to specify ordering for the query result set.

Example

cmtool createFilter ec_agents linuxAgents ""platform = 'linux'" --global true

Creates a global filter that selects Linux agents only.

deleteFilter

Description

Deletes a named filter for a specific table.

Required Arguments

- <tableName> - A short string that uniquely identifies the table being filtered.
  Possible table names are: ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.
- <filterName> - A short string that uniquely identifies the filter.

Optional Arguments

- --global <true|false> - If true, this is a globally visible filter.

Usage Example

cmtool deleteFilter ec_agents linuxAgents --global true

gGetCurrentServerLoad

Description

Retrieves information about the current resource load.
**Required Arguments**

None

**Optional Arguments**

None

**Result Tags**

- **agentsAvailable** - The total number of active agents in the cluster.
- **agentClusterShortage** - The difference between the maximum number of agents requested by all builds and the number of agents that were assigned.
- **agentDemand** - The total maximum number of requests for agents by all running builds.
- **agentLicenseShortage** - The difference between the maximum request for agents by all builds and the number of agents the license allows.
- **agentsInUse** - The total number of agents assigned to builds.
- **buildsDuration** - The average amount of time the current builds have been running.
- **buildsRunning** - Average number of simultaneous builds running during a specific time period.
- **createTime** - The time when the item was created.
- **duration** - The number of milli-seconds the build has been running.
- **resourceName** - This name is used on the eMake parameter: `--emake-resource`, and can be specified in a build class. It is used in the `ea_resource` table and also matches the resource requirement string for eMake.
- **resourceStatId** - The resource ID number that uniquely identifies every resource.

**Example**

cmtool getcurrentServerload

g**etFilter**

**Description**

Retrieves a named filter for a specific table.

**Required Arguments**

- `<tableName>` - A short string that uniquely identifies the table being filtered. Possible table names are: `ec_agent`, `ec_build`, `ec_build_class`, `ec_filter`, `ec_message`, `ec_resource`, `ec_resource_stat`.
- `<filterName>` - A short string that uniquely identifies the filter.

**Optional Arguments**

- `--global <true|false>` - If true, this is a globally visible filter.

**Example**

cmtool getFilter ec_agent agentFilter
getFilters

Description
Retrieves a list of saved filters for the current user.

Required Arguments
None

Optional Arguments
- --filter <SQL query used to limit the result set> - For a list of possible SQL values, see the getAgents command.
- --maxResults <maximum number of elements to run from a query>
- --firstResult <starting index for the query result set>
  Note: --firstResult takes values beginning with 0. A negative value indicates a record starting from the end of the set, counting backwards, so -1 is the last record, -2 is the next to last, and so on.
- --order <SQL order by clause> - Used to specify ordering for the query result set.

Example
cmtool getFilters --filter "table_name = 'ec_agent' && user_name is null"
Retrieves a list of all global filters for the agent table.

modifyFilter

Description
Updates a named filter for a specific table.

Required Arguments
- <tableName> - A short string that uniquely identifies the table being filtered.
  Possible table names are: ec_agent, ec_build, ec_build_class, ec_filter, ec_message, ec_resource, ec_resource_stat.
- <filterName> - A short string that uniquely identifies the filter.
- <filterQuery> - A SQL order by clause for the associated table.

Optional Arguments
- --global <true|false> - If true, this is a globally visible filter. This parameter is required for global filters.
- --order <SQL order by clause> - Used to specify ordering for the query result set.

Example
cmtool modifyFilter ec_agent agentFilter "id 750" --order agent_name
Chapter 6: User Management

Note: All database examples provided in this guide are specific to MySQL. If you use a different database, use syntax that is appropriate for your respective database.

Topics:

- addGroupMember
- changeOwnUser
- createGroup
- createUser
- deleteGroup
- deleteUser
- getAccessEntries
- getGroupMembers
- getGroups
- getEffectivePermissions
- getPermissions
- getUser
- getUsers
- getUserSettings
- login
- logout
- modifyGroup
- modifyUser
- removeGroupMember
- setBuildEndNotification
- setPermissions
- setUserSettings
**addGroupMember**

*Description*

Adds a user name to the member list for a specific group.

*Required Arguments*

- `<groupName>` - The unique name of the group.
- `<userName>` - The unique name of the user.

*Optional Arguments*

None

*Example*

cmtool addGroupMember DevGroupA ec123

Adds user 'ec123' to group DevGroupA.

**changeOwnUser**

*Description*

Modifies the settings for the currently logged-in user.

*Required Arguments*

- `<userName>` - The unique name of the user.

*Optional Arguments*

- `--fullUserName <real world name of the user>`
- `--email <the associated user email address>`
- `--password <password for a particular user>`
- `--passwordFile <path to password file>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

*Example*

cmtool ec123 --fullUserName "Mary Smith"

**createGroup**

*Description*

Creates a new local group.

*Required Arguments*

- `<groupName>` - The unique name of the group.

*Optional Arguments*

None
Chapter 6: User Management

Example
cmtool createGroup DevGroupA

createUser

Description
Creates a new local user.

Required Arguments
- <userName> - The unique name of the user.
- <password> - The password for a particular user.

Optional Arguments
- --fullUserName <real world name of the user>
- --email <the associated user email address>
- --passwordFile <path to password file> - If --password is also specified, --passwordFile overrides its value in the command line.

Example
cmtool createUser ec123 psword --fullUserName "Bob Smith" --email "ec123@ourhost.com"
Creates a new user named "ec123" whose real-world name is Bob Smith; with "psword" as his password.

Note: If you do not wish to expose passwords on the command line, you can omit the password from the example above. Press the Enter key after typing the command string (without the password) and you will be prompted for the password.

deleteGroup

Description
Deletes a local group.

Required Arguments
- <groupName> - The unique name of the group.

Optional Arguments
None

Example
cmtool deleteGroup DevGroupA
Removes the 'DevGroupA' group from the Cluster Manager.

deleteUser

Description
Deletes a local user.
**getAccessEntries**

**Description**
Retrieves permissions for all users and groups that were granted server access.

**Required Arguments**
None

**Optional Arguments**
None

**Result Tags**
- entityName - A user or group name in an access entry.
- permissions - The list of permission flags for a particular entity.

**Example**

cmtool getAccessEntries

**getGroupMembers**

**Description**
Retrieves a list of users in a specific group.

**Required Arguments**
- <groupName> - The unique name of the group.

**Optional Arguments**
None

**Result Tags**
- userName - The unique name of the user.

**Example**

cmtool getGroupMembers

Retrieves a list of user name elements.
getGroups

Description
Finds all groups known to the server. If “local” is true, returns local groups only.

Required Arguments
- `<userName>` - The unique name of the user.

Optional Arguments
- `--local <true|false>` - If true, returns local users only.

Result Tags
- `groupName` - The unique name of the group.
- `mutable` - True if the associated user or group record is modifiable.
- `providerName` - The human-readable name configured for the directory provider of a specific user or group.

Example
```
cmtool getGroups
```
Returns a list of groupInfo elements.

getEffectivePermissions

Description
Retrieves the permissions for the currently logged-in user.

Required Arguments
None

Optional Arguments
None

Result Tags
- `permissions` - The list of permission flags for a particular entity.
Possible Results

<table>
<thead>
<tr>
<th>Agent Delete</th>
<th>Maintenance Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agents Read</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>Agents Write</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>Builds Delete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>Builds Read</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>Builds Write</td>
<td>ReportsRead</td>
</tr>
<tr>
<td>Classes Delete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>Classes Read</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>Classes Write</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EM Make Impersonate</td>
<td>ResourcesWrite</td>
</tr>
<tr>
<td>EM Make Invoke</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>Maintenance Delete</td>
<td>UserModify</td>
</tr>
<tr>
<td>Maintenance Read</td>
<td></td>
</tr>
</tbody>
</table>

Example

cmtool getEffectivePermissions

Retrieves the permissions for the currently logged-in user.

getPermissions

Description

Retrieves permissions for a particular user or group.

Required Arguments

- `<principalType>` - Can be `user` or `group`.
- `<entityName>` - A user or group name in an access entry.

Optional Arguments

None

Result Tags

- `permissions` - The list of permission flags for a particular entity.

Possible Results

<table>
<thead>
<tr>
<th>Agent Delete</th>
<th>Maintenance Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agents Read</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>Agents Write</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>Builds Delete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>Builds Read</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>Builds Write</td>
<td>ReportsRead</td>
</tr>
<tr>
<td>Classes Delete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>Classes Read</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>Classes Write</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EM Make Impersonate</td>
<td>ResourcesWrite</td>
</tr>
<tr>
<td>EM Make Invoke</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>Maintenance Delete</td>
<td>UserModify</td>
</tr>
<tr>
<td>Maintenance Read</td>
<td></td>
</tr>
</tbody>
</table>
Example
cmtool getPermissions group DevGroupA
Retrieves permissions for group DevGroupA.

getUser

Description
Finds a specific user known to the server.

Required Arguments

- `<userName>` - The unique name of the user.

Optional Arguments
None

Result Tags

- `email` - The associated user email address.
- `fullUserName` - The real world name of the user.
- `groupName` - The unique name of the group.
- `mutable` - True if the associated user or group record is modifiable.
- `providerName` - The human-readable name configured for the directory provider of a specific user or group.
- `userName` - The unique name of the user.

Example
cmtool getUser ec123
Retrieves the attributes for user ec123.

getUsers

Description
Finds all users known to the server. If “local” is true, returns local users only.

Required Arguments
None

Optional Arguments

- `--pattern <pattern>` - `<pattern>` is a wildcard pattern for a user name where "*" matches any character or SQL “like” string. If LDAP is set up for getting users, the * is the preferred wildcard, as % is not understood by LDAP (this limits the result set to records in the local database).
- `--local <true|false>` - If true, returns local users only.

Result Tags
See `getUser` for descriptions.
**Example**

cmtool getUsers --pattern ec*

Retrieves information on all user IDs that begin with 'ec'.

**getUserSettings**

**Description**

Retrieves settings for the currently logged-in user.

**Required Arguments**

None

**Optional Arguments**

None

**Example**

cmtool getUserSettings

**login**

**Description**

Logs in to the client with the appropriate credentials and creates a session file in the users home directory, which allows subsequent calls to cmtool to connect to the Cluster Manager.

**Required Arguments**

- `<userName>` - The unique name of the user.
- `<password>` - The password for a particular user.

**Optional Arguments**

- `--passwordFile <path to password file>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

**Result Tags**

- `sessionId` - This is a session "cookie."

**Example**

cmtool login ec123 bobs

Logs in a user named "ec123" whose password is "bobs".
Note: If you do not wish to expose passwords on the command line, you can omit the password from the example above. Press the Enter key after typing the command string (without the password) and you will be prompted for the password.

**logout**

*Description*
Logs out of the client session.

*Required Arguments*
None

*Optional Arguments*
None

**modifyGroup**

*Description*
Modifies a local group.

*Required Arguments*
- `<groupName>` - The unique name of the group.

*Optional Arguments*
- `--newName <new group name>`

*Example*
cmtool modifyGroup DevGroupA --newName GroupDevA

**modifyUser**

*Description*
Modifies a local user.

*Required Arguments*
- `<userName>` - The unique name of the user.

*Optional Arguments*
- `--fullUserName <real world name of the user>`
- `--email <the associated user email address>`
- `--password <password for a particular user>`
- `--passwordFile <path to password file>` - If `--password` is also specified, `--passwordFile` overrides its value in the command line.

*Example*
cmtool modifyUser ec123 --fullUserName "Mary Smith"
removeGroupMember

Description
Deletes a user name from a specific group member list.

Required Arguments
- <groupName> - The unique name of the group.
- <userName> - The unique name of the user.

Optional Arguments
None

Example
cmtool removeGroupMember DevGroupA ec123

setBuildEndNotification

Description
Enables/disables notification when builds of this class end for the currently logged-in user.

Required Arguments
- <buildClassId> - A unique number assigned by the Cluster Manager for each build class. Use getBuildClasses to retrieve a list of build class IDs.
- <enabled> - Set this to true to enable notification and to false to disable it.

Optional Arguments
None

Example
cmtool setBuildEndNotification 1 true
Enables build 'end notification' for build class 1.

setPermissions

Description
Creates or modifies permissions for a user or group. The permissions are a space-separated list of permission names.

Required Arguments
- <principalType> - Can be user or group.
- <entityName> - A user or group name in an access entry.
- <permissions> - The list of permission flags for a particular entity. See the available permissions flags below.
Optional Arguments

None

Available Permission Flags

<table>
<thead>
<tr>
<th>Flag</th>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentsDelete</td>
<td>MaintenanceWrite</td>
</tr>
<tr>
<td>AgentsRead</td>
<td>MessageLogDelete</td>
</tr>
<tr>
<td>AgentsWrite</td>
<td>MessageLogRead</td>
</tr>
<tr>
<td>BuildsDelete</td>
<td>MessageLogWrite</td>
</tr>
<tr>
<td>BuildsRead</td>
<td>ReportsDelete</td>
</tr>
<tr>
<td>BuildsWrite</td>
<td>ReportsRead</td>
</tr>
<tr>
<td>ClassesDelete</td>
<td>ReportsWrite</td>
</tr>
<tr>
<td>ClassesRead</td>
<td>ResourcesDelete</td>
</tr>
<tr>
<td>ClassesWrite</td>
<td>ResourcesRead</td>
</tr>
<tr>
<td>EMakrmpersonate</td>
<td>ResourcesWrite</td>
</tr>
<tr>
<td>EMakrInvoke</td>
<td>ServerAccess</td>
</tr>
<tr>
<td>MaintenanceDelete</td>
<td>UserModify</td>
</tr>
<tr>
<td>MaintenanceRead</td>
<td></td>
</tr>
</tbody>
</table>

Example

cmtool setPermissions user ec123 "BuildsRead AgentsRead"

Restricts user ec123 to read-only privileges for builds and agents.

setUserSettings

Description

Updates settings for the currently logged-in user.

Required Arguments

- <watchMessages> - Indicates whether you want to receive notifications when messages of the specified notification level arrive. Values are Y, N, y, n, yes, no, Yes, or No.

Optional arguments

- --notificationLevel <Info|Warning|Error>

Example

cmtool setUserSettings yes --notificationLevel Info

Sets the current user to receive notifications for 'Info' level messages.