

Maven Integration

Table of contents

1 Overview.....	2
2 Plugin Details.....	2
3 Maven Repository.....	2
4 Usage.....	2
5 Invoking via the command-line.....	4
6 Installing the Plugin.....	5
6.1 Install Manually.....	6
7 Windows.....	6
8 Using with ProjectBuilder build-library command.....	6

1. Overview

This section describes how to use the maven-ctl-plugin to invoke CTL commands from a Maven 2.0.x pom.

For a use-case, see [Using Maven to Build a Library Jar](#).

2. Plugin Details

groupId	artifactId	latest version
com.controltier.ctl.maven.mojo	maven-ctl-plugin	1.0

3. Maven Repository

The maven-ctl-plugin is available from the ControlTier maven repository at this address:

<http://open.controltier.com/repo>

4. Usage

The maven-ctl-plugin is a Maven 2 plugin that can be used to execute any CTL command during a maven build phase. The plugin should be configured in your pom.xml to attach to a build phase, and to specify the appropriate parameters. The goal specified should be "command":

Example pom.xml:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>myproject</artifactId>
  <packaging>jar</packaging>
  <version>1.0</version>
  ...
  <build>
    ...
    <plugins>
      <plugin>
        <groupId>com.controltier.ctl.maven.mojo</groupId>
        <artifactId>maven-ctl-plugin</artifactId>
        <version>1.0</version>
        <executions>
```

```

        <execution>
          <phase>generate-resources</phase>
          <goals>
            <goal>command</goal>
          </goals>
          <configuration>
            <ctlBase>${CTL_BASE}</ctlBase>
            <depot>DEPOT</depot>
            <type>TYPE</type>
            <name>NAME</name>
            <command>COMMAND</command>
            <!-- Optional Arguments -->
            <!--
            <args>-arg1 value -arg2 value2</args>
            -->
          </configuration>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
</project>

```

The available parameters are listed below.

Parameter name	Description	Default/Expression	Required
ctlBase	The CTL_BASE directory path for your installation of CTL.	\${CTL_BASE} (the environment variable)	Yes
depot	Name of the depot	\${context.depot}	Yes
type	Name of the Type	\${context.type}	Yes
name	Name of the Object	\${context.name}	Yes
command	Name of the Command to invoke on the object	\${ctl.command}	Yes
args	Any additional commandline arguments to pass to the command, as a single string.	\${ctl.args}	No
loglevel	Loglevel to use: debug, verbose, info, warning, or error	warning	No
environment	Map of environment variables to pass to the		No

	executed CTL command. [See Windows Note]		
--	--	--	--

Note:

Windows Note: When using the maven-ctl-plugin in a Windows operating system, certain environment variables have to be passed through from Maven to the CTL command. These must be explicitly done in the configuration of the maven-ctl-plugin. See [Windows](#) below.

5. Invoking via the command-line

The CTL plugin doesn't have to be attached to a specific build-phase, and can be invoked using the `ctl:command` goal on the maven command-line. For example:

```
mvn ctl:command
```

This will invoke the plugin with the configuration specified in the POM that is directly below the `<plugin>` element:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>myproject</artifactId>
  <packaging>jar</packaging>
  <version>1.0</version>
  ...
  <build>
    ...
    <plugins>
      <plugin>
        <groupId>com.controltier.ctl.maven.mojo</groupId>
        <artifactId>maven-ctl-plugin</artifactId>
        <version>1.0</version>
        <configuration>
          <ctlBase>${CTL_BASE}</ctlBase>
          <depot>DEPOT</depot>
          <type>TYPE</type>
          <name>NAME</name>
          <command>COMMAND</command>
          <!-- Optional Arguments -->
          <!--
          <args>-arg1 value -arg2 value2</args>
          -->
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

```
        </plugins>
    </build>
</project>
```

Note that in this example the `<configuration>` is not wrapped within `<executions>` `<execution>` ... `</execution>` `</executions>`.

If you want to specify the configuration elements manually, either via command-line flags or via properties passed into the maven execution (for example within a Cruise Control config file), use the property name listed as `#{property}` in the table of parameters above:

```
mvn ctl:command -DCTL_BASE=/some/ctl/base -Dcontext.depot=MyDepot
-Dcontext.type=MyBuilder \
-Dcontext.name=anObject -Dctl.command=repoImport -Dctl.args="-some args"
```

6. Installing the Plugin

To install the plugin, configure your maven settings.xml file to contain a `<pluginRepository>` definition as shown:

```
<settings xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <profiles>
    <profile>
      <id>default</id>
      <activation>
        <activeByDefault>true</activeByDefault>
      </activation>
      <pluginRepositories>
        <pluginRepository>
          <id>openctier</id>
          <name>ControlTier Repository</name>
          <releases>
            <enabled>true</enabled>
            <updatePolicy>always</updatePolicy>
            <checksumPolicy>fail</checksumPolicy>
          </releases>
          <snapshots>
            <enabled>true</enabled>
            <updatePolicy>daily</updatePolicy>
            <checksumPolicy>warn</checksumPolicy>
          </snapshots>
          <url>http://open.controltier.com/repo</url>
          <layout>default</layout>
        </pluginRepository>
      </pluginRepositories>
    </profile>
```

```
</profiles>
</settings>
```

Configure pom.xml as described above. Now you can execute the Maven command to perform a build phase, and the plugin will be downloaded and installed to your local repository.

6.1. Install Manually

Alternatively, you may download the latest maven-ctl-plugin.jar from the Repository URL listed above and install it manually. Use the groupId, artifactId and version shown below. (See [Maven - Guide to installing 3rd party JARs.](#))

```
mvn install:install-file -Dfile=<path-to-file>
-DgroupId=com.controltier.ctl.maven.mojo \
  -DartifactId=maven-ctl-plugin -Dversion=1.0 -Dpackaging=jar
```

7. Windows

Windows requires that the SystemRoot and TEMP environment variables be set when executing the CTL command. In order to pass the values used at Maven build time to the CTL command, these need to be added in an <environment> configuration element. Using the values \${SystemRoot} and \${TEMP} will pass the current values from the environment.

```
<configuration>
  ...
  <environment>
    <TEMP>${TEMP}</TEMP>
    <SystemRoot>${SystemRoot}</SystemRoot>
  </environment>
</configuration>
```

8. Using with ProjectBuilder build-library command

See the [Maven Build-library use-case.](#)