Deploying Applications using maven

### IMPORTANT!

Make sure to set the local repo for your Maven installation! This prevents issues with deployment using Jenkins (CI server).

1. Navigate to Maven installation settings file (MAVEN\_HOME / conf / settings.xml)
2. Open with a text editor
3. Add the path to the default Maven repo, replacing “Owner” with your computer’s user name.

<localRepository>C:\Users\Owner\.m2\repository</localRepository>

# Tomcat Deployment

### Configure a Deployment User

1. Navigate to TOMCAT\_PATH / conf / tomcat-users.xml
2. Add the manager-script role. Unlike the manager-gui role that unlocks the Tomcat Manager App in the console, manager-script is used solely for automated scripts.

<role rolename="manager-script"/>

1. Add a user that has the manager-script role. Apache advises against giving a user both manager-gui and manager-script roles.

<user username="tomcat" password="secret" roles="manager-script"/>

### Add the Credentials to Maven Configuration

1. Navigate to MAVEN\_HOME / conf / settings.xml
2. Inside of the <servers> tag, add the credentials:

<server>

<id>tommy</id>

<username>tomcat</username>

<password>secret</password>

</server>

### Add Tomcat-Maven-Plugin to Project’s POM

1. Declare the plugin in the POM. Select the proper version for your Tomcat server.
2. Point the URL to the server’s manager/text application. This accepts scripts from users that have the manager-script role.
3. Reference the credentials by the arbitrary ID you supplied in settings.xml.
4. Specify the relative path for the context root of your application.

Example POM file:

<project ...>

...

...

<build>

<finalName>example</finalName>

<plugins>

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat6-maven-plugin</artifactId>

<version>2.2</version>

<configuration>

<url>http://localhost:8080/manager/text</url>

<server>tommy</server>

<path>/example</path>

</configuration>

</plugin>

</plugins>

</build>

</project>

### Build and Deploy Application

1. Perform your Maven build as usual.
2. The following is an abbreviated list of the tomcat7 goals you can run.
   1. mvn tomcat7:deploy
   2. mvn tomcat7:undeploy
   3. mvn tomcat7:redeploy

# Weblogic Deployment

### Configure Weblogic-Maven-Plugin

1. Move to the directory where oracle-maven-sync is found in your WebLogic install

MW\_HOME / oracle\_common / plugins / maven / com / oracle / maven / oracle-maven-sync / 12.1.3

1. Use Maven’s install:install-file goal to add the oracle-maven-sync jar to your local repository.

mvn install:install-file –DpomFile=oracle-maven-sync-12.1.3.pom –Dfile=oracle-maven-sync-12.1.3.jar

1. Run the Oracle Maven Synchronization Plugin to populate the local repository. Replace oracleHome variable with your MW\_HOME.

mvn com.oracle.maven:oracle-maven-sync:push

–Doracle-maven-sync.oracleHome=C:\WLS\wls12130

1. Ensure the installation was successful by using the Maven help:describe goal.

mvn help:describe –DgroupId=com.oracle.weblogic

–DartifactId=weblogic-maven-plugin –Dversion=12.1.3-0-0

### Add Weblogic-Maven-Plugin to Project’s POM

1. Configure build in maven project’s POM file.
2. <source> is where “mvn install” goal outputs the WAR or EAR file
3. <targets> attribute can be found in:
   1. MW\_HOME / user\_projects / domains / mydomain / servers / **myserver**

<project ...>

...

...

<build>

<finalName>example</finalName>

<plugins>

<plugin>

<groupId>com.oracle.weblogic</groupId>

<artifactId>weblogic-maven-plugin</artifactId>

<version>12.1.3-0-0</version>

<configuration>

<adminurl>t3://localhost:7001</adminurl>

<user>weblogic</user>

<password>welcome1</password>

<upload>true</upload>

<targets>myserver</targets>

<action>deploy</action>

<remote>true</remote>

<verbose>true</verbose>

<source>target/example.war</source>

<name>arbitrary</name>

</configuration>

<executions>

<execution>

<phase>deploy</phase>

<goals>

<goal>deploy</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

</project>

### Build and Deploy Application

1. Perform your Maven build as usual.
2. The following is an abbreviated list of the weblogic goals you can run.
   1. mvn weblogic:deploy
   2. mvn weblogic:undeploy
   3. mvn weblogic:redeploy

**NOTE:** Credentials need not be mapped in POM. Simply remove the <configuration> elements and use this Maven goal and parameters to deploy.

com.oracle.weblogic:weblogic-maven-plugin:deploy -Dadminurl=t3://localhost:7001

-Duser=weblogic -Dpassword=welcome1

-Dsource=project/target/project.war

-Dtargets=myserver -Dname=project