Headstrong

2008

Develeopment Environment

Maven Plugin for Flex

Arunkumar

JP Techno Park, #3/1 Millers Road, Bangalore - 560052

<http://www.israfil.net/projects/mojo/maven-flex2-plugin/usage.html>  
**Usage**

Environment Setup

Flex SDK or Flex Builder (which contains the SDK) must be installed in a known location on the machine building. Flex 2 is not available through maven, and so this plugin assumes the existance of a valid Flex 2 SDK installation.

<project>

...

<properties>

<flex.home>c:/stuff/flex2-sdk</flex.home>

</properties>

...

</project>

Additionally, your $flex.home /frameworks/flex-config.xml must be set to any appropriate local conditions. The most pressing seems to be the font caching setting. Make sure you have the following line uncommented (substituting the platform-appropriate font cache file):

<flex-config>

<compiler>

...

<fonts>

<local-fonts-snapshot>xxxFonts.ser</local-fonts-snapshot>

</fonts>

...

</compiler>

...

</flex-config>

Various ptions are found in http://www.israfil.net/projects/mojo/maven-flex2-plugin in the plugin report. Note that dataServices and useNetwork are optional. The plugin supports about 1/3 of the mxmlc options, so if it's missing somethign you need, shoot me a mail. I'm adding them all eventually.

SWC or SWF projects built with maven are propagated to repositories (local or deployment) as per any other type of resource with pom metadata.

Flex 2 project configuration

To build a flex application or component, a maven project needs to be created. This project should be of type swc or swf

<project>

...

<packaging>swf</packaging>

...

</project>

or

<project>

...

<packaging>swc</packaging>

...

</project>

The plugin must be explicitly defined in the pom.xml like so:

<project>

...

<build>

...

<plugins>

...

<plugin>

<groupId>net.israfil.mojo</groupId>

<artifactId>maven-flex2-plugin</artifactId>

<extensions>true</extensions>

<configuration>

<flexHome>${flex.home}</flexHome>

<useNetwork>true</useNetwork>

<dataServicesConfig>src/main/resources/services-config.xml</dataServicesConfig>

<main>myApplication.mxml</main>

</configuration>

</plugin>

...

</plugins>

...

</build>

...

</project>

A similar plugin setup for a swc project setup would be

<project>

...

<build>

...

<plugins>

...

<plugin>

<groupId>net.israfil.mojo</groupId>

<artifactId>maven-flex2-plugin</artifactId>

<extensions>true</extensions>

<configuration>

<flexHome>${flex.home}</flexHome>

<useNetwork>true</useNetwork>

<dataServicesConfig>src/main/resources/services-config.xml</dataServicesConfig>

<properties>.flexLibProperties</properties>

</configuration>

</plugin>

...

</plugins>

...

</build>

...

</project>

Setting up SWC dependencies in a SWC or SWF project.

Setting up a .swc dependency is almost identical to a standard .jar dependency except that the additional xml tag "type" must be included to ensure that maven interprets the repository contents correctly.

<project>

...

<dependencies>

...

<dependency>

<groupId>my.group.id</groupId>

<artifactId>my-swc-project</artifactId>

<version>1.0-SNAPSHOT</version>

<type>swc</type>

</dependency>

...

</dependencies>

...

</project>

Setting up a Flex 2 project as a dependency of a web application project

Because swf and swf dependencies are not natively understood by the maven-war-plugin, a war project will use the maven-dependency-plugin to extract desired .swf or .swc projects into a working folder, from which the war plugin can then insert them into the final packaged war.

Add any flex dependencies to the project's pom.xml (be sure to add the type tag):

<dependency>

<groupId>a.group.id</groupId>

<artifactId>an-artifact-id</artifactId>

<version>X.Y.Z</version>

<type>swf</type>

</dependency>

Then add a plugin execution of of the copy-dependencies goal to pull in any swf dependencies into a working folder, optionally stripping version numbers:

....

<plugin>

<groupId>net.israfil.mojo</groupId>

<artifactId>maven-flex2-plugin</artifactId>

<executions><execution>

<id>copy-flex</id>

<phase>process-classes</phase>

<goals><goal>copy-flex-applications</goal></goals>

<configuration>

<modules>

<swfModule>

<groupId>a.group.id</groupId>

<artifactId>an-artifact-id</artifactId>

<targetPath>${project.build.directory}/flex-resources/swf</targetPath>

<!-- optionally set the target file name -->

<targetFilename>SwfModuleFile.swf</targetFilename>

</swfModule>

</modules>

</configuration>

</execution></executions>

</plugin>

...

Finally, configure the maven-war-plugin to use the flex-resources working directory as a supplementary source of web resources.

...

<plugin>

<artifactId>maven-war-plugin</artifactId>

<configuration>

<webResources>

<resource>

<!-- this is relative to the pom.xml directory -->

<directory>${project.build.directory}/flex-resources</directory>

</resource>

</webResources>

</configuration>

</plugin>

...

</plugins>

...

</build>

This will deposit any .swf or .swc artifacts (optionally without version numbers) into the .war file.

Note: You should always use a common base path, as maven-war-plugin and maven-flex2-plugin will not always share default assumptions.