

Maven Jetty Jspc Plugin

Jetty Jspc Maven Plugin Usage Guide

To use the jetty jspc plugin, you need to put a `<plugin>` element in your pom.xml for it, and also to configure the `war` plugin so that it uses the `web.xml` file produced by the jspc plugin.

Your pom.xml will look like this:

```
<plugin>
  <groupId>org.mortbay.jetty</groupId>

  <artifactId>maven-jetty-jspc-plugin</artifactId>
  <version>6.1-SNAPSHOT</version>
  <executions>
    <execution>
      <id>jspc</id>
      <goals>
        <goal>jspc</goal>
      </goals>
      <configuration>
      </configuration>
    </execution>
  </executions>
</plugin>
<plugin>

  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-war-plugin</artifactId>
  <configuration>

  <webXml>${basedir}/target/web.xml</webXml>
  </configuration>
</plugin>
```

Note: change the <version> tag value to match your preferred jetty jspc plugin release.

i New configuration parameters

New configuration parameters will be available from jetty-6.1.10 onwards, to allow you to control the file extensions of the jsp's which are compiled. Here's an example:

```
<configuration>
  <includes>**/*.foo, **/*.bah</includes>
  <excludes>**/*.roo, **/*.bar</excludes>
</configuration>
```

The above would compile all jsp files in the webapp with extensions of .foo and .bah, but ignore those with .roo and .bar extensions. This can be useful if you have chosen something other than the usual .jsp and .jspx file extensions.

For a full listing of the <configuration> items supported by this plugin, see [jspc parameter reference](#).

Invoking the plugin will now be done by simply building your webapp to at least the `package` phase:

```
mvn package
```

However, as compiling jsp's is usually done during preparation for a production release and not usually done during development, it is more convenient to put the plugin setup inside a <profile> which can be deliberately invoked during prep for production .

For example, the following profile will only be invoked if the flag `-Dprod` is present on the run line:

```
<profiles>
  <profile>
    <id>prod</id>
    <activation>

<property><name>prod</name></property>
    </activation>
    <build>
    <plugins>
      <plugin>

<groupId>org.mortbay.jetty</groupId>

<artifactId>maven-jetty-jspc-plugin</artifactId>
      <version>6.1-SNAPSHOT</version>
      . . .
    </plugin>
    <plugin>

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

<artifactId>maven-war-plugin</artifactId>
      . . .
    </plugin>
  </plugins>
</build>
</profile>
</profiles>
```

So, the following invocation would cause your code to be compiled, the jsps to be compiled, the <servlet> and <servlet-mapping>s inserted in the web.xml and your webapp assembled into a war:

```
mvn -Dprod package
```