Maven Jetty plugin

Table of contents

Using the Maven Jetty plugin

This note is to describe the above plugin. which will run your web application in an embedded Jetty6 instance by just typing `"mvn jetty:run"` - No need to download or install Jetty manually, it's all automatic once the Maven project descriptor's set up (the pom.xml)

Configuring HowTo

Add the following to the `<build><plugins>` block of your projects `pom.xml`

```
<plugin>
    <groupId>org.mortbay.jetty</groupId>
    <artifactId>maven-jetty-plugin</artifactId>
</plugin>
```

Running

In order to run Jetty on a webapp project which is structured according to the usual Maven defaults, you don't need to configure anything.

Simply type:

```
            mvn jetty:run
```

Due to a bug in maven or the maven-jetty-plugin this only works if no jetty artifact is present in the dependencies section of the pom.xml file. This is the case for wicket-quickstart, wicket-examples and wicket-threadtest. So the tips given here can't be applied to those projects.

Running the webapp in debug mode using Java Platform Debugger Architecture (JPDA)

Using maven command line:

I couldn't find a command line reference for maven2, but the one given for maven1 still aplies for the feature used here:

First set MAVEN_OPTS environment variable with the following command:

```
    export MAVEN_OPTS='-Xdebug -Xrunjdwp:transport=dt_socket,address=4000,server=y,suspend=y'
```

After setting this property, run "maven jetty:run" and it will block, waiting for a debug connection. If "suspend=n" is set, it will start right away.

Using eclipse external tools:

Running Eclipse Open "Run --> External Tools --> External Tools... --> Program". Press "New launch configuration". On the "Main" tab, fill in the "Location:" as the full path to your "mvn" executable. For the "Working Directory:" select the workspace that matches your webapp. For "Arguments:" add jetty:run.

Move to the "Environment" tab and click the "New" button to add two new variables:

name	value
MAVEN_OPTS	-Xdebug -Xnoagent -Djava.compiler=NONE -Xrunjdwp:transport=dt_socket,address=4000,server=y,suspend=y
JAVA_HOME	Path to your java executable

As above the jvm will start right away when "suspend=n" is set.

Attaching to the server running in debug mode

Using Eclipse:

Running Eclipse Open "Run --> Debug... --> Remote Java Application". Press "New launch configuration". Fill in the dialog by selecting your webapp project for the "Project:" field, and ensure you are using the same port number as you specified in the address= property above.

Now all you need to do is Run/Debug and select the name of the debug setup you setup above.

This article is a gathering of information, credits need to be given to the authors of the pages behind the given links.

Using Netbeans (thanks to mrhaki)

This setup will also allow you to debug your application under Netbeans.

Pom setup

Add the following to the `<build><plugins>` block of your projects `pom.xml`

```
      <plugin>
        <groupId>org.mortbay.jetty</groupId>
        <artifactId>maven-jetty-plugin</artifactId>
        <configuration>
          <stopPort>9966</stopPort>
          <stopKey>jetty-stop</stopKey>
          <scanIntervalSeconds>10</scanIntervalSeconds>
        </configuration>
      </plugin>
```

The stopPort and stopKey parameters can have arbitrary values.

Configure custom Run and Debug actions

Open the properties window of your Maven project and select Actions from the Categories list.
Find the **Run** action and change the **Execute Goals** value to `jetty:stop jetty:run`.
Then, do the same for the **Debug project** action and set the following properties:

```
jpda.listen=maven
netbeans.deploy.debugmode=true
```

That's all!