# Dude! Where's my Eclipse!?

Éibhear Ó hAnluain, Version 1

*[2017-08-14 Mon]*

## Contents

**1 Dude! Where's my Eclipse 2**
1.1 Introduction . . . . . . . . . . 2
1.1.1 Examples . . . . . . . . . . 2
1.2 Pre-requisites . . . . . . . . . . 2
1.3 Setting up . . . . . . . . . . 3
1.4 Lesson 1 – compile the code. . . . . . . . . . 3
1.4.1 Starting point . . . . . . . . . . 3
1.4.2 Goal . . . . . . . . . . 3
1.4.3 Steps . . . . . . . . . . 4
1.5 Lesson 2 – build the jar file . . . . . . . . . . 4
1.5.1 Starting point . . . . . . . . . . 4
1.5.2 Goal . . . . . . . . . . 5
1.5.3 Steps . . . . . . . . . . 5
1.6 Lesson 3 – build the war file . . . . . . . . . . 5
1.6.1 Starting point . . . . . . . . . . 5
1.6.2 Goal . . . . . . . . . . 6
1.6.3 Steps . . . . . . . . . . 6
1.7 Lesson 4 – deploy and test . . . . . . . . . . 7
1.7.1 Starting point . . . . . . . . . . 7
1.7.2 Goal . . . . . . . . . . 7
1.7.3 Steps . . . . . . . . . . 7
1.8 Lesson 5 – Add new functionality . . . . . . . . . . 8
1.8.1 Starting point . . . . . . . . . . 8
1.8.2 Goal . . . . . . . . . . 8
1.8.3 Steps . . . . . . . . . . 8
1.9 Fin . . . . . . . . . . 10

# 1 Dude! Where's my Eclipse

## 1.1 Introduction

From time to time, a developer will be in a situation where code needs to be fixed, but there isn't a familiar IDE available.

Similarly, in order to progress from developer to senior developer and on to architect, it's important to understand not just what's going on in the code, but also outside the code.

This module takes a simple java tomcat application and brings you on a tour from compiling the java files, to creating a jar file and then a war file and on to deploying the application, using just a basic editor to make changes and the command-line to build the relevant artefacts.

The goal is to give a flavour of what goes on behind the IDE so that a developer can understand that taking a peek behind the curtain from time-to-time is not just informative and interesting, but is also of value to the goal of Making A Real Difference.

### 1.1.1 Examples

1. You've just discovered that there's a critical bug in the code, but you got your new development PC last night and you haven't had the chance to install your preferred IDE. You don't have the time, but you know the integration test environment has the correct JDK installed, so you can work there. Only problem: it's a command-line only environment.

2. You're working on a bug but it can only be reproduced in the test environment, where your IDE isn't installed. You need to work on it there until the issue has been identified and you know what the fix is going to be.

## 1.2 Pre-requisites

You will need a linux server with the following installed:

**tomcat** This demo was prepare with tomcat-8.

**tomcat manager** Used for deploying your built application

**A web browser** For accessing the tomcat manager

**git** To access the project

**a text editor** To fix any bugs you find (**Don't** use `eclipse`, or any other "IDE", because that's the whole point.)

**java 8 JDK** To build code

If you develop on one machine and deploy to the other, then **tomcat** and **tomcat manager** are to be on the system you're deploying to, and the rest where you're developing.

If your development environment is not on the same server as the **tomcat** instance, you'll need a copy of the file `tomcat8-servlet-api.jar` on the same system as your java code.

## 1.3 Setting up

0. All commands assume a Linux environment. `${HOME}` refers to the home directory of the user, and `${DEV_HOME}` refers to where the git repository has been cloned to.

1. Get the code

   ```
   git clone https://gitlab.com/eibhear/javaBootcampNoEclipse.git
   ```

   The repository is now in `${DEV_HOME}`. Set that as an environment variable and change into it:

   ```
   export DEV_HOME=$(pwd)/javaBootcampNoEclipse
   cd ${DEV_HOME}
   ```

2. Make sure you're at the starting point:

   ```
   git checkout lesson-1
   ```

## 1.4 Lesson 1 – compile the code.

### 1.4.1 Starting point

A clean git working area

### 1.4.2 Goal

Java files compiled into class files

#### 1.4.3 Steps

0. Confirm you can access the java compiler:

   ```
   javac -version
   ```

   If this doesn't result in a message that looks something like `javac 1.8.0_141`, you'll need to find where `javac` is installed and place it on your `${PATH}`.

1. Change to where the code is:

   ```
   cd ${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse
   ```

2. Compile the files:

   ```
   javac *.java
   ```

3. Check the errors. If you see the message

   ```
   error: package javax.servlet does not exist
   ```

   then look at your `CLASSPATH` setting, as dependent libraries are missing.

4. Fix your `CLASSPATH` and compile the java files:

   ```
   export CLASSPATH=/usr/share/java/tomcat8-servlet-api.jar:${CLASSPATH}
   javac *.java
   ```

5. If successful, you will have the following additional files under `${DEV_HOME}/source/java/`:

   ```
   org/gibiris/javaBootcampNoEclipse/MyHelloWorldBean.class
   org/gibiris/javaBootcampNoEclipse/MyHelloWorldServlet.class
   ```

   If this is the case, move on to lesson 2:

   ```
   git checkout lesson-2
   ```

## 1.5 Lesson 2 – build the jar file

### 1.5.1 Starting point

A clean git working area with two additional class files in `${DEV_HOME}/source/java/`:

```
org/gibiris/javaBootcampNoEclipse/MyHelloWorldBean.class
org/gibiris/javaBootcampNoEclipse/MyHelloWorldServlet.class
```

### 1.5.2 Goal

Class files archived into a jar file

### 1.5.3 Steps

1. Decide whether you want to include the source (`*.java`) files in the jar file.
2. Determine other files to be included in the jar file (e.g. config files, property files, etc.).
3. Change to the top-level of the package in the source:

   ```
   cd ${DEV_HOME}/source/java
   ```

   Create the `lib/` directory to take the jar file:

   ```
   mkdir -pv ../lib
   ```

4. If you're **not** including the source files in the jar:

   ```
   jar cvf ../lib/javaBootcampNoEclipse.jar \
       org/gibiris/javaBootcampNoEclipse/*.class
   ```

5. If you **are** including the source files in the jar:

   ```
   jar cvf ../lib/javaBootcampNoEclipse.jar org
   ```

6. If successful, you will now have a new file:

   ```
   ${DEV_HOME}/source/lib/javaBootcampNoEclipse.jar
   ```

   If this is the case, move on to lesson 3:

   ```
   git checkout lesson-3
   ```

## 1.6 Lesson 3 – build the war file

### 1.6.1 Starting point

A clean git working area with two additional class files in `${DEV_HOME}/source/java/`:

```
org/gibiris/javaBootcampNoEclipse/MyHelloWorldBean.class
org/gibiris/javaBootcampNoEclipse/MyHelloWorldServlet.class
```

and one additional jar file in `${DEV_HOME}/source/lib/`:

```
javaBootcampNoEclipse.jar
```

### 1.6.2 Goal

A deployable war file

### 1.6.3 Steps

0. Review the following:

   - `http://tomcat.apache.org/tomcat-8.0-doc/appdev/deployment.html` to understand the structure of the file `web.xml` and the `war` file
   - `${DEV_HOME}/source/res/web.xml` to understand how this application is to be used

1. Create an empty directory in `${DEV_HOME}` called `webapp`, then its directory structure, and then change into it:

   ```
   mkdir -vp ${DEV_HOME}/webapp
   mkdir -vp ${DEV_HOME}/webapp/WEB-INF/lib
   cd ${DEV_HOME}/webapp
   ```

2. Copy in the jsps:

   ```
   cp -rv ${DEV_HOME}/source/jsps/* ${DEV_HOME}/webapp
   ```

3. Copy in the jar file:

   ```
   cp -rv ${DEV_HOME}/source/lib/javaBootcampNoEclipse.jar \
       ${DEV_HOME}/webapp/WEB-INF/lib
   ```

4. Copy in the `web.xml` file:

   ```
   cp -rv ${DEV_HOME}/source/res/web.xml \
       ${DEV_HOME}/webapp/WEB-INF
   ```

5. Create the war file:

   ```
   jar cvf ${DEV_HOME}/source/lib/javaBootcampNoEclipse.war *
   ```

6. If this was successful you will now have a new file:

   ```
   ${DEV_HOME}/source/lib/javaBootcampNoEclipse.war
   ```

7. If this is the case, remove the direcory `${DEV_HOME}/webapp` and move on to lesson 4:

   ```
   cd ${DEV_HOME}; rm -fr ${DEV_HOME}/webapp
   git checkout lesson-4
   ```

## 1.7 Lesson 4 – deploy and test

### 1.7.1 Starting point

1. A clean working area with two additional class files in `${DEV_HOME}/source/java/`:

   ```
   org/gibiris/javaBootcampNoEclipse/MyHelloWorldBean.class
   org/gibiris/javaBootcampNoEclipse/MyHelloWorldServlet.class
   ```

   one additional jar file and one additional war file, both in `${DEV_HOME}/source/lib`:

   ```
   javaBootcampNoEclipse.jar
   javaBootcampNoEclipse.war
   ```

### 1.7.2 Goal

A working application

### 1.7.3 Steps

0. Some pre-steps

   - If this isn't the first time to deploy, be sure to *undeploy* the previous version of the application.
   - To follow these steps, you'll need your file `javaBootcampNoEclipse.war` to be accessible locally to your browser.

1. Navigate to `http://server:8080/manager` (where `server` is where your tomcat instance is running)

2. Go to the *WAR file to deploy* section and use the *Browse...* button to select the war file to deploy (`javaBootcampNoEclipse.war`).

3. Press the *Deploy* button. If successful, you'll see your application shown in the list.

4. Go to `http://server:8080/javaBootcampNoEclipse` to access your app.

5. Test the application. If all is well, clean up your development environment altogether:

   ```
   rm -vf ${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse/*.class
   rm -vf ${DEV_HOME}/source/lib/javaBootcampNoEclipse.?ar
   rm -vfr ${DEV_HOME}/webapp
   ```

6. Move on to lesson 5:

   ```
   git checkout lesson-5
   ```

## 1.8 Lesson 5 – Add new functionality

### 1.8.1 Starting point

A clean working area, brought up to date to include the two new files – `Astro/AstroFun.java` in `${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse` and `AstroLib-1.1.5ws.jar` in `${DEV_HOME}/source/lib` – and an update to `MyHelloWorldServlet.java` in `${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse`.

### 1.8.2 Goal

A working, updated application deployed to tomcat.

### 1.8.3 Steps

0. Pre-steps

   - Make sure your environment is set up as per the previous lessons:
     - `DEV_HOME` is set to where the git repository is cloned to
     - `CLASSPATH` contains `tomcat8-servlet-api.jar`
   - Clean out the working area if there are unnecessary/unwanted files:

     ```
     cd ${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse; \
         rm -vf *.class; rm -vf Astro/*.class
     rm -vf ${DEV_HOME}/source/lib/javaBootcampNoEclipse.?ar
     rm -vfr ${DEV_HOME}/webapp
     ```

   - Review the changes that have been applied to the working area

1. Compile the java files (including the new java file) as per *Lesson 1*.

   ```
   cd ${DEV_HOME}/source/java/org/gibiris/javaBootcampNoEclipse
   javac *.java
   ```

   Does it compile? If not, why not? Three hints:

   - There was a new sub-package added, `org.gibiris.javaBootcampNoEclipse.Astro`, which your java compiler needs to be able to find
   - There was a new jar file introduced to the repository
   - There's a bug in the package specification in the new java file. Looking closely at the branches of this git repository, you might even find a fix for it.

2. Build the jar file as per *Lesson 2* (except, this time, the `lib` directory exists already):

   ```
   cd ${DEV_HOME}/source/java
   jar cvf ../lib/javaBootcampNoEclipse.jar org
   ```

3. Build the war file as per *Lesson 3*.

   ```
   mkdir -vp ${DEV_HOME}/webapp
   mkdir -vp ${DEV_HOME}/webapp/WEB-INF/lib
   cd ${DEV_HOME}/webapp
   cp -rv ${DEV_HOME}/source/jsps/* ${DEV_HOME}/webapp
   cp -rv ${DEV_HOME}/source/lib/javaBootcampNoEclipse.jar \
       ${DEV_HOME}/webapp/WEB-INF/lib
   cp -rv ${DEV_HOME}/source/res/web.xml \
       ${DEV_HOME}/webapp/WEB-INF
   jar cvf ${DEV_HOME}/source/lib/javaBootcampNoEclipse.war *
   cd ${DEV_HOME}; rm -fr ${DEV_HOME}/webapp
   ```

4. Deploy the updated application as per *Lesson 4*. You will need to undeploy the previous version first.

   Does it deploy? Why not? If it deploys, does it work? Why not? (Hint: there was a new jar file introduced to the repository)

5. Fix the new bug, undeploy the bad application and deploy the corrected one.

## 1.9 Fin

**Look closely. At last, you get to see an eclipse!**

Hint: what's happening[1] in Grand Island in Nebraska at ~13:00 local time on *[2017-08-21 Mon]*?

[1]... or "what *happened*" if it has gone past