Home » Blog » Programming » Java EE » Upgrading To The Java EE 6 Web Profile

Upgrading To The Java EE 6 Web Profile

By Roger Keays, 21 March 2012

Here are my notes from our upgrade of the Sunburnt SEO software to the Java EE 6 Web Profile. Previously our stack was pretty similar to Java Web Profile, just with a lot more dependencies than is now necessary with EE 6.

We can now deploy to Glassfish 3.1 or a modified Tomcat 7.0, since we aren't using the EJB features of Java EE 6.

Code Changes

persistence.xml is now version 2.0

```
<persistence version="2.0" xmlns="http://java.sun.com/xml/ns/persistence"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
             http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
```

web.xml is now version 3.0

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee" version="3.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
         http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
```

Remove vendor-specific JPA annotations.

- //@Index
- //@ForeignKey
- //@ElementForeignKey
- //@ContainerTable
- //@PersistentMap
- //import org.apache.openjpa

Issues

[1] Missing API Code

Bytecode enhancement fails with OpenJPA and EclipseLink because of the default API jars missing code. This also affects JSF.

- java.lang.ClassFormatError Exception With EclipseLink Static Weaving [SOLVED]
- Weaving fails with ClassNotFoundException for already weaved class
- Weaver Throws ClassNotFoundException For Available Classes
- java.lang.ClassFormatError : Absent Code attribute in method that is not native or abstract in class file
- Getting JNDI to work with RDS

The workaround is to change the pom.xml dependency to

```
<!-- default Jave EE jars don't include code necessary fo
     bytecode enhancement so we use these instead -->
<dependency>
  <groupId>org.jboss.spec</groupId>
  <artifactId>jboss-javaee-6.0</artifactId>
  <version>1.0.0.Final</version>
  <scope>provided</scope>
  <type>pom</type>
</dependency>
```

[2] EL expressions throw an exception

Passing bean references by EL fails when they are dereferenced. This is a known bug.

- java.lang.ClassNotFoundException: com.sun.el.lang.VariableMapperImpl [Fixed] for Glassfish JSF
- http://java.net/jira/browse/JAVASERVERFACES-1828

Adding this to the pom.xml "fixes" the problem.

```
<!-- default Glassfish EL is broken
     http://java.net/jira/browse/JAVASERVERFACES-1828 -->
<dependency>
  <groupId>com.sun.el</groupId>
  <artifactId>el-ri</artifactId>
  <version>1.0</version>
  <scope>compile</scope>
</dependency>
```

[3] Multipart Forms Fail Silently

- how to enable multipart/form-data in Tomcat 7.0.8 server
- when form is 'multipart/form-data', form component will never process update.

commons-upload doesn't work with Servlet-3.0, but the Servlet-3.0 Multipart API is almost identical so it is easy to replace. Tomcat needs

```
<Context path="" allowCasualMultipartParsing="true">
```

since we cannot add the @MultipartConfig tag to the Faces Servlet and <multipart-config/> in web.xml doesn't seem to work. Glassfish does not seem to need this configuration.

[4] Glassfish ELCalls Incorrect Method

EL 2.2 doesn't seem to distinguish between methods like getItem(String url) and getItem(Long id) which used to work with JBoss EL. I just changed the method names to fix this.

They are the main problems we hit. Hopefully these links will help you out upgrading your app.

About Roger Keays

Roger Keays is an artist, an engineer, and a student of life. He has no fixed addressand has leftfootprints on 40-something different countries around the world. Roger is addicted to surfing. His other interests are music, psychology, languages, the proper use of semicolons, and finding good food.