JBoss 5.1.0.GA with EBJ 3.0
Quick Start Tutorial

Abstract

Probably one of the most confusing things about getting started with EBJ 3.0 is setting up a useful working environment. This document contains a quick tutorial on getting started with JBoss 5.1.0 with Eclipse IDE for developing EBJ 3.0 applications. Note that this document should be used only as a starting point for exploring EBJ 3.0.

Required Software

Please make sure you have J2SE 6.0 (update 16) installed before starting.

Download the following software:

- JBoss Application Server 5.1.0 GA, JDK 6 Edition (*jboss-5.1.0.GA-jdk6.zip*)
  Available at: http://labs.jboss.com/jbossas/downloads/

- Eclipse IDE for Java EE Developers (*eclipse-jee-galileo-SR1-win32.zip*)
  Available at: http://www.eclipse.org/downloads/

- JBoss Tools 3.1
  Available at: http://labs.jboss.com/tools/download
  (*I recommend that you install this directly using the Eclipse Update Manager. The necessary update URL is: http://download.jboss.org/jbosstools/updates/development/*)

Step 1 – Software Installation

We will not cover the installation of Eclipse since it's trivial. You just have to unpack it to a directory, install the necessary updates. The only thing extra is making sure that you install JBoss Tools 3.1. For that use the Update Manager (Help→Install New Software) and use the URL:

http://download.jboss.org/jbosstools/updates/development/

Install, at least, JBoss Tools 3.1.0.

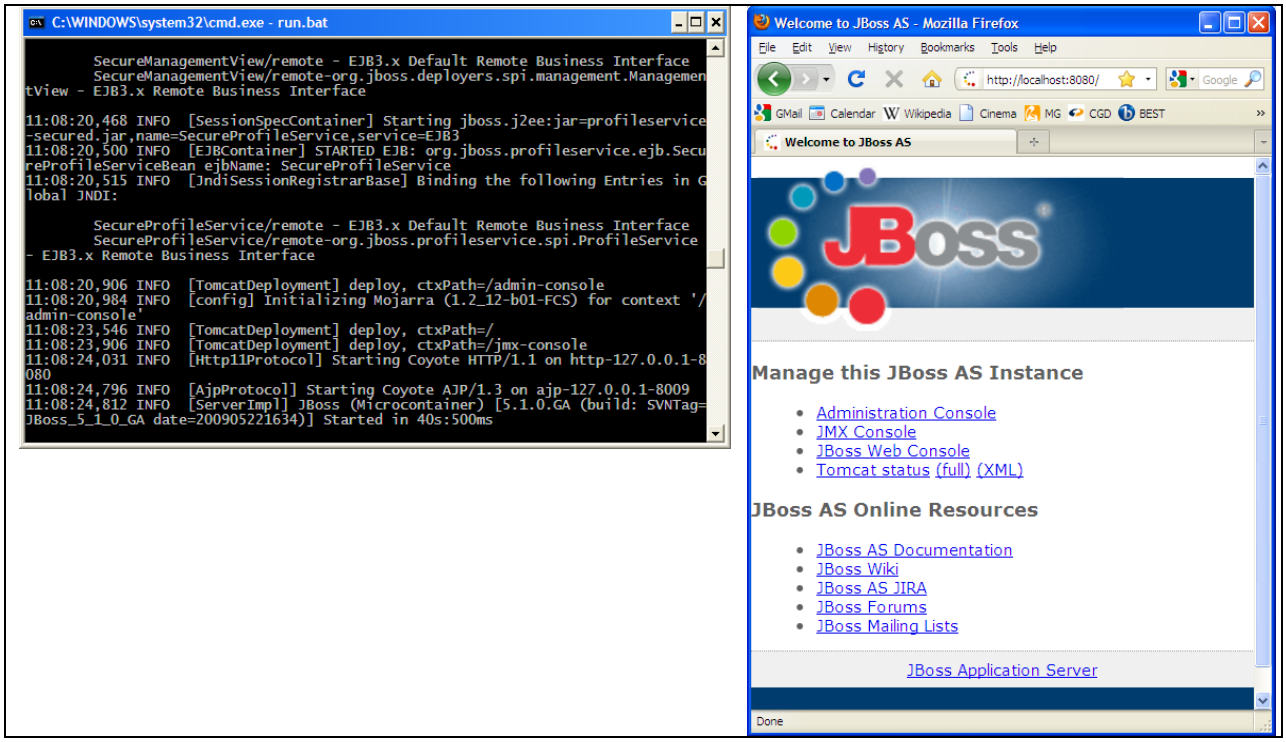
Regarding the installation of JBoss, it's also trivial. Nevertheless, let's do it step-by-step and run a sanity check after the installation.

Unzip "*jboss-5.1.0.GA-jdk6.zip*" to "*c:*\".

You will get a directory named "C:\jboss-5.1.0.GA". That contains the application server ready to run, being configured with its default values. Try out the application server by running the following commands:

```
cd c:\jboss-5.1.0.GA\bin
run.bat
```

The server should start nicely and you should be able to access its default webpage at http://localhost:8080, as the next image shows. (By sure to have the JAVA_HOME environment variable correctly set in your computer.)



If you need to shutdown the server, you only have to type the command:

```
shutdown.bat -S
```

Please do so now.

Now it's time to configure Eclipse so that it knows about the JBoss 5.1 runtime. Startup Eclipse and then:

In Eclipse, choose:

Window→Preferences→Server→Runtime Environments

Then click:

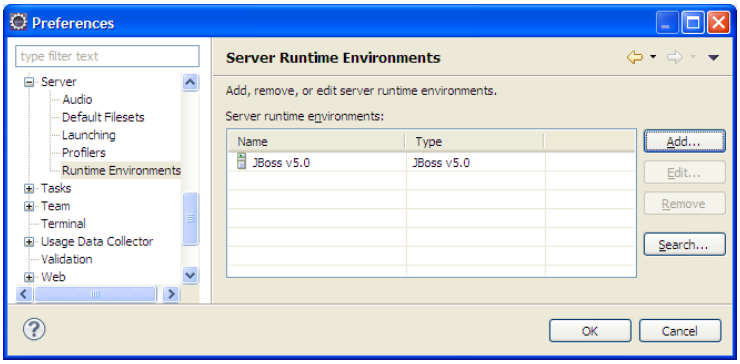
Add→JBoss→JBoss 5.0

Make sure you tick the option "Create a new local server".

At this point you will need to configure the JRE and JBoss server to use. For the JRE choose the one included with your JDK 6 (it's probably named "jdk1.6.0u16"). Regarding the "Application Server Directory", it should be "C:\jboss-5.1.0.GA". Use the default configuration.

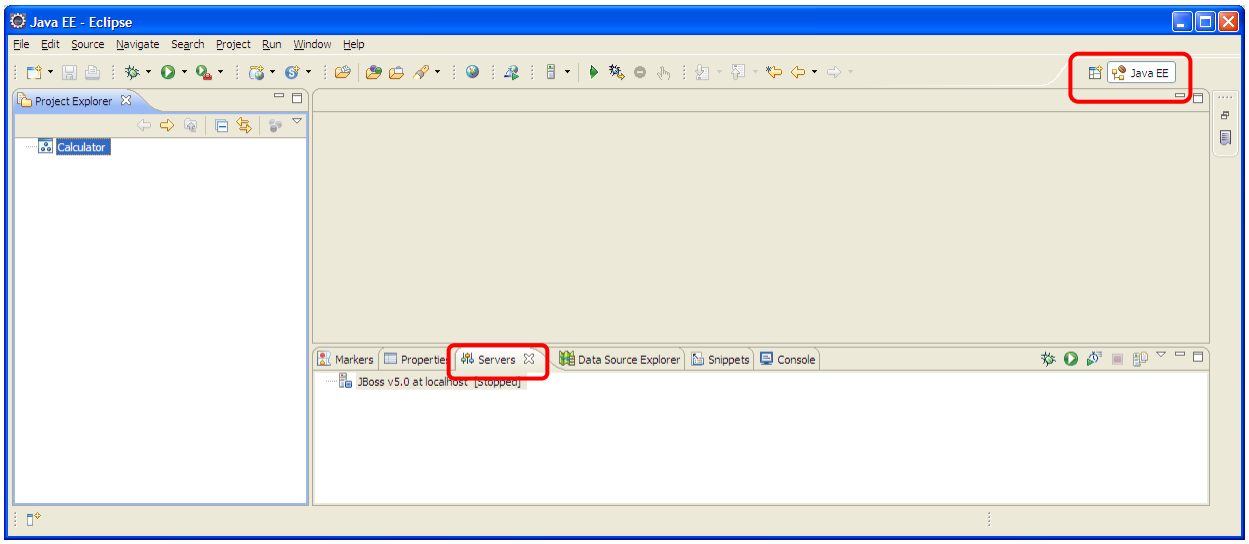
(Note that although you are using JBoss 5.1 the 5.0 works fine. I don't recommend that you use Add→JBoss Community→JBoss 5.1)

At this point, your JBoss server runtime should be correctly configured, as the next image shows:

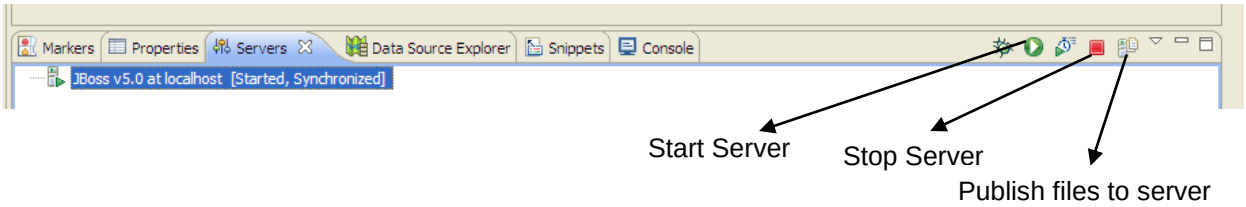


Assuming that you didn't forget to tick the option "Create a new local server" you should be able to find your runtime environment and server on the "Servers" tab. Let's check it.

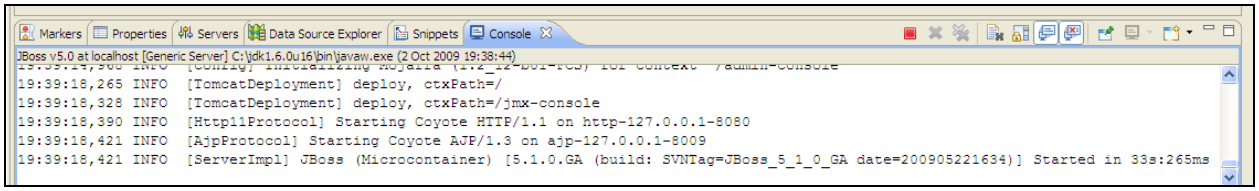
Make sure that you are using the "Java EE" view of Eclipse and that you have the "Servers" tab selected. You should be able to see your server as the next image shows.



If all went well you can now press the "Play" button getting the server "Started" and "Synchronized". ("Synchronized" means that any files and projects that you have associated to this server are the same that you are editing in Eclipse.)



You can check if the server started correctly on the "Console" tab.



One other important button is the "Publish" button (shown in the previous image). This button packages and copies the executable content of your current project to the server's deploy directory (C:\jboss-5.1.0.GA\server\default\deploy). This happens independently of the server being running or not. The reason why it's so important is because, in general, I do not recommend that you start JBoss from inside Eclipse. Using JBoss from inside Eclipse slows both JBoss and Eclipse! Also, if by any chance Eclipse crashes, JBoss is left running and you'll have to deal with that manually. My recommendation is:

- Always start JBoss using the command line and keep a command prompt window open so that you can understand what's going on.
- Use the "Publish" button for Eclipse to package and deploy your project into JBoss.

I'm not saying that you shouldn't start the server inside of JBoss – some people do work that way. I just don't find it practical. In any case, you always need to configure the JBoss server for working.

Step 2 – A Simple Stateless EJB

You will now create a simple stateless session bean that performs calculations.

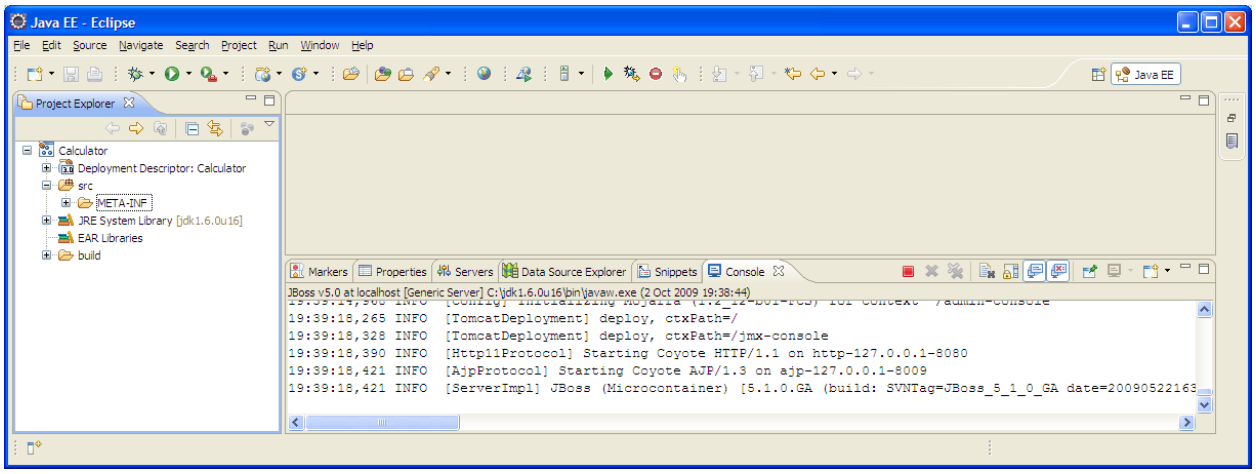
→ Make sure that you have the JBoss Application server running (command line or console)
→ Make sure that you have the JBoss Eclipse IDE open

In Eclipse, create a new "EBJ" project:
File→New→Project…→EBJ→EBJ Project

Name the project "Calculator" and accept its defaults. You should see (or select) the "Target Runtime" to be "JBoss 5.0", using the "Default Configuration for JBoss 5.0". In the "EAR Membership" option, keep it "unchecked". Make sure that "EJB module version" is 3.0.

After pressing Next you will be asked to name the source folder. The default is "ejbModule". Personally, I prefer to call it "src". Press "Finish".

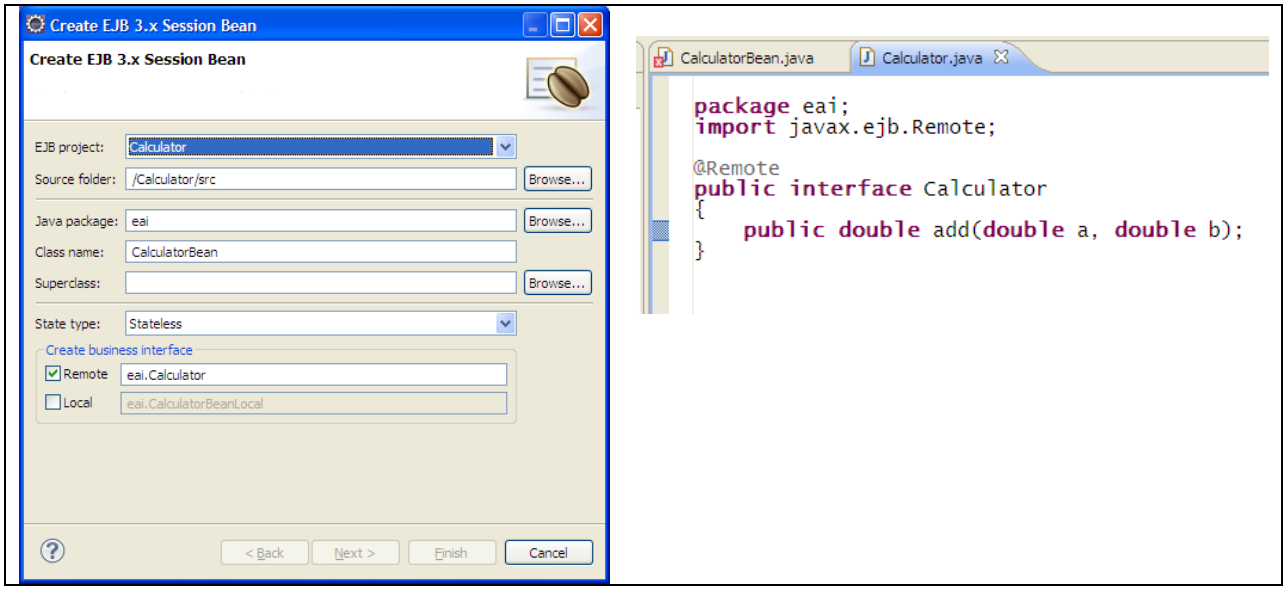
After these steps you should have an environment as shown bellow.



You are now ready to create a session bean! First you need to create the "Remote Interface" that the bean exports.

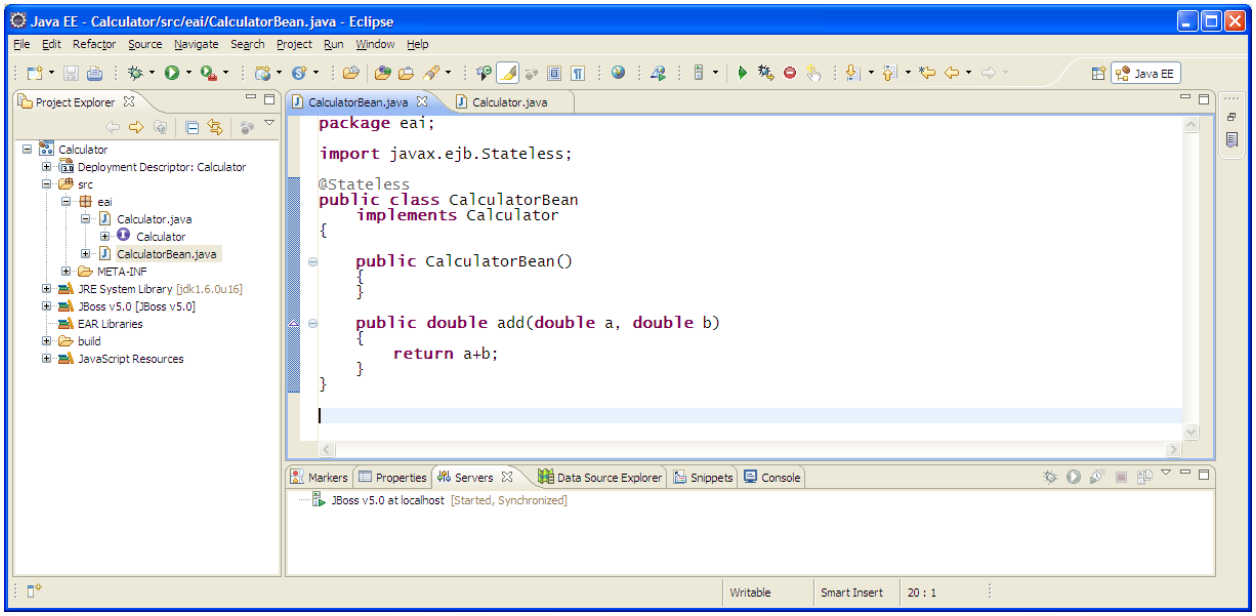
With the right mouse button click on the "src" folder and choose "New→Interface". Then, name it "Calculator", specifying that it lives under the package "eai". Press Finish and code the interface. Mark this interface as "Remote" This is shown on the next image.

Note:
You may also choose to create the EBJ by doing "New→Session Bean (EJB3.0)" and configuring the appropriate properties. It may actually be easier. ☺



So, now you have the interface that your business code exports. You just need to code the implementation on a file named "CalculatorBean.java".

Repeat the procedure described above for creating "CalculatorBean.java" containing the following code shown on the next image. Note that you now need to create a Class and not an Interface.



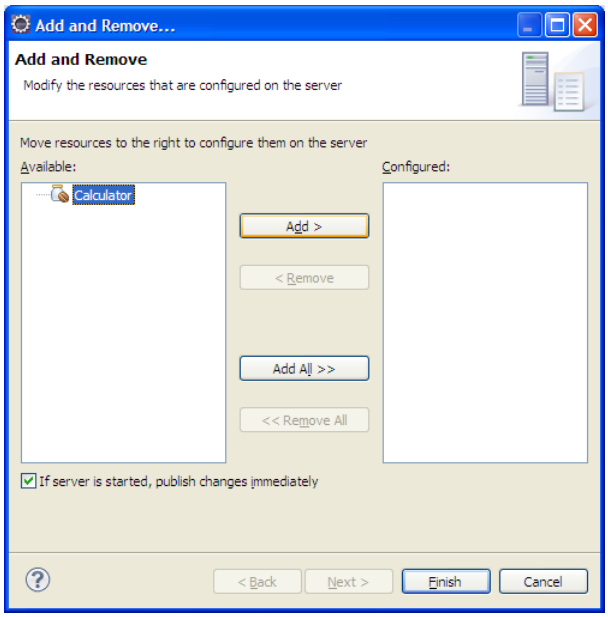
If everything was done right, Eclipse shouldn't be complaining about any error in the project. In fact, in the directory "build/classes" (on disk) you should have two new files: "eai/Calculator.class" and "eai/CalculatorBean.class".

Although at this point you already have the code that must be run on the application server, you still have to deploy it. For that, you just need to create a JAR file with the contents of the server code and copy it

into “C:\jboss-5.1.0.GA\server\default\deploy”. JBoss is constantly checking this directory looking for new server code. But wait, don’t do it yet! The good news is that Eclipse can automatically build the necessary JAR file and deploy it into JBoss without you having to do anything!

In order to benefit from this, you need to do the following:

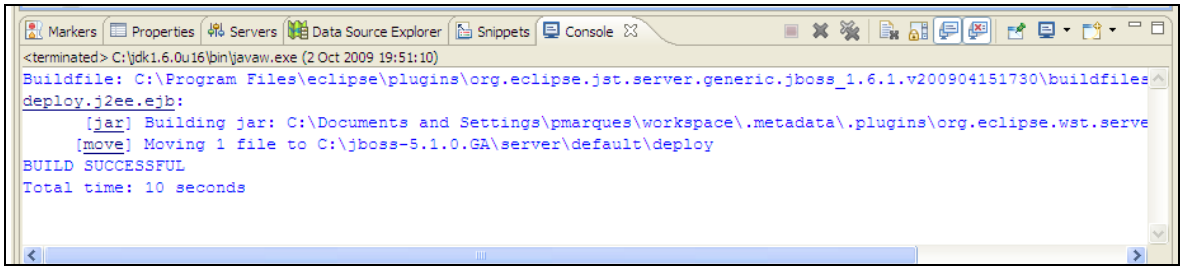
Make sure that you are using the Java EE view of Eclipse. Then, on the Servers tab, right click over JBoss 5.0 and choose “Add and Remove…”. You will get a dialog box as the one showed bellow, in which you can Add “Calculator” as a configured project.



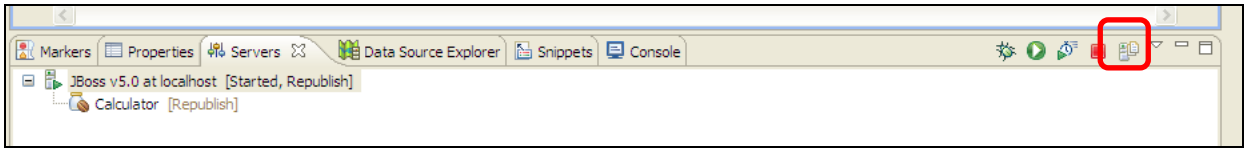
After this step you can:

Press the Publish button.

JBoss will push your code into the server, completing successfully. Also, you can check that everything went on Ok on the JBoss command line console. As I mentioned, I like to run the server using the command line. But you can also easily do it from inside Eclipse.



That’s it! You now have a session bean available at your server. Don’t forget that whenever you change your server code you will have to republish the project. But, for that, you only have to click on the Publish button:



Step 2 – A Simple Web Application

For testing this simple application you now need to create a client application. You could create a stand-alone application, which would be fine, or a web application. Since in the 2nd assignment of this course you will need to create a web tier, we are going to do just that.

If one day you need to create a stand-alone application you need to know two things:

a) The JBoss libraries that you need to include in the client are located inside "C:\jboss-5.1.0.GA\client". In particular, "**jbossall-client.jar**" is particularly useful. Note that you will need more than this file. Read the "readme.txt" file inside of "**jbossall-client.jar**".

b) In the main directory of your application you must create a file named "jndi.properties". This file enables your client to locate the JNDI JBoss directory where the references to all remote objects are registered. Its contents should be something like this:

```
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory
java.naming.factory.url.pkgs=org.jboss.naming:org.jnp.interfaces
java.naming.provider.url=localhost:1099
```

But now, let's concentrate on creating a web tier for our Calculator bean. Make sure that:

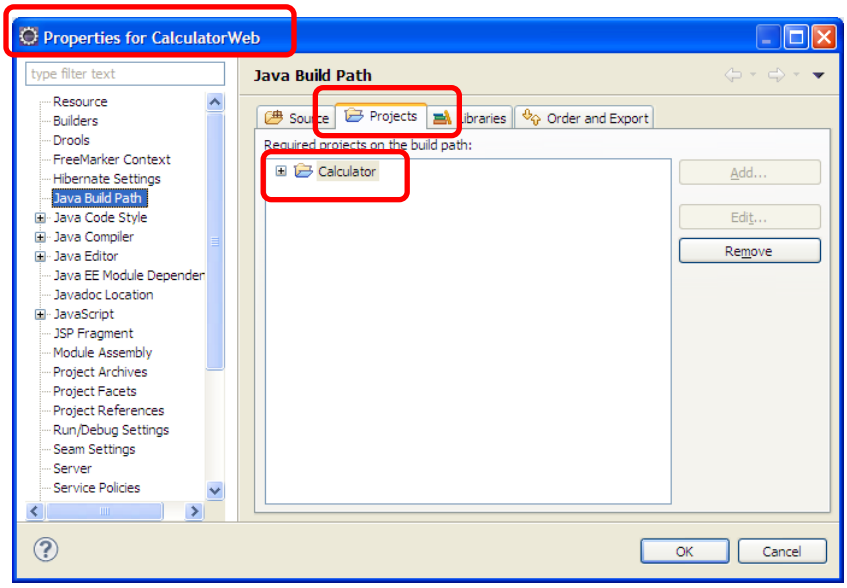
→ You have the JBoss Application server running
→ You have the Eclipse IDE open
→ The "Calculator.jar" EJB has been deployed

In Eclipse, create a new "Dynamic Web Project":
File→New→Project…→Web→Dynamic Web Project

Call the project "CalculatorWeb" and accept its defaults.
(Of course, specify "JBoss 5.0" as your application server.

Since this project will need to know about the *eai.Calculator* interface, you need to add a reference to the "Calculator" project. For doing that please perform these steps:

Right click over the "CalculatorWeb" project, choosing "Properties". Select "Java Build Path", the "Projects" tab and Add "Calculator". The result should be as shown bellow. After this step you can use the "eai.Calculator" interface in your code.



Now, you will create two new JSP pages. For that:

Select:

File→New→Other→Web→JSP

Name it "index.jsp" and add it under "CalculatorWeb/WebContent".

Repeat the same step creating a new JSP named "add.jsp".

Code the files as shown bellow.

index.jsp

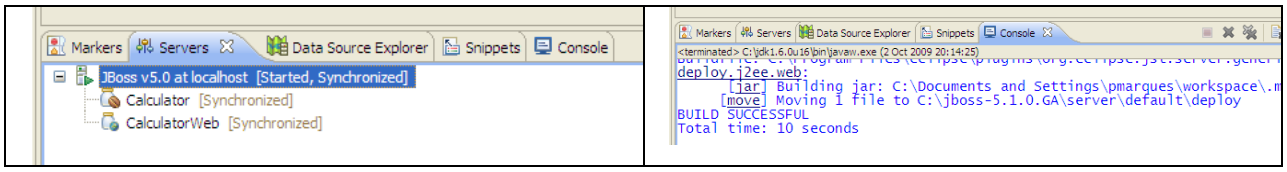
```
<html><body>

<form method="post" action="add.jsp">
    Value A: <input name="value_a" type="text" /><br/>
    Value B: <input name="value_b" type="text" /><br/>

    <input name="Submit" type="submit" value="Add" />
</form>

</body></html>
```

add.jsp

```
<%@ page import="javax.naming.*" %>
<%@ page import="eai.*" %>

<html><body>

<%
    InitialContext ctx = new javax.naming.InitialContext();
    Calculator calculator = (Calculator) ctx.lookup("CalculatorBean/remote");

    double a = Double.parseDouble(request.getParameter("value_a"));
    double b = Double.parseDouble(request.getParameter("value_b"));

    double result = calculator.add(a, b);
%>

Value A: <%= a %> <br/>
Value B: <%= b %> <br/>
Result:  <%= result %> <br/>

</body></html>
```

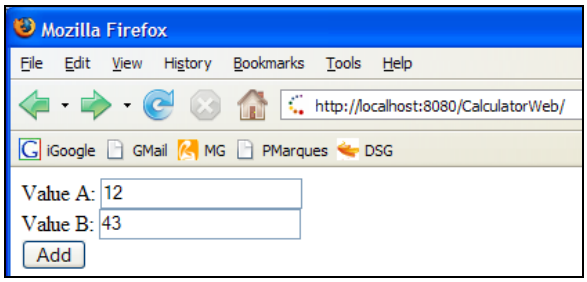
Eclipse shouldn't complain about any errors.

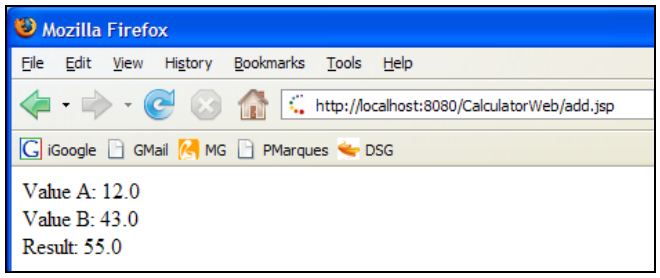
At this point you are all set! You can deploy the application at the server! After making sure that you are on the "Java EE" view, select the "Servers" tab and add your "CalculatorWeb" project to the JBoss 5.0 server. Press the "Publish to Server" button. The result should be as shown on the next image.



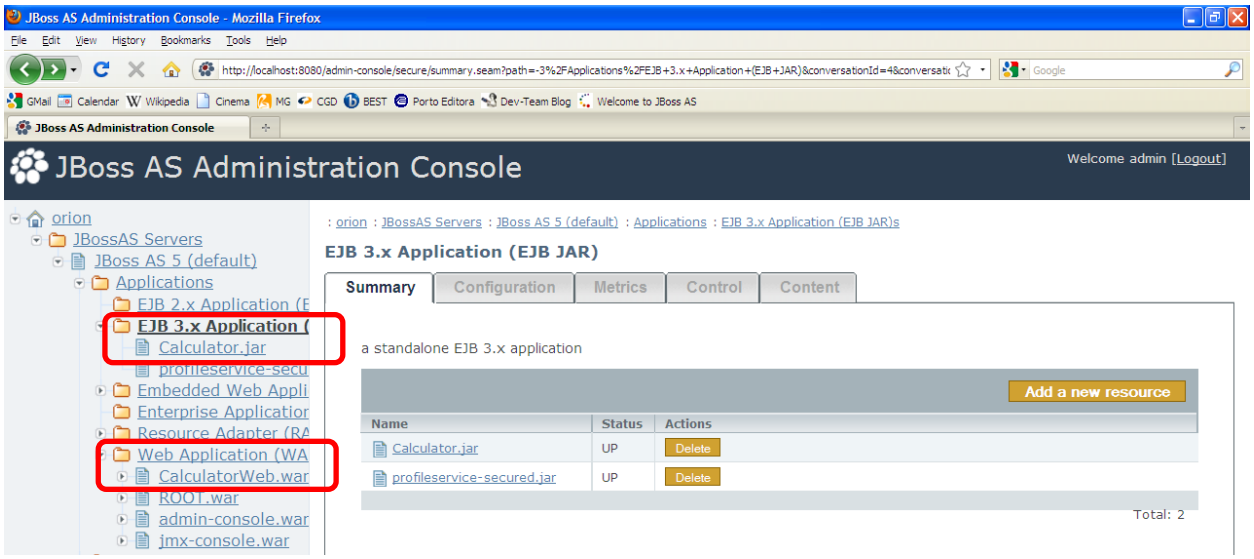
One important point: when developing web modules in J2EE, they are deployed using WAR (Web ARchive) files. If you check the deploy directory of JBoss you will find there a file named CalculatorWeb.war. (It's a simple ZIP file with a different extension.)

You can now test your application. Point your browser to http://localhost:8080/CalculatorWeb and introduce two numbers. After you press Submit you should get the sum. This is shown on the next images.





As a final step, you can you the JBoss console to see what's currently deployed. Goto to http://localhost:8080/admin-console/ and introduce admin/admin as username and password. As you can see, both applications that you've created are deployed.



Step 3 – What's Next?

This tutorial is only the tip of the iceberg! It has taught you how to start developing J2EE applications using EJB 3.0 in conjunction with JBoss 5.1 and the Eclipse IDE. Now you have to understand how to write stateful session beans, how to add persistence to the system (i.e. use a database or Hibernate), how to use and deploy web services, how to use transactions, how to add security, etc. If you can, spend some time exploring Hibernate. It's really a cool tool.

Regarding the two previous examples, it should be noted that we have deployed them independently of each other. On a more realistic scenario you will want to deploy them together using a single EAR file. (EAR stands for Enterprise ARchive and it packages several modules into a single archive.) You may have noticed that every time you created a project in Eclipse, it has an option for adding it to an EAR.