Build your first JSP with Tuscany

Build your first JSP with Tuscany

This guide will give you step by step instructions on how to use Eclipse to build a simple HelloWorld JSP which uses Apache Tuscany. In the first part, we will learn how to set up Eclipse with the Tuscany Runtime and the Eclipse Web Tools Platform. The second part shows how to create a JSP which uses Apache Tuscany.

Setting up Eclipse

Install the Eclipse Web Tools Platform (WTP)

http://download.eclipse.org/webtools/updates/

The first thing you do is to start Eclipse and go to **Help -> Software Updates -> Find and Install**,
select "Search for new features to install" and then click next. Click "New Remote Site" and enter "Eclipse WTP" in the name and "http://download.eclipse.org/webtools/updates/" in the URL, click "Finish" and follow the rest of the dialogs to install the Eclipse WTP.

Install a Tuscany distribution

This guide requires the latest Tuscany code (post 1.2 release) so you need to get a Tuscany distribution from the SNAPSHOT repository.

Create your application

Create a new JSP Project

In this step you create a JSP Project in Eclipse to hold the composite service application.
On the menu bar click on "File" - "New" - "Project", then expand "Web" and select "Dynamis Web Project" and click "Next". Enter "HelloWorld" in the "Project name" and click "Finish".

You should see your new "HelloWorld" project in the Eclipse Project Explorer, select it and right click and choose "Properties". In the properties dialog on the left panel find and click on "J2EE module dependencies" and then on the right click "Add Variable". Select TUSCANY_LIB and then click "Extend". In the Variable Extension dialog select all the jars - do that by clicking on the top jar then scroll to the bottom, hold down the shift key and click on the bottom jar. Now you need to unselect two jars - catalina-6.0.14.jar and coyote-6.0.14.jar - so scroll back up to find those, hold down the ctrl key and click on each of those jars to deselect them. Click OK, and OK.

Your HelloWorld project is now ready to use Tuscany.

Click on ***New Java Project*** button in the toolbar to launch the project creation dialog.
Next you enter "ws" as the ***Project name***, and for ***Project Layout*** select ***Create separate folders for sources and class files.***

Hit the ***Next*** button, and on the following page go to the ***Libraries*** tab. Use the ***Add Library...*** button on the right to add the **Tuscany Library** library to the project.

Hit the ***Finish*** button to complete the ***New Java Project*** dialog to create the "ws" java project.

Construct Services

First you create the "helloworld" package folders into which later in this step you place service implementations.
Select the "ws" project and click on the ***New Java Package*** button in the toolbar to launch the package creation dialog.

Next you enter "helloworld" as the package ***Name***, and press the ***Finish*** button to complete the dialog.

HelloWorld

In this step you create the HelloWorld service interface and implementation.
Select the "helloworld" package. Next you click on the dropdown arrow next to the ***New Java Class*** button and select the ***New Java Interface*** option from the dropdown list. In the dialog enter "HelloWorld" as the ***Name*** of the interface and select the Finish button to complete the dialog.
The Java editor will open on the new created Java interface. Replace the content of the editor by ***copy-paste*** of the following Java interface code snippet.

```
package helloworld;
import org.osoa.sca.annotations.Remotable;
@Remotable
public interface HelloWorld {
    String sayHello(String name);
}
```

Select the "helloworld" package again. Select the ***New Java Class*** button . In the dialog enter "HelloWorldImpl" as the ***Name*** of the class, add "Catalog" as the interface this class implements, and then select ***Finish*** to complete the dialog.

The Java editor will open on the new created Java class. Replace the content of the editor by ***copy-paste*** of the following Java class code snippet.

```
package helloworld;
public class HelloWorldImpl implements HelloWorld {
        public String sayHello(String name) {
                return "Hello " + name;
        }
}
```

After completing these steps the content of the "ws" project will look as follows.

Compose Services

Now that you have all the required service implementations you compose them together to provide the helloworld composite service. The composition is stored in a .composite file.

Select the "src" folder of the "ws" project. ***Right click*** to get the context menu, select ***New***, and then ***File***. In the ***New File*** dialog enter "helloworld.composite" for the ***File name***, and then select ***Finish*** to complete the dialog.

The Text editor will open on the new created composite file. Replace the content of the editor by ***copy-paste*** of the following composite snippet.

```
<?xml version="1.0" encoding="UTF-8"?>
<composite        xmlns="http://www.osoa.org/xmlns/sca/1.0"
                        xmlns:t="http://tuscany.apache.org/xmlns/sca/1.0"
                        xmlns:c="http://helloworld"
                        name="helloworld">


        <component name="HelloWorldComponent">
                <implementation.java class="helloworld.HelloWorldImpl"/>
        </component>
</composite>
```

After completing these steps the content of the "ws" project will look as follows.

Congratulations you completed your 1st composite service applications, now its time to take it into action.

Use Services

In this step you launch and use the ws composite service application you created.

First select the "helloworld.composite" file, in your "ws" project. ***Right click*** to get the context menu, select ***Run As***, and then ***Tuscany***. The Tuscany runtime will start up adding the helloworld composition to its domain and will make the helloworld web service live.

The Eclipse console will show the following messages.

Next Launch your Web browser and enter the following address:

[http://localhost:8080/HelloWorldJSP

You should now have your web service live, and the url should give you back a generated wsdl for the service.