Your First Framework

Introduction

Now that we have a working REST project, let's create a framework. A framework in a Project Wonder context is a special folder that contains Java libraries (.jar), other resources (EOModels, images, CSS, etc.) and a Info.plist file. If you are a Cocoa developer, a Project Wonder framework structure have a similar structure as a Cocoa framework.

When a framework is built (with Ant or Maven), the structure is like this:

MyStuff.framework
--> Resources
----> Info.plist
----> Java
------> mystuff.jar
------> some-third-party.jar
----> MyEOModel.eomodeld
----> Properties
--> WebServerResources
----> someimage.png
----> somestyling.css

So why using frameworks? Simple: to be able to use them in more than one project. Putting your EOModel into a framework is a good idea since you can built many different projects for the same business logic.

Creating the framework

So let's make a new framework. In Eclipse, open the **File** menu, select **New** and select **Wonder Framework**. Name it **BlogCommon** and click **Finish**.

Now you have another project in your Eclipse workspace. Open it and you will see that the folder structure is the same as for an Wonder application. A framework and an application in Eclipse is almost the same, the difference is the nature of the project and when the project is built, the product will be a bit different.

The next step is to copy the EOModel and its related Java code to the framework. Open the **BlogRest** project, open its **Resources** folder, and select **BlogModel.eogen** and **BlogModel.eomodel**. Right-click on one of the two files and select **Refactor** -> **Move**.

You will see the list of all opened projects in your Eclipse workspace, and we want to copy the two files into the **BlogCommon** project. Select the **Resources** folder of the **BlogCommon** project and click **OK**.

We need to do similar steps for the Java code. Open the **Sources** folder of the **BlogRest** project. Select the **your.app.model** and **your.app.model.migrations** packages, right-click on one of the packages, and select **Refactor** -> **Move**. For the destination, select the **Sources** folder, and click **OK**.

Linking the application with the framework

If you go back to the **BlogRest** project, you will see compilation errors. We need to link the application project with the framework project. To do so, right-click on the **BlogRest** project, and select **Build Path** -> **Configure Build Path**.

Click the **Libraries** tab, and you will see the list of Java archives (.jar) and Project Wonder frameworks.

To add **BlogCommon** to the build path (aka the class path), click on **Add Library**.

Select **WebObjects Frameworks** and click **Next**.

You will see a list of all frameworks that you can add to your project. Check **BlogCommon** in the list and click **Finish**. Click **OK** in the **Properties** window to go back to the main Eclipse window.

You can see that all compilation errors are now gone. The next thing we need to do is to link the H2PlugIn with the **BlogCommon** framework. Why? Because since we moved the database model to the framework, the framework needs the JDBC plugin included in the H2PlugIn in its build path. To do so, right-click on the **BlogCommon** project, and select **Build Path** -> **Configure Build Path**.

Click the **Libraries** tab, and you will see the list of Java archives (.jar) and Project Wonder frameworks. Click on **Add Library**, select **WebObjects Frameworks** and click **Next**. Select **H2PlugIn**, click on **Add Library**. Close the **Properties** window to go back to the main Eclipse window.

You can run the **BlogRest** application and everything will work. By moving the EOModel to the framework, you will be able to use the model in the next tutorial: building a DirectToWeb application to manage the blog.