

## ModelPro Installation and Introductory Guide

### Overview

This document describes how to install ModelPro and provides an introduction on its use.

### Resources

Following are resources on the web:

- Information on ModelPro: <http://modelpro.modeldriven.org/>
- Information and specifications for SoaML: <http://soaml.org>
- ModelPro download site: <http://download.modeldriven.org/modelpro>
- Cameo SOA+ suite: <http://soaplus.cameosuite.com>

Following are other resources:

- Eclipse Help (see topic “ModelPro”)

### ModelPro Versions

ModelPro comes in multiple versions. *ModelPro Gold* is an all-in-one package that contains everything needed to run ModelPro (except MagicDraw). This package includes Eclipse, ModelPro, dependent plugins, JRE, JDK, and optionally two application servers.

There are two versions of this all-in-one package, *ModelPro Gold AS* and *ModelPro Gold*. They are identical except that *ModelPro Gold AS* comes bundled with the Glassfish and JBoss application servers.

*ModelPro Gold AS* is the recommended way to use ModelPro since everything required is supplied and configured, and this approach requires the least user intervention. *ModelPro Gold* is recommended if the user already has an application server (e.g., Glassfish or JBoss) installed on their system.

Although the ModelPro plugin can be installed on any existing Eclipse application, ModelPro (or its dependent plugins) can potentially conflict with other installed plugins. As a result, the *ModelPro Gold* or *ModelPro Gold AS* packages, with a preconfigured Eclipse, are recommended.

### Environment

This document assumes Cameo SOA+ suite is already installed.

## PREPARING ECLIPSE

### ModelPro Gold and ModelPro Gold AS

#### Unzipping the File

The *ModelPro Gold* and *ModelPro Gold AS* packages both come in a zip file. This file should be unzipped to a new directory specifically created for ModelPro (e.g., C:\ModelPro). This new directory path must be short, since the lengths of the file paths in the zip file are long and can exceed the maximum length specified by Windows.

Do not unzip this file onto the Windows desktop or the user's My Documents directory, since this makes the file path lengths too long.

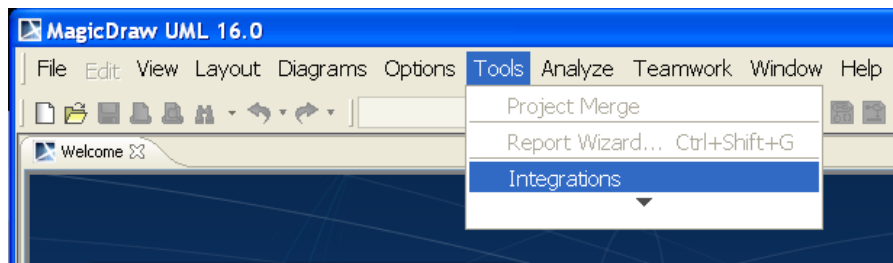
## MAGICDRAW INTEGRATION

All Eclipse installations, whether they are the *ModelPro Gold* version or a user's personal Eclipse, must be integrated with MagicDraw. User versions of Eclipse should integrate with MagicDraw before installing the ModelPro plugin.

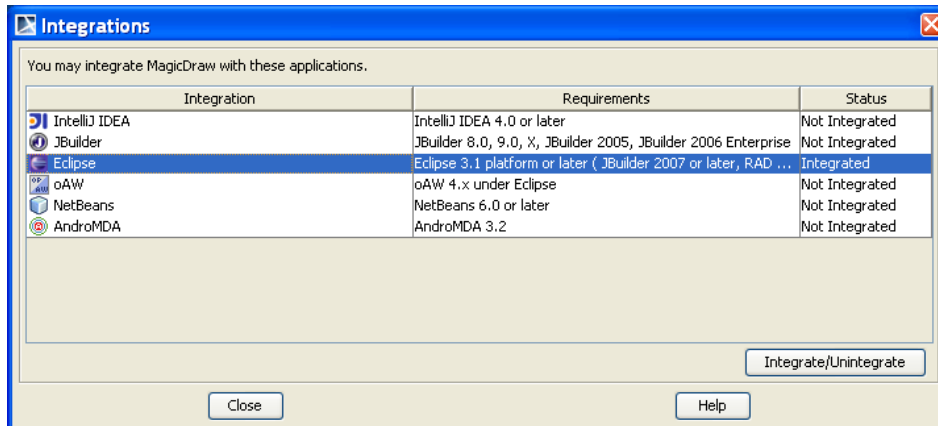
#### Integrating with MagicDraw

Launch MagicDraw.

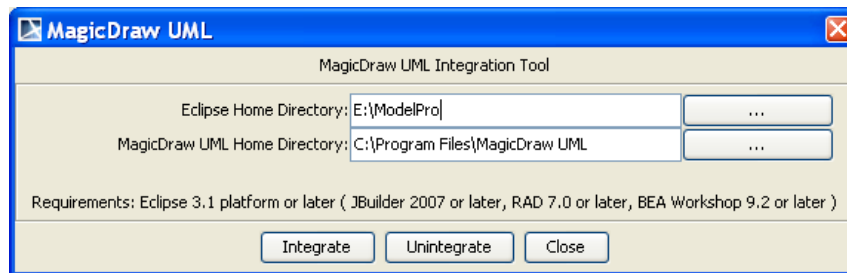
In MagicDraw, select Tools->Integrations.



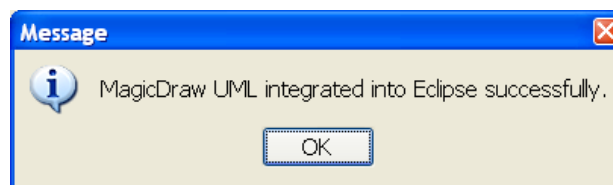
Select the Eclipse entry and press the "Integrate/Unintegrate" button.



Specify the “Eclipse Home Directory” where the Eclipse application is installed. For *ModelPro Gold* or *ModelPro Gold AS*, this is the directory where the package was unzipped.



Press the “Integrate” button. A message indicating a successful integration will be displayed.



## PROVISIONING WITH MODELPRO AND ECLIPSE

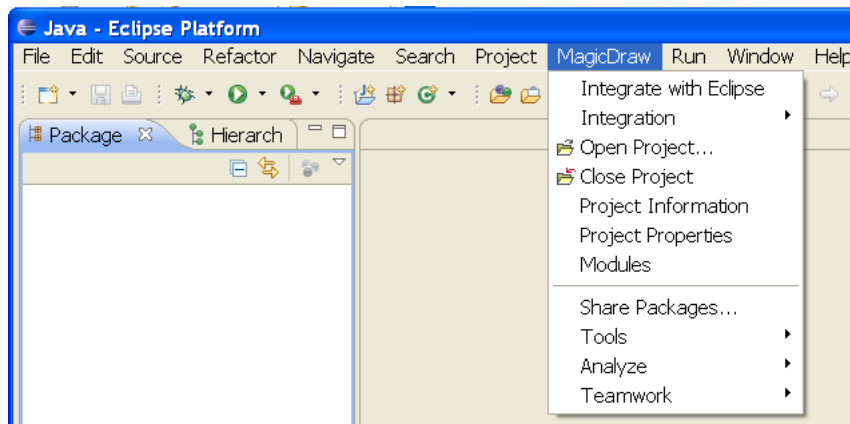
This section describes how to use Eclipse and the ModelPro plugin to provision a JEE application from a SoaML model.

### Launch Eclipse

Create a new, empty folder in Windows to use as the workspace folder.

Launch Eclipse by clicking on the <install directory>/eclipse.exe file. Specify the newly created folder for the workspace.

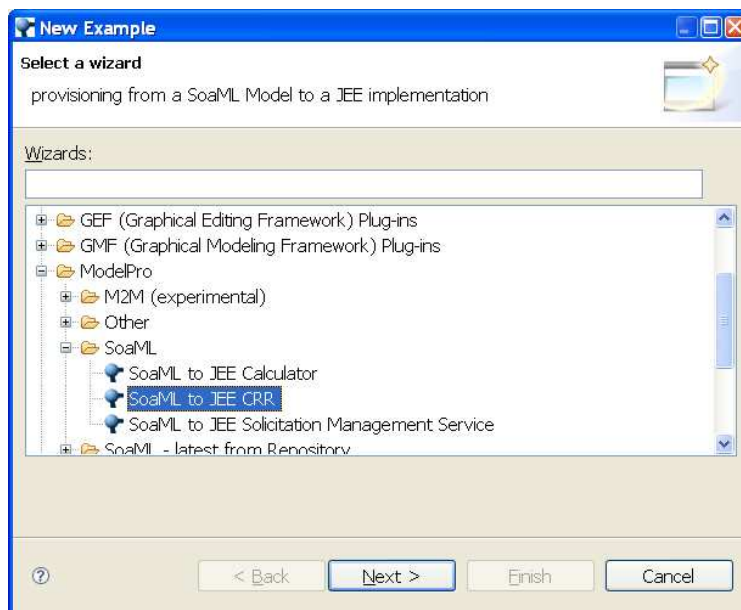
Verify that the MagicDraw drop down menu is available in Eclipse.



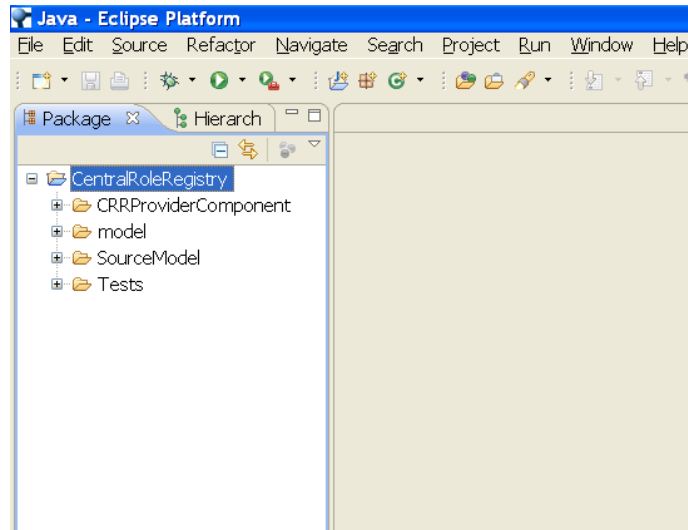
### Create an Example Project

In order to show the use of ModelPro, a built-in example project is used. The same procedures can be used for a user-defined project.

In Eclipse, select File->New->Example. When the dialog box appears, select the ModelPro->SOA-ML to JEE CRR example.



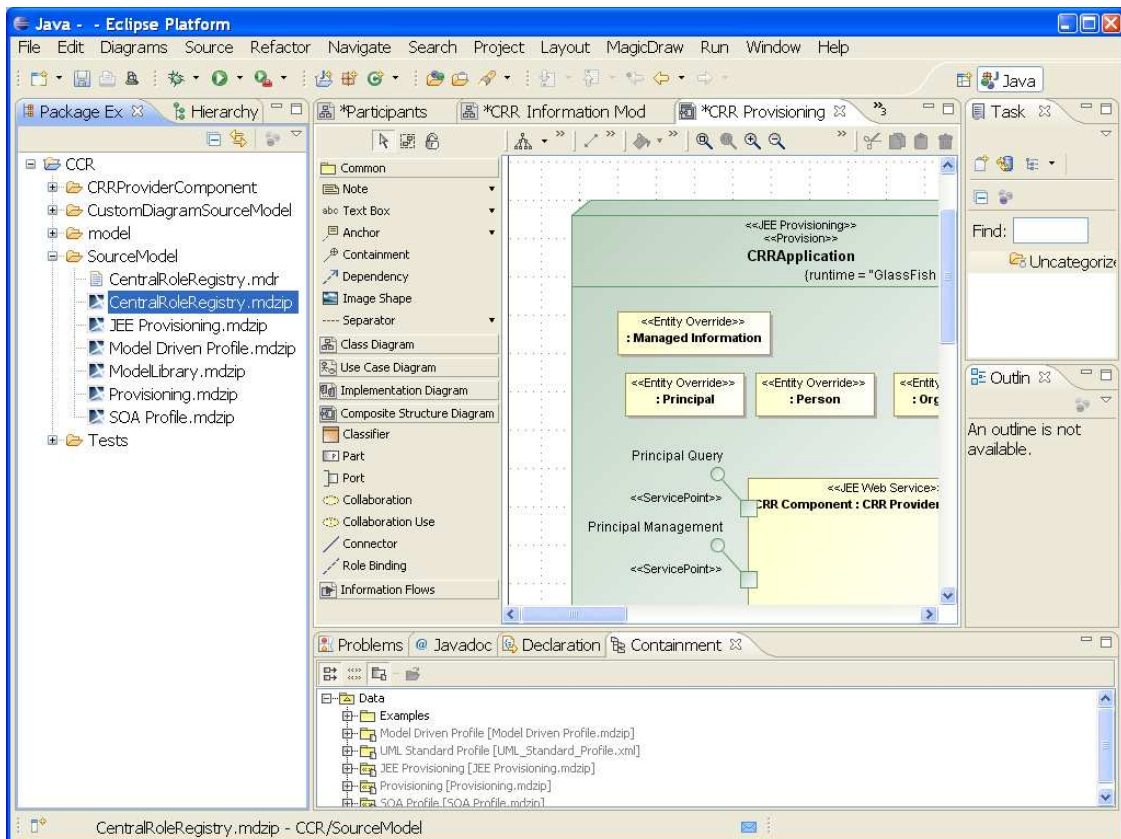
A new project in Eclipse called CentralRoleRegistry will appear.



Expand the SourceModel folder to display the MagicDraw mdzip files.

Double click on the CCR->SourceModel->CentralRoleRegistry.mdzip file to launch MagicDraw inside Eclipse.

The UML model will be displayed inside MagicDraw.

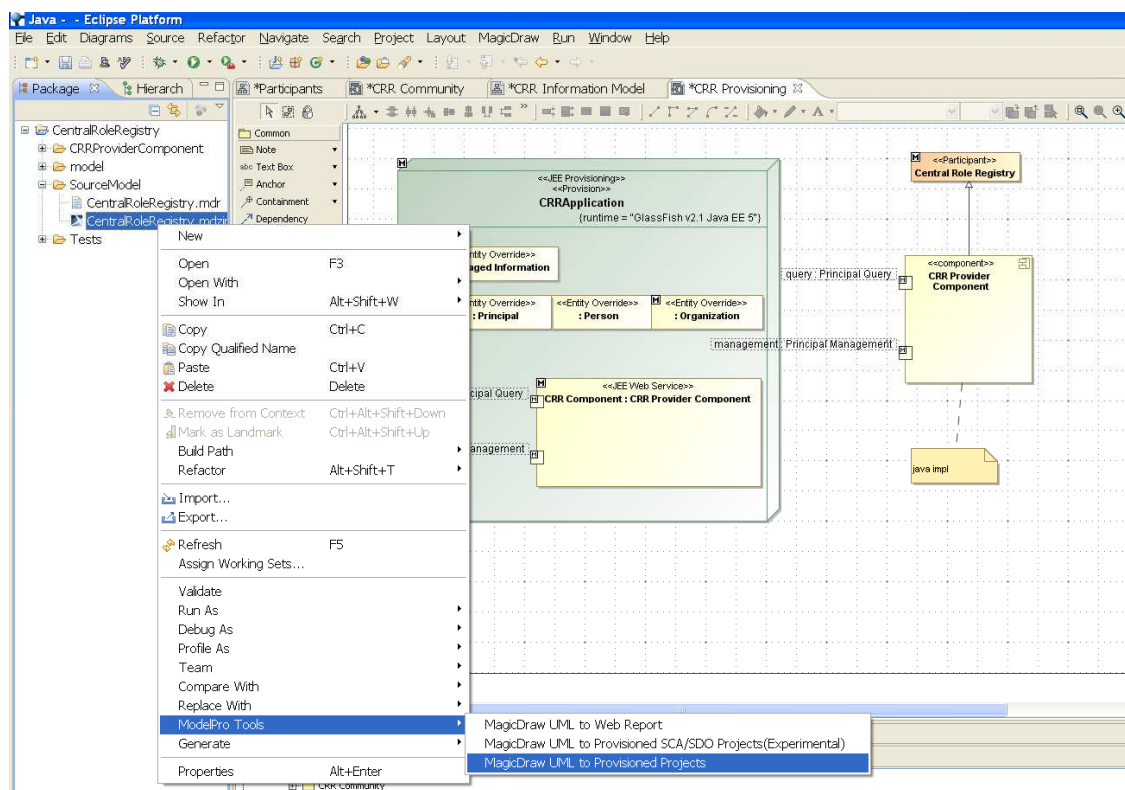


## Provision a JEE Application

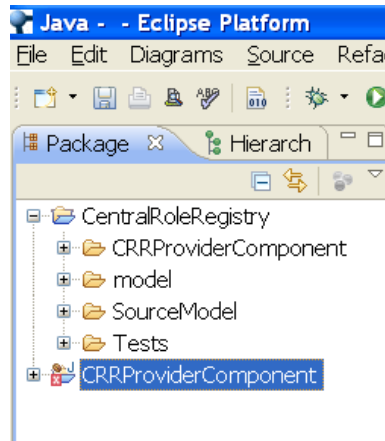
Provisioning is the generation of code from a model. ModelPro takes a SoaML model and generates a JEE application. The user can then add custom code to define the business logic.

This guide continues using the `CentralRoleRegistry` example to demonstrate how to use the provisioning capabilities. However, any user-defined SoaML model can be used as well.

Right click on the file `CentralRoleRegistry.zip` and select `ModelPro Tools->MagicDraw UML to Provisioned Projects`.



A new project named `CRRProviderComponent` will appear in the workspace containing the newly provisioned JEE application.



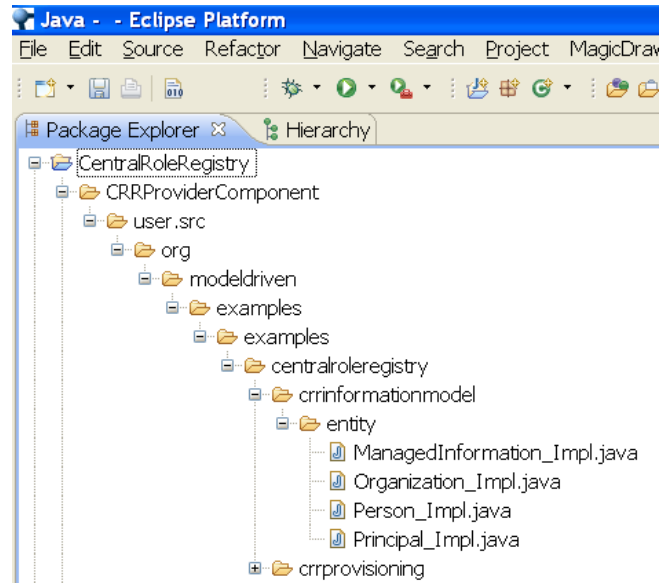
At this point, a JEE application has been provisioned from the model. Although it does not yet have custom business logic, it can be compiled, deployed, and executed with default behavior.

## ADDING BUSINESS LOGIC

The provisioned Eclipse project contains three source code directories. One of these directories, called `user.src`, contains stub classes which can be filled in with business logic. The rest of the code is automatically generated from the model and should not be modified.

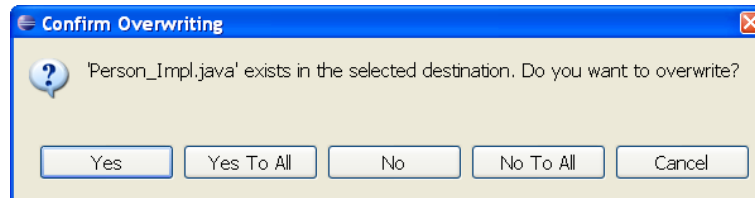
The example `CentralRoleRegistry` project contains sample business logic code that simulates a developer adding custom logic.

Expand the `CentralRoleRegistry` project and the `CRRProviderComponent` folder inside the `CentralRoleRegistry` project to view the sample business logic Java files.

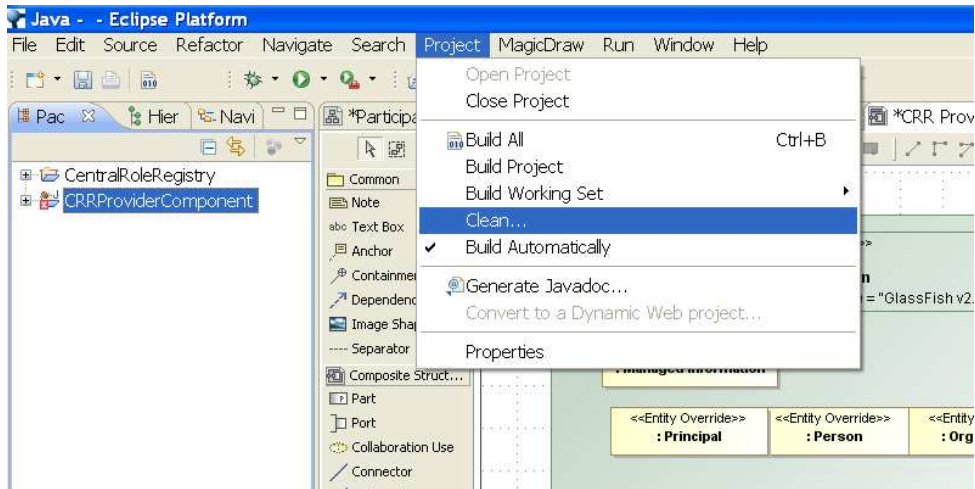


Copy the files from the user.src folder in CentralRoleRegistry /CRRProviderComponent and paste them into the same directories in the CRRProviderComponent project. This simulates a business developer modifying the provisioned files in user.src.

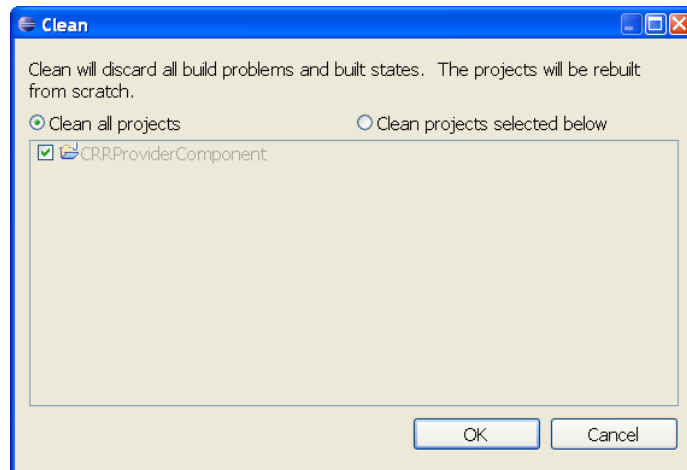
When asked if the files should be overwritten, press the “Yes to All” button.



Click on the CRRProviderComponent project and select Project->Clean to rebuild the Eclipse project.



Press OK in the Clean dialog box.



Press OK in the Clean dialog box.

This rebuilds the JEE application with the custom business logic.

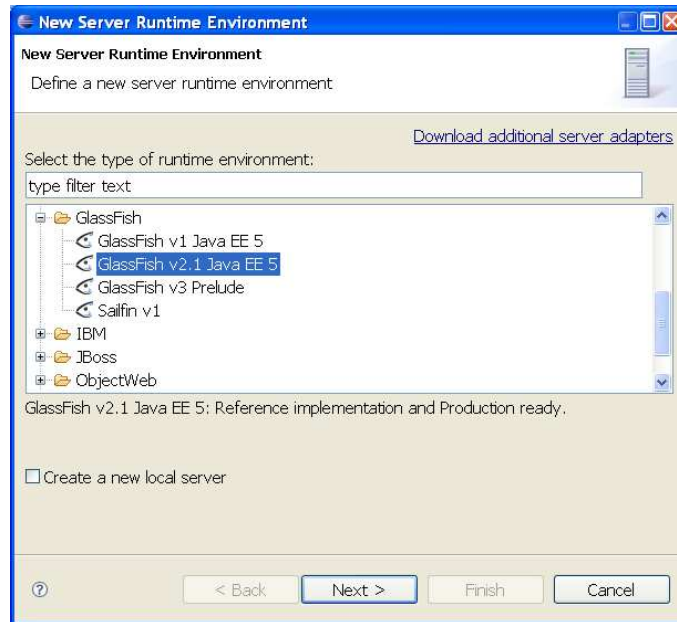
## CONFIGURING THE APPLICATION SERVER

For those users with a local installation of an application server, and who are not using the *ModelPro Gold AS* package, this section defines how to configure an application server for use with Eclipse and the provisioned JEE application.

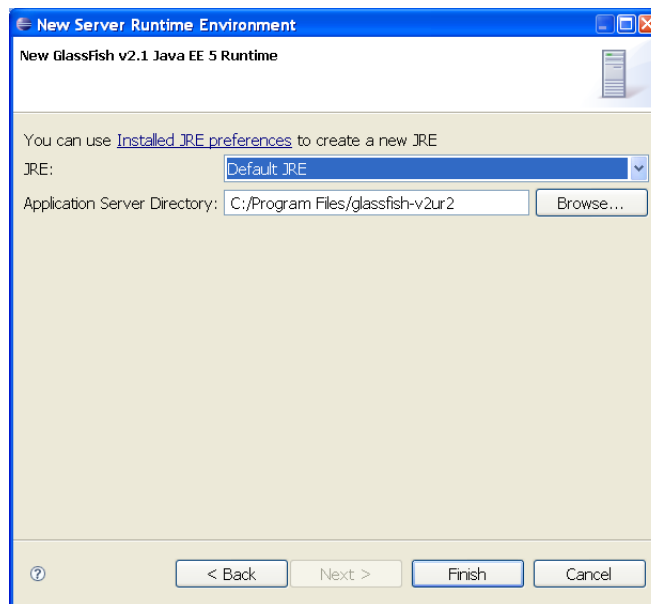
The following instructions are specific to a Glassfish v2.1 installation, but can be followed for any application server.

In Eclipse, select Window->Preferences. When the dialog box appears, select Server->Runtime Environments. Press the "Add" button.

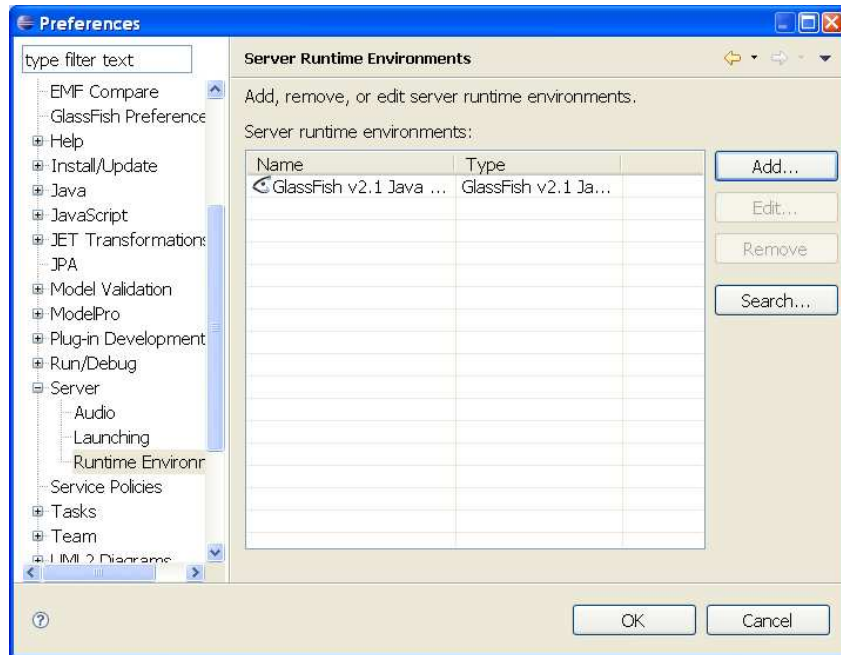
Select the Glassfish adapter matching the local Glassfish installation.



Browse to the local Glassfish installation location.

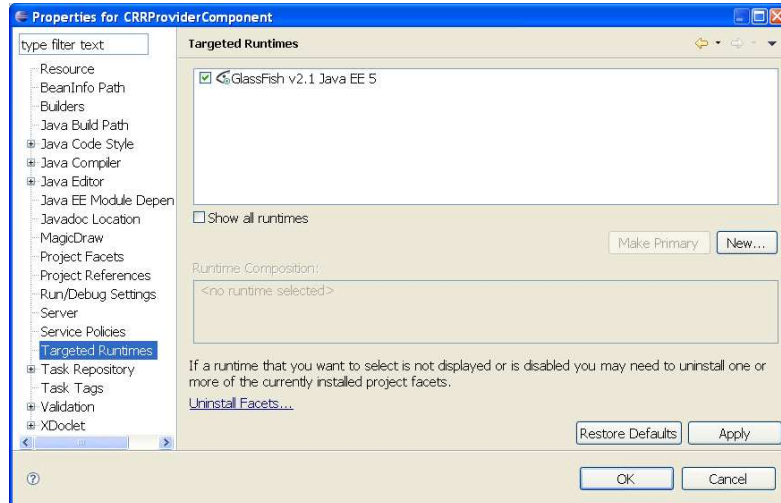


The new Glassfish JEE runtime should appear in the list.



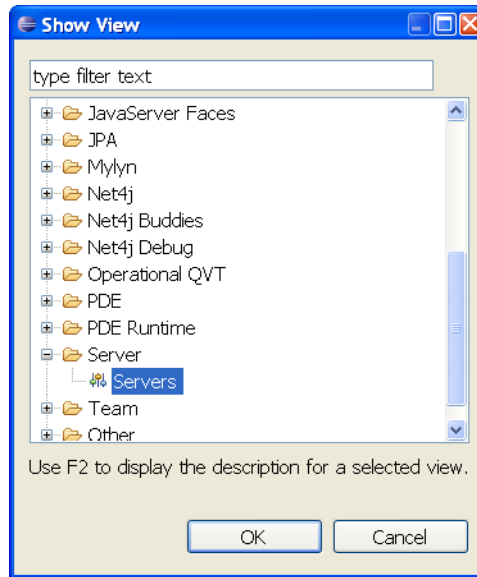
The Eclipse project created for the provisioned JEE application must be associated with the Glassfish runtime.

In the Package Explorer view, right click on the project and select “Properties”. Then select “Targeted Runtimes”. Ensure the desired configured runtime is selected.

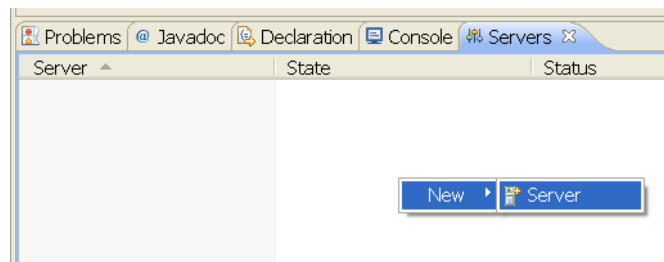


The application servers are displayed in the server view in Eclipse, which is one of the tabs in the lower right hand corner.

In Eclipse, select Window->Show View->Other. Then select Server->Servers.



Press “OK”, and the Server tab will be displayed. It may be empty.  
Right click in this empty tab area and select New->Server.



Follow the wizard to add the Glassfish server.

**New Server**

**Define a New Server**

Choose the type of server to create

Server's host name: localhost

[Download additional server adapters](#)

Select the server type:

type filter text

- GlassFish
  - GlassFish v1 Java EE 5
  - GlassFish v2.1 Java EE 5
  - GlassFish v3 Prelude
  - Sailfin v1

GlassFish v2.1 Java EE 5: Reference implementation and Production ready.

Server name: GlassFish v2.1 Java EE 5 at localhost

Server runtime environment: GlassFish v2.1 Java EE 5 [Add...](#)

[Configure runtime environments...](#)

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

**New Server**

**New GlassFish v2.1 Java EE 5 Server**

Create a new GlassFish v2.1 Java EE 5 server

Domain Name: domain1

Domain Directory: C:/Program Files/glassfish-v2ur2/domains [Browse...](#)

Administrator Id: admin

Administrator Password: adminadmin

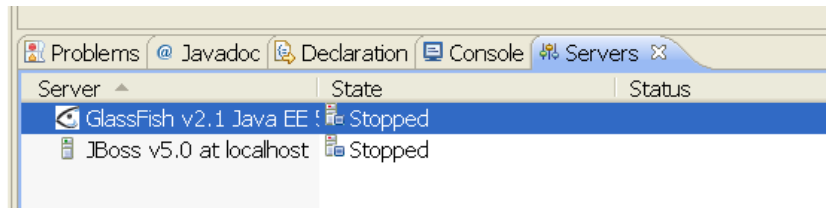
[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

## DEPLOY THE WEB SERVICE

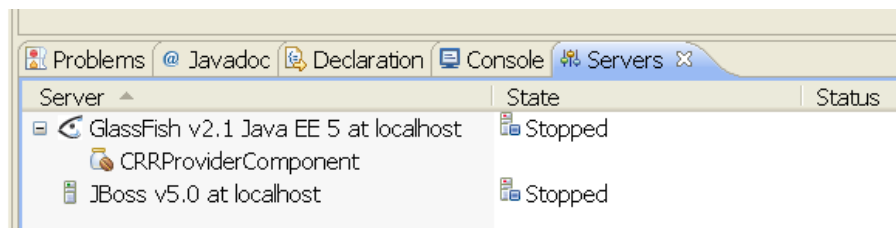
Now that the provisioned JEE application is built and the application server configured in Eclipse, the web service can be deployed to an application server.

This section uses Glassfish as an example application server.

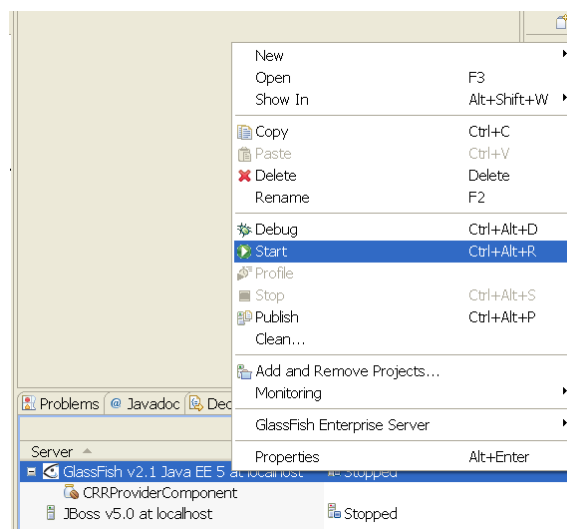
Open the Servers view, which may contain multiple servers.



Drag and drop the CRRProviderComponent project from the Package Explorer view onto the Glassfish server. The project will appear under the server.



Right click on the GlassFish v2.1 server and select Start.



The Console view will show that the application is deployed successfully.

```

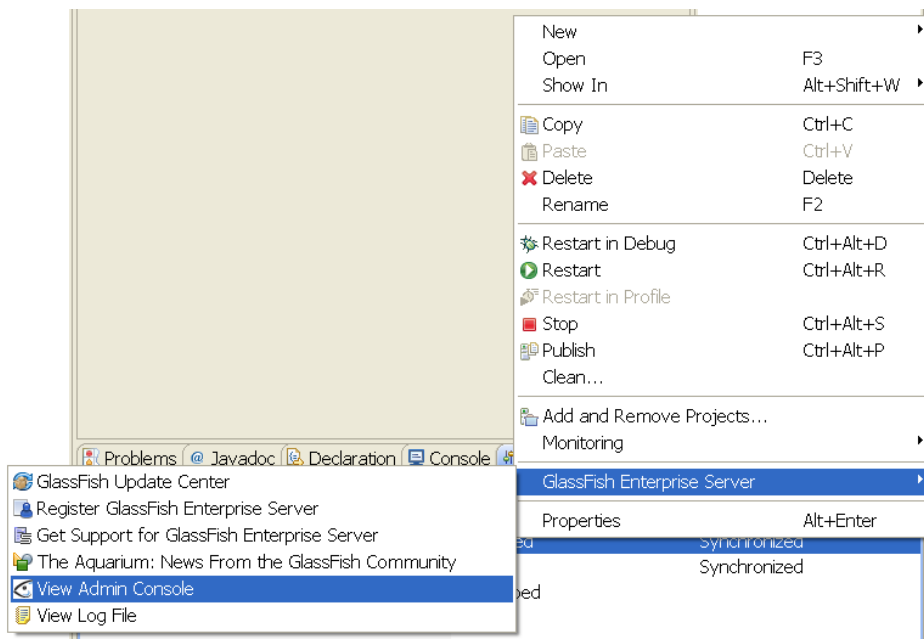
<terminated> E:\ModelProGold\eclipse\jre\bin\javaw.exe (Mar
Buildfile: E:\ModelProGold\workspace\.metadata\.plugins\org.eclipse.jst.server.gener
deploy.j2ee.ejb:
[jar] Building jar: E:\ModelProGold\workspace\.metadata\.plugins\org.eclipse.j:
tools:
deploy:
[exec] Command deploy executed successfully.
deploy-url-message:
[echo] Application Deployed at: http://localhost:8080/CRRProviderComponent
BUILD SUCCESSFUL
Total time: 32 seconds

```

## ADMINISTER THE WEB SERVICE

This section describes how to use the Glassfish admin console to view and administer the web service.

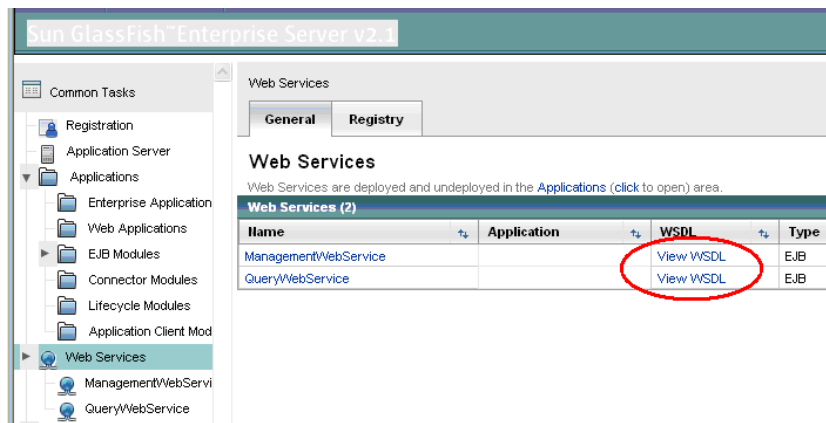
In the Servers view, right click on the Glassfish server and select GlassFish Enterprise Server->View Admin Console.



In the login page, enter the username and password. The default values for Glassfish are “admin” for the username and “adminadmin” for the password.



Click on the “Web Services” link to view the web services.



Click on the “View WSDL” link to view the WSDL and the URL of the service.