

Getting Started Guide

Version: 3.3.0.Beta1

1. Installation Instructions	1
1.1. Installing JBoss Tools Plugins	1
1.2. Usage Reporting	10
1.2.1. Collected usage information guide	11
2. JBoss Central	13
2.1. Getting Started with JBoss Central	13
2.2. Software installation and updates from within JBoss Central	23
2.3. Project Example Installation	29
3. JBoss Perspective	33
4. Setting up a JBoss runtime and managing the server	37
4.1. Adding and configuring a JBoss server runtime	37
4.2. Starting JBoss Server	41
4.3. Stopping the JBoss Server	42
4.4. Server Container Preferences	42
5. Developing a simple JSP web application	45
5.1. Setting Up the Project	45
5.2. Creating JSP Page	47
5.2.1. Editing a JSP Page	49
5.2.2. web.xml file	50
5.2.3. Deploying the project	52
5.2.4. JSP Page Preview	55
5.2.5. Launch JSP Project	55
6. Rapid Application Development of a JSF application	57
6.1. Setting up the project	57
6.2. Creating JSP Pages	60
6.3. Creating Transition between two views	63
6.4. Creating Resource File	65
6.5. Creating a Java Bean	69
6.6. Editing faces-config.xml File	74
6.7. Editing the JSP View Files	75
6.7.1. Editing inputnumber.jsp page	75
6.7.2. Editing success.jsp page	86
6.8. Creating index.jsp page	88
6.9. Running the Application	89
7. Uninstalling the JBoss Developer Studio	95
8. FAQ	97
8.1. What should I do if the Visual Page Editor does not start under Linux	97
8.2. Visual Editor starts OK, but the Missing Natures dialog appears	98
8.3. I have an existing Seam 1.2.1 project. Can I migrate or import the project into a JBoss Developer Studio Seam project?	99
8.4. I have an existing Struts or JSF project. Can I open the project in JBoss Developer Studio?	99
8.5. Can I import a WAR file?	99
8.6. Is it possible to increase the performance of Eclipse after installing your product?..	99

8.7. How can I add my own tag library to the JBoss Tools Palette?	99
8.8. How to get Code Assist for Seam specific resources in an externally generated project?	100
8.9. How to import an example Seam project from jboss-eap directory?	100
8.10. Is a cross-platform project import possible for JBoss Developer Studio?	100
9. Further Reading	101

Installation Instructions

1.1. Installing JBoss Tools Plugins

The JBoss Tools plugins can be installed in Eclipse from the JBoss.org update site. JBoss Tools 3.2 requires Eclipse 3.6, which can be downloaded from the [Eclipse web site](http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/helios/) [http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/helios/].

To install the JBoss Tools plugins start Eclipse and select **Help** → **Install New Software...**

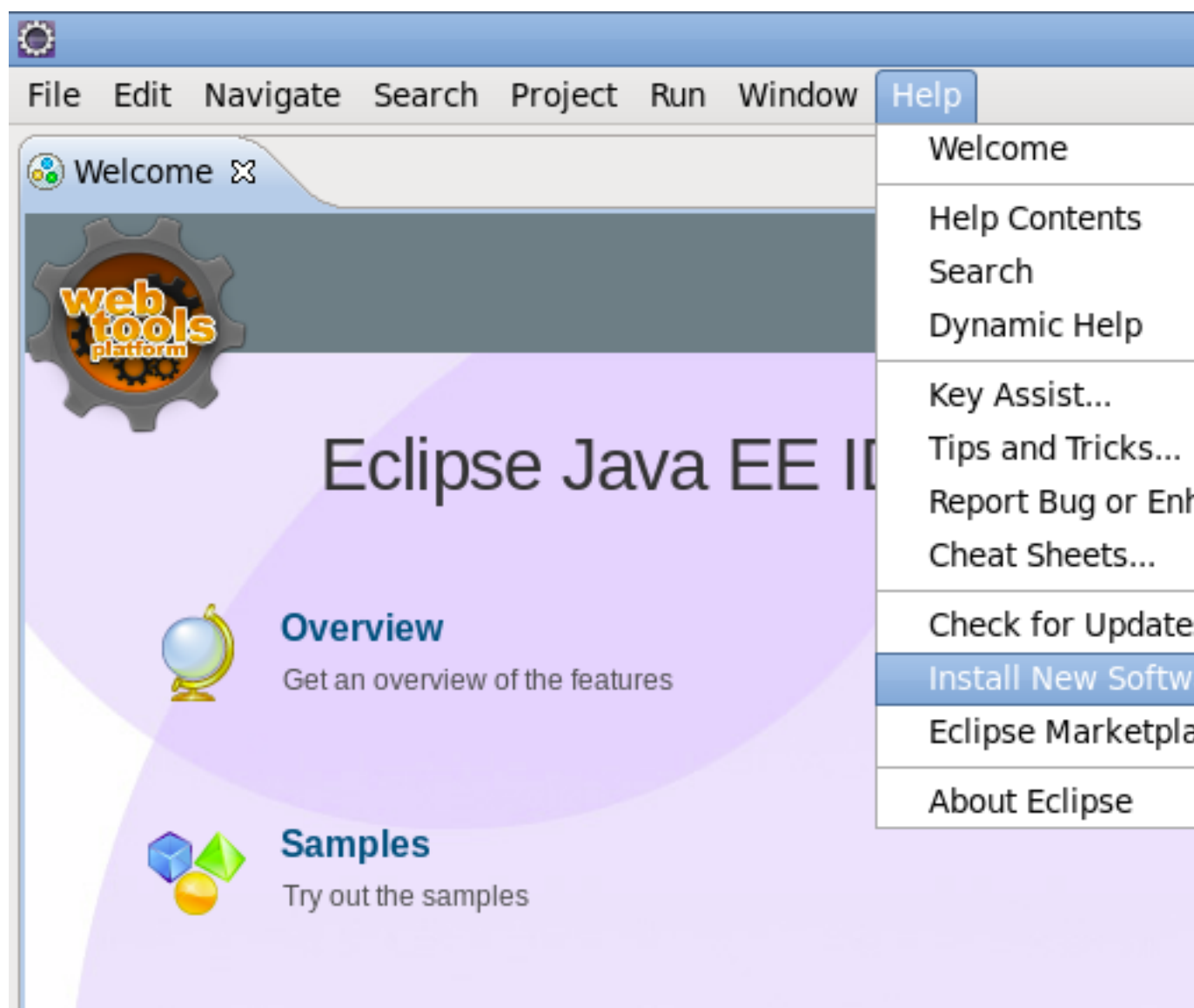


Figure 1.1. Install New Software

Click the **Add...** button.

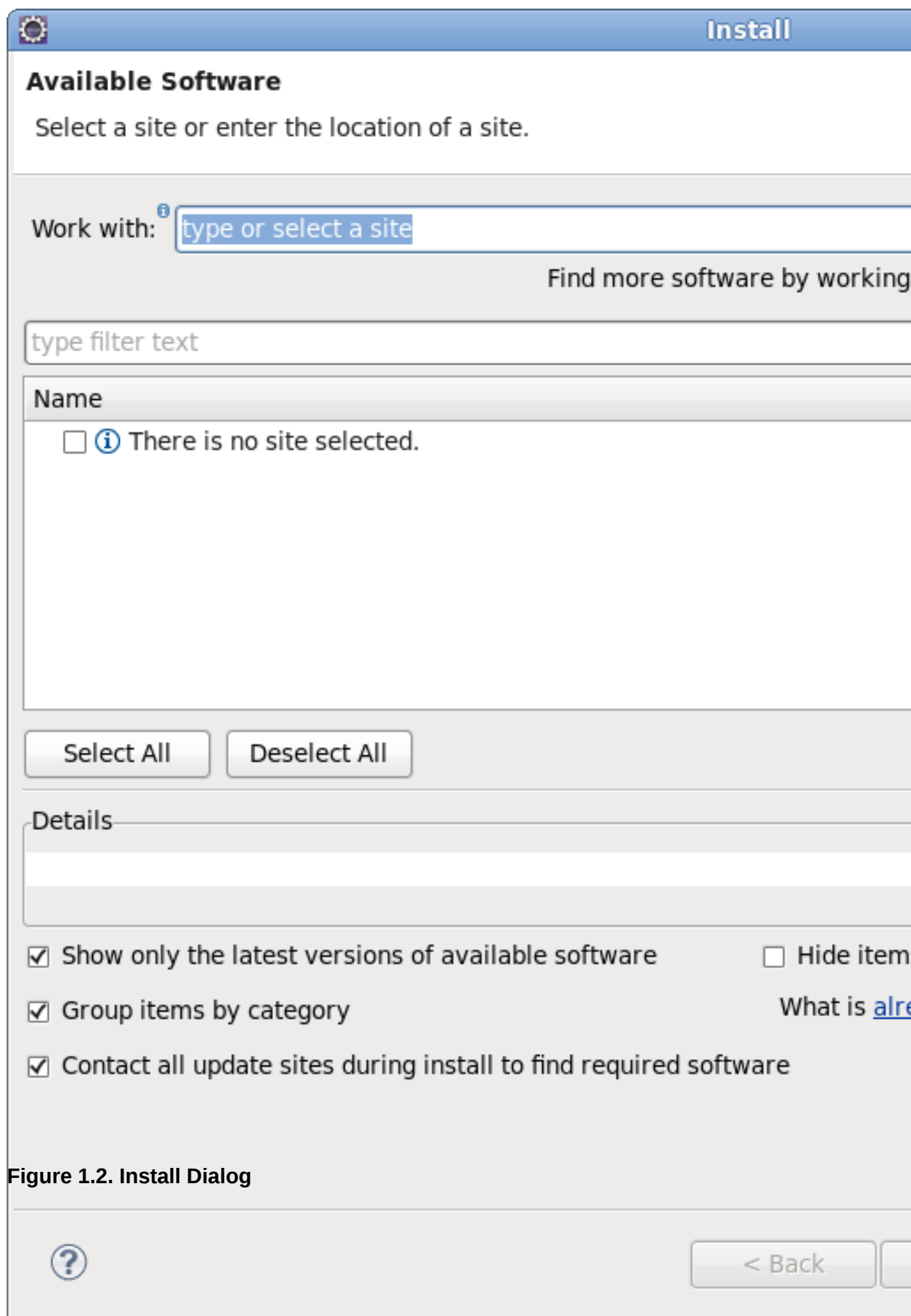


Figure 1.2. Install Dialog

This will display the **Add Repository** dialog. Enter **JBoss.org Tools** in the **Name** field, and <http://download.jboss.org/jbosstools/updates/JBossTools-3.2.0.GA/> in the **Location** field. Click the **OK** button to save the changes and close the dialog.

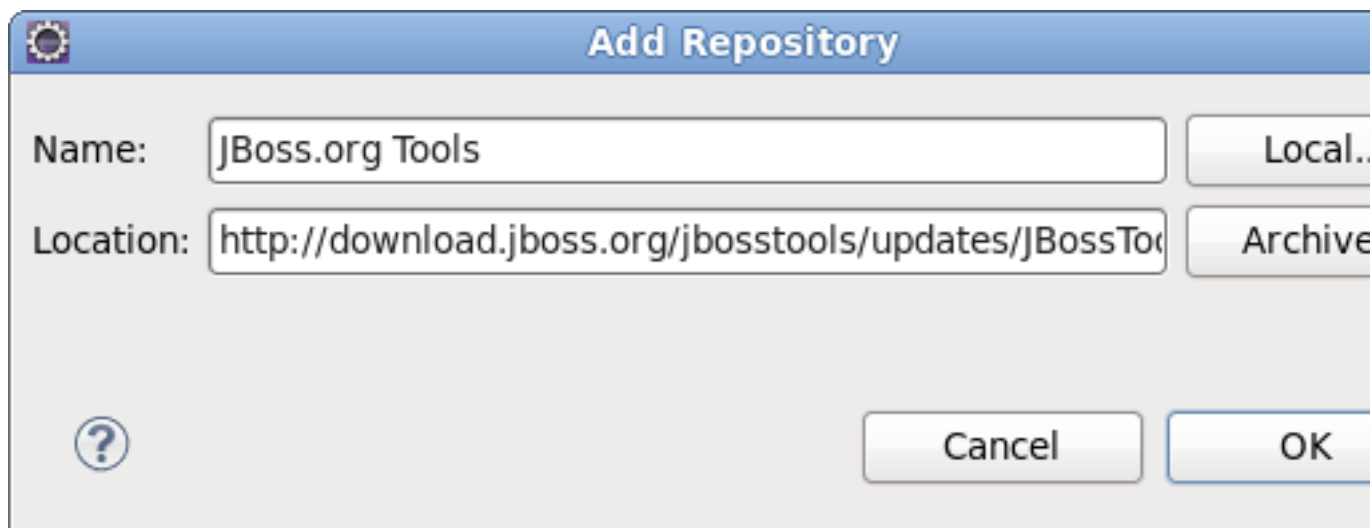


Figure 1.3. Add Repository

The **JBoss.org Tools** site will be selected in the **Work with** drop down list, and after a moment the list of plugins that are included in the JBoss Tools package will be listed. From this list you can individually select the desired plugin, or select the **All JBoss Tools 3.2.0** option to install all the plugins.

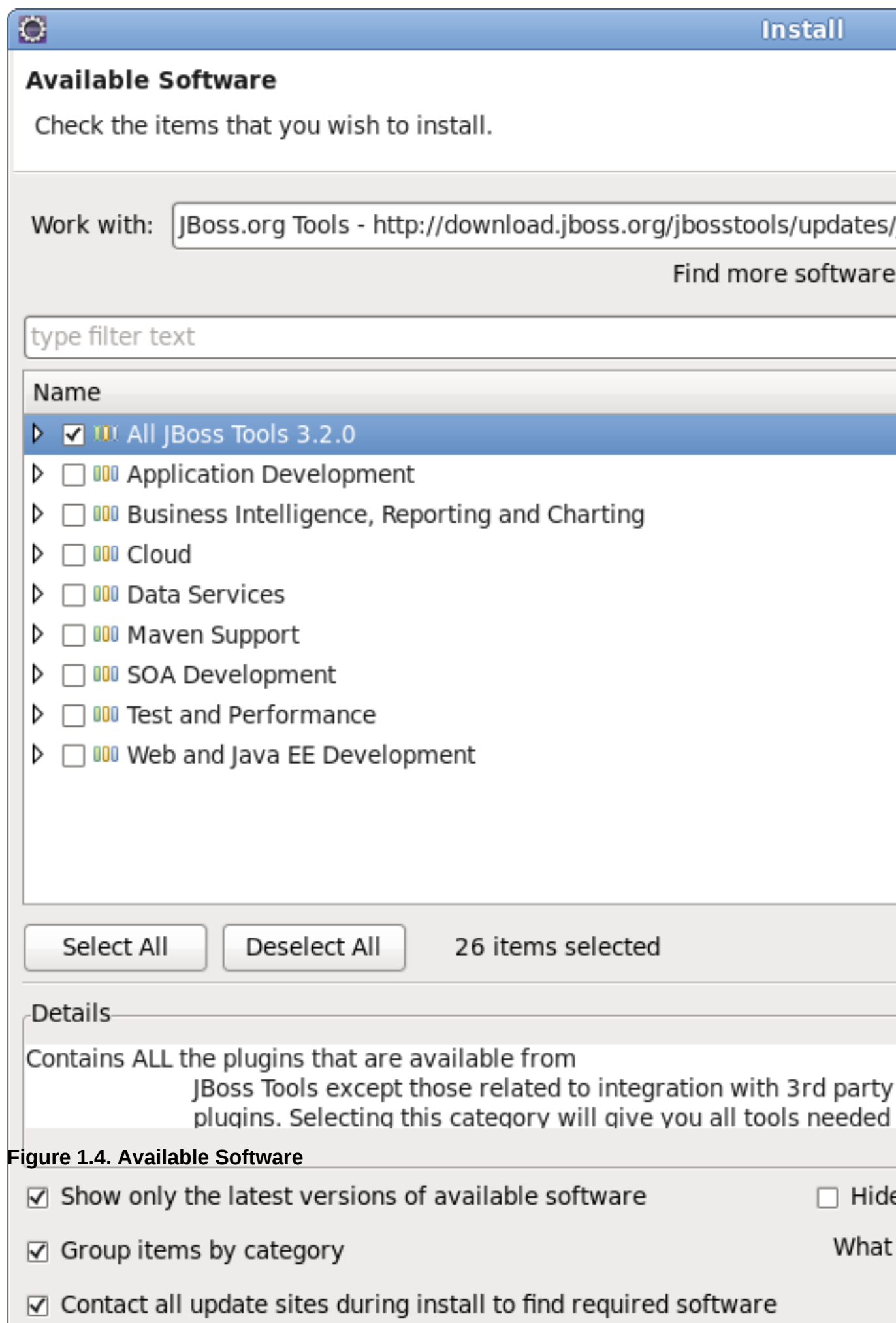


Figure 1.4. Available Software

Click the **Next** button to calculate the system requirements and dependencies (this may take a little while). You will then be given an opportunity to review the plugins that will be installed.

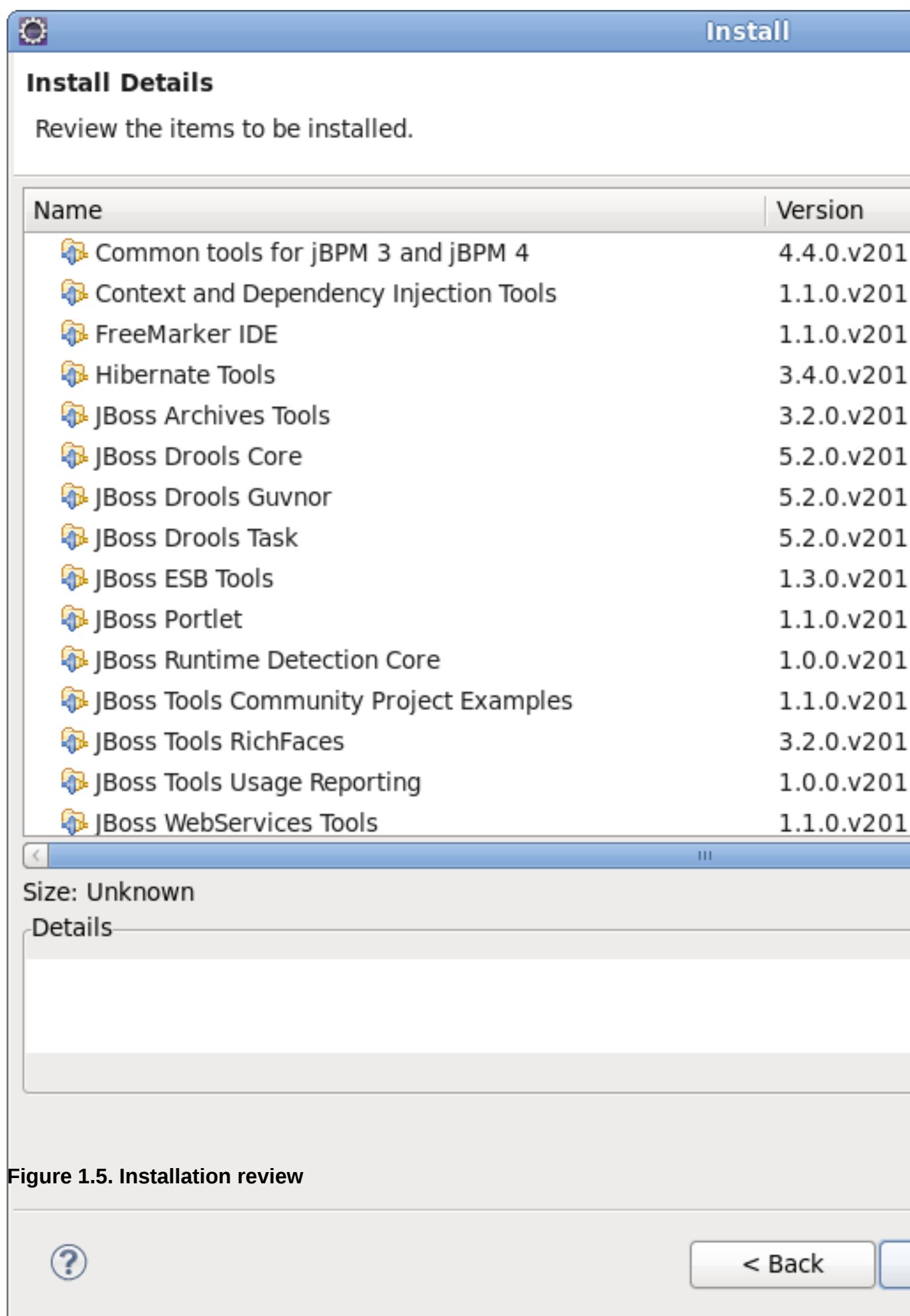


Figure 1.5. Installation review

Click the **Next** button to install the selected plugins. You will be prompted to accept the various license agreements that cover the plugins that are to be installed. Review the licenses, select the **I accept the terms of the license agreements** option, and click the **Finish** button to install the plugins.

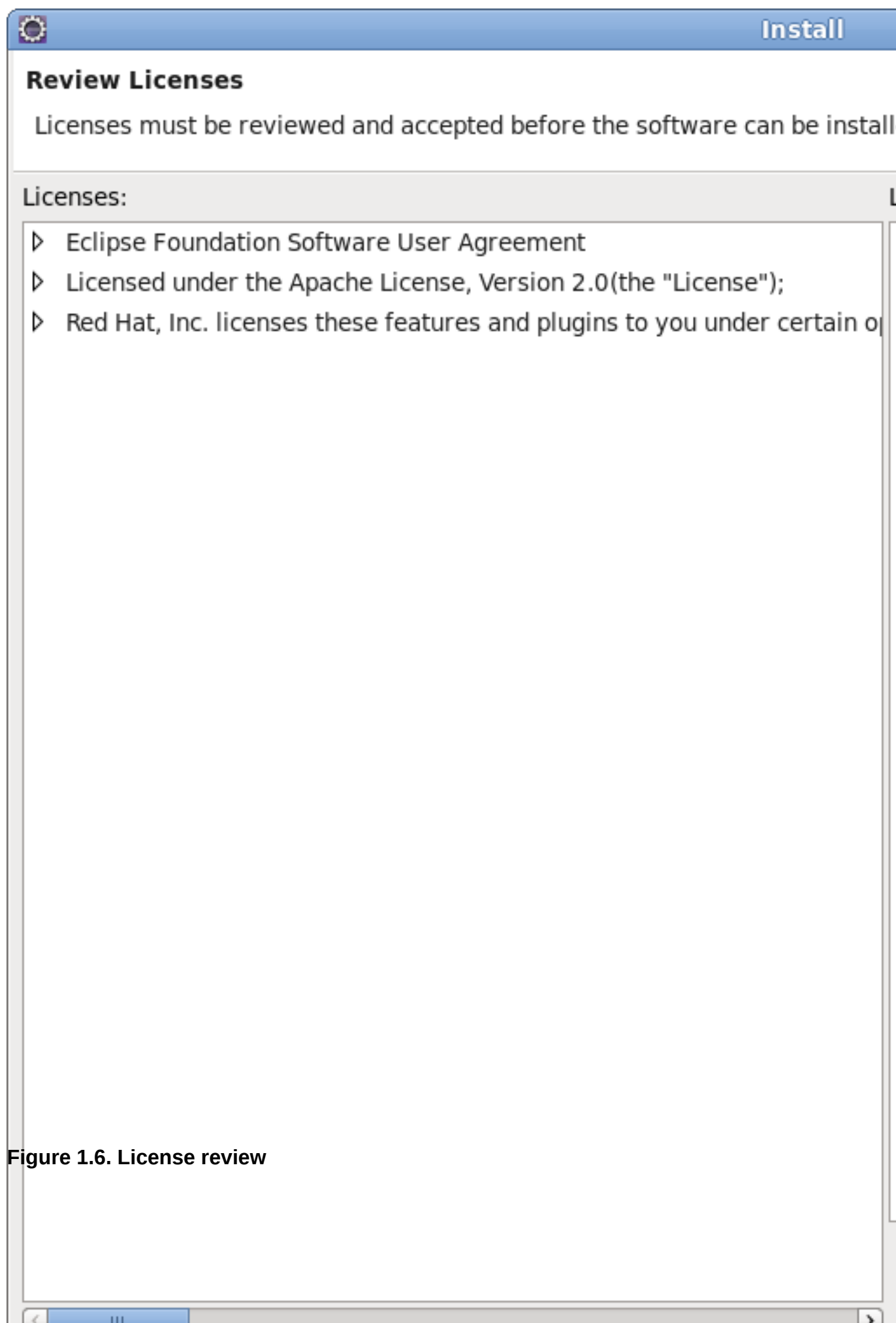


Figure 1.6. License review

Wait while the plugins are downloaded and installed.

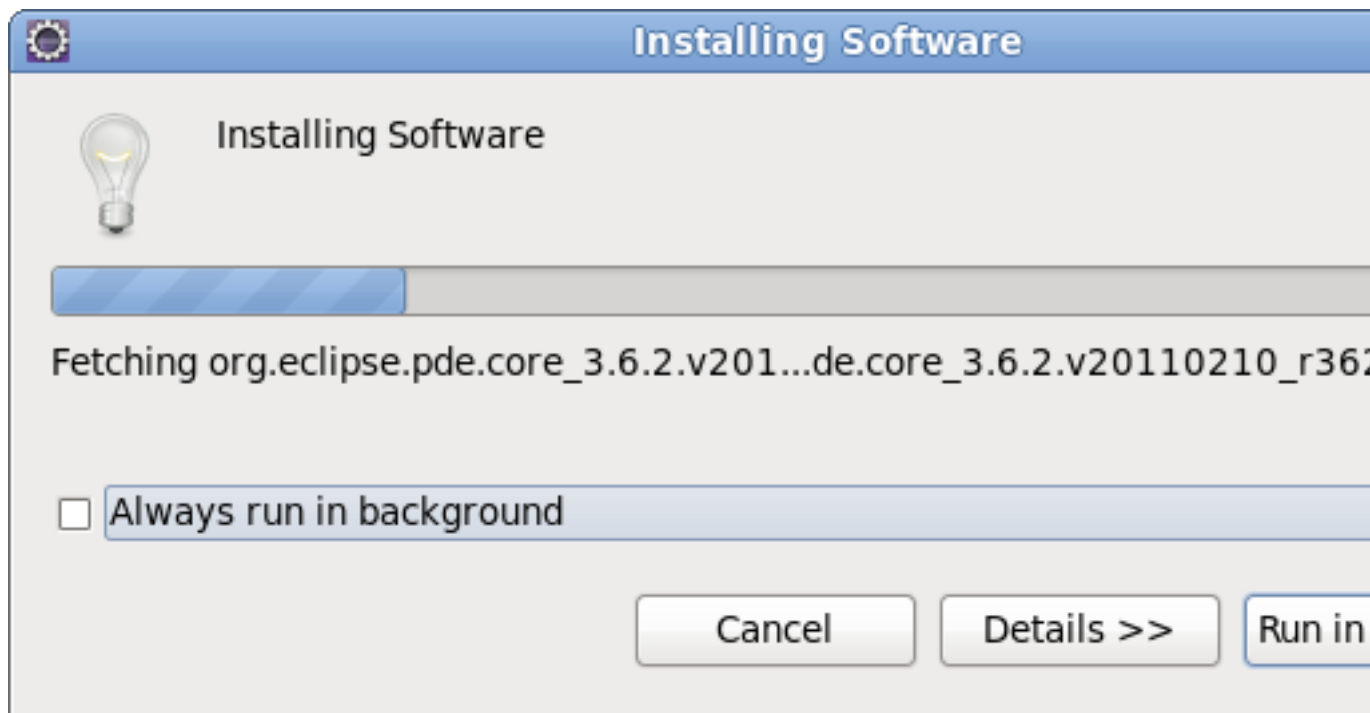


Figure 1.7. Installing Software

You may be prompted with a warning informing you that you are attempting to install unsigned content. Click the **OK** button to continue.



Figure 1.8. Unsigned Software Warning

You will then have to restart Eclipse to apply the new plugins. Click the **Restart Now** button to restart Eclipse.

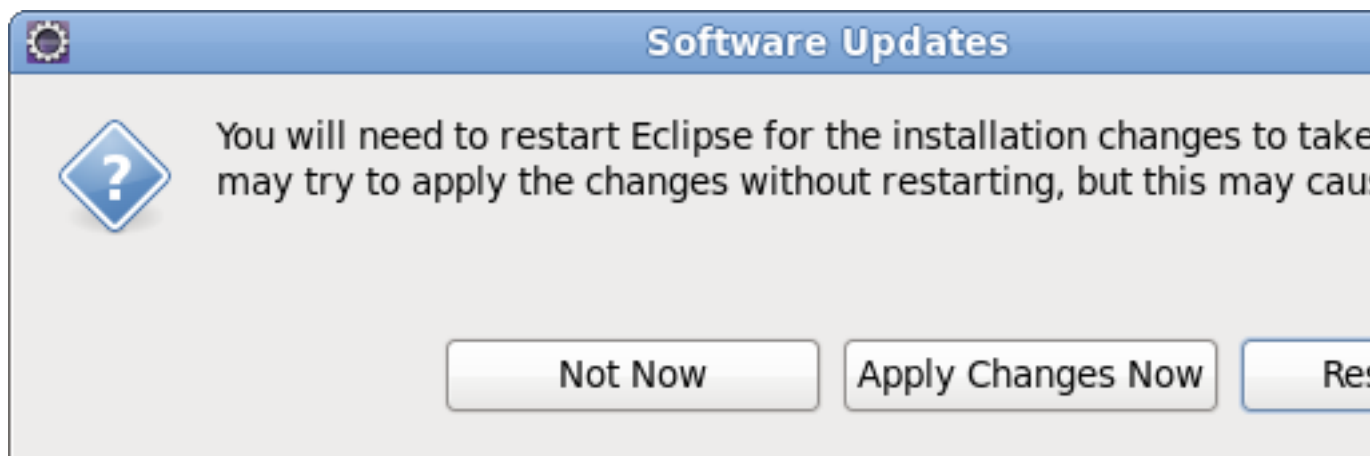


Figure 1.9. Restart Eclipse

The plugin is now installed and ready to use.

1.2. Usage Reporting

JBoss Tools now includes a usage plug-in that anonymously reports information back to JBoss. The plug-in is not enabled by default. To enable, click the **Yes** button.

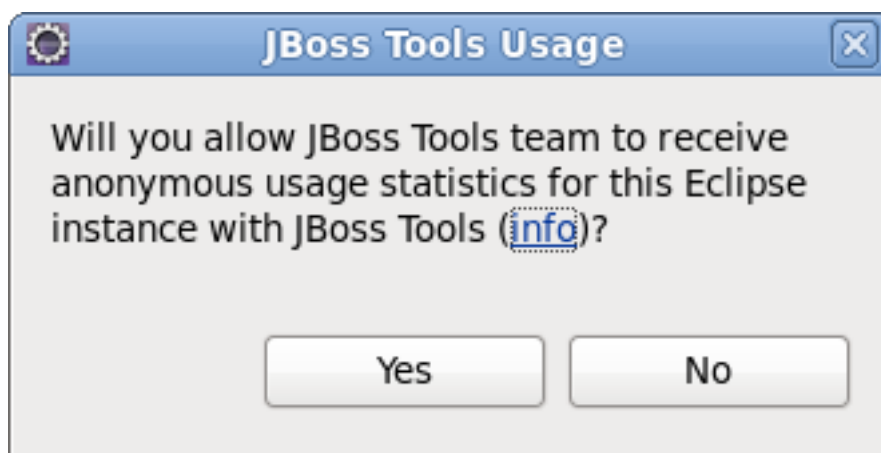


Figure 1.10. Usage plug-in pop-up

Once enabled, the plug-in will remain active until turned off. To turn the active plug-in off, navigate to **Window** → **Preferences** → **JBoss Tools** → **Usage Reporting**.

The gathered data allows JBoss to see how the tools are being used and where they are being used geographically. Currently we are looking into the operating systems being used, screen resolution and how often the tooling environment is started. In the future geographic information will assist in focusing translation resources to areas where the developer environment is most used.

The plug-in uses Google Analytics to track and report data by acting as if you were visiting the site <http://jboss.org/tools/usage/>. To view the type of information being collected, refer to [Section 1.2.1, “Collected usage information guide”](#).

To view the source code of the usage plug-in visit <http://anonsvn.jboss.org/repos/jbosstools/trunk/usage/>.

1.2.1. Collected usage information guide

Below you will find an outline of the information that is reported and the Google Analytics fields that are used to gather this information.

Version

The **Content** field has been modified to report the installed JBoss Developer Studio version. Sample returned values include: `jbdevstudio-linux-gtk-x86_64-4.0.0.v201009301221R-H20-Beta1.jar` and `jbdevstudio-linux-gtk-3.0.2.v201009161622R-H138-GA.jar`.

Installed components

The **Keyword** field has been modified to report the installed JBoss Developer Studio components. Sample returned values include: JBoss AS, Drools, Teiid and ModeShape.

Visitor type

The **Visitor type** field reports if the current user is new or returning.

Language

The **Language** field reports the localized language the product is being used in. Sample returned values include: `en-US`, `de-DE` and `fr-FR`.

Location fields

The location fields report the geographical location where the product is being used based on the continent, country and city. Sample returned values include: `Europe` (continent), `Germany` (country) and `Munich` (city).

Eclipse interface and version

The **Browser** field has been modified to report the Eclipse interface and version being used. Sample returned values include: `JBoss Developer Studio: 5.0.0` and `JBoss Developer Studio: 5.0.1`.

Operating System

The **Operating System** field reports the Operating System and its version that the product is running on (with Linux distribution version reporting conducted through the **User Defined Value** field). Sample returned values include: `Linux`, `Macintosh 10.6`, `Macintosh 10.7` and `Windows 7`.

Linux distribution version

The **User Defined Value** field reports the distribution and version of Linux, if one is being used as the Operating System. Sample returned values include: `Red Hat Enterprise Linux 6` and `Fedora 16`.

Screen colors

The **Screen colors** field reports the color depth being used. Sample returned values include: 32-bit and 24-bit.

Screen resolution

The **Screen resolution** field reports the resolution being used. Sample returned values include: 2048x1536 and 1920x1080.

Java version

The **Flash version** field has been modified to report the Java version used. Sample returned values include: 1.6.0_20.

Connection speed

The **Connection speed** field reports the type of internet connection being used. Sample returned values include: T1, Cable and DSL.

JBoss Central Enabled

The **JBoss Central Enabled** field reports whether JBoss Central is set to be seen upon startup or not. Returned value is either true or false.

JBoss Central

When viewing the workbench for the first time you will be greeted with *JBoss Central*. JBoss Central assists you in getting started and keeps you up to date with JBoss technologies.

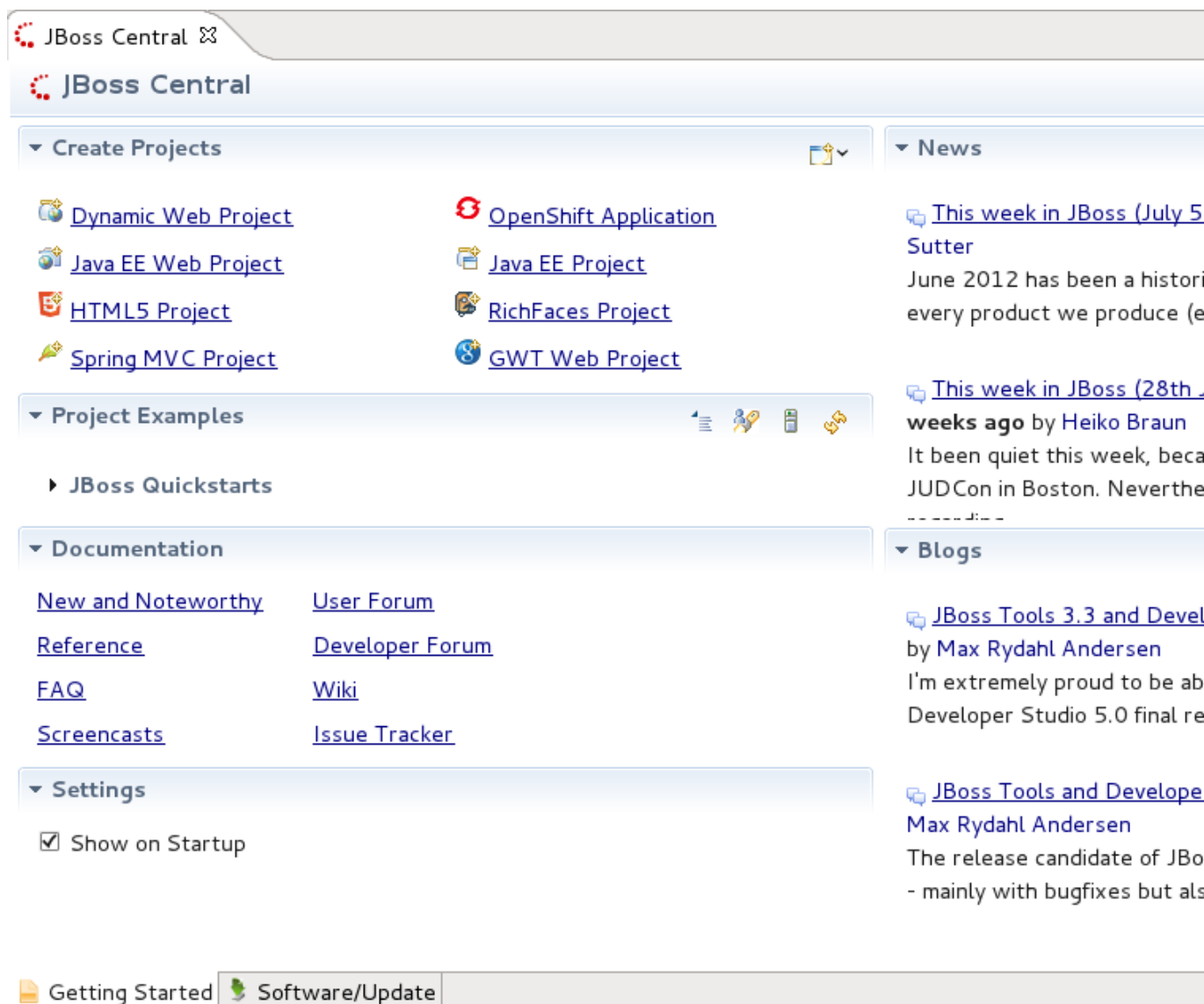


Figure 2.1. JBoss Central

2.1. Getting Started with JBoss Central

The **Getting Started** tab of JBoss Central is the first tab you will see, and is setup as a functional starting point for developers.

At the top of the tab is a search box. By clicking the down arrow to the left you can select to perform a search on either the **Red Hat Customer Portal** or the **JBoss Community**.

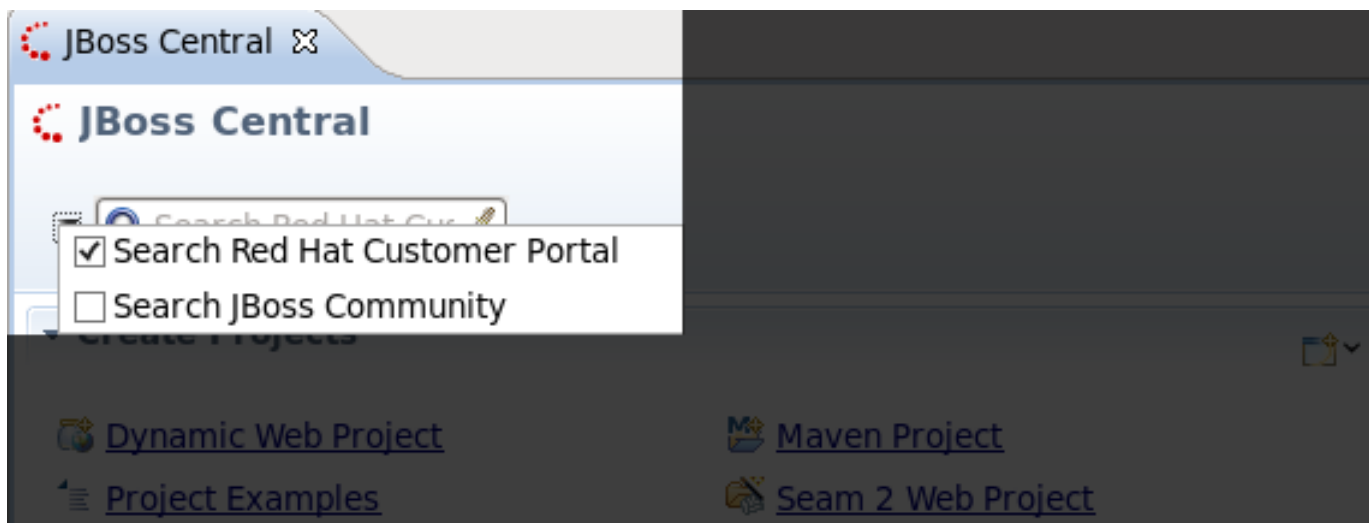


Figure 2.2. Search Options

Performing a search will launch a web browser as a new tab, displaying the results page. You can interact with the browser as you would any other web browser.

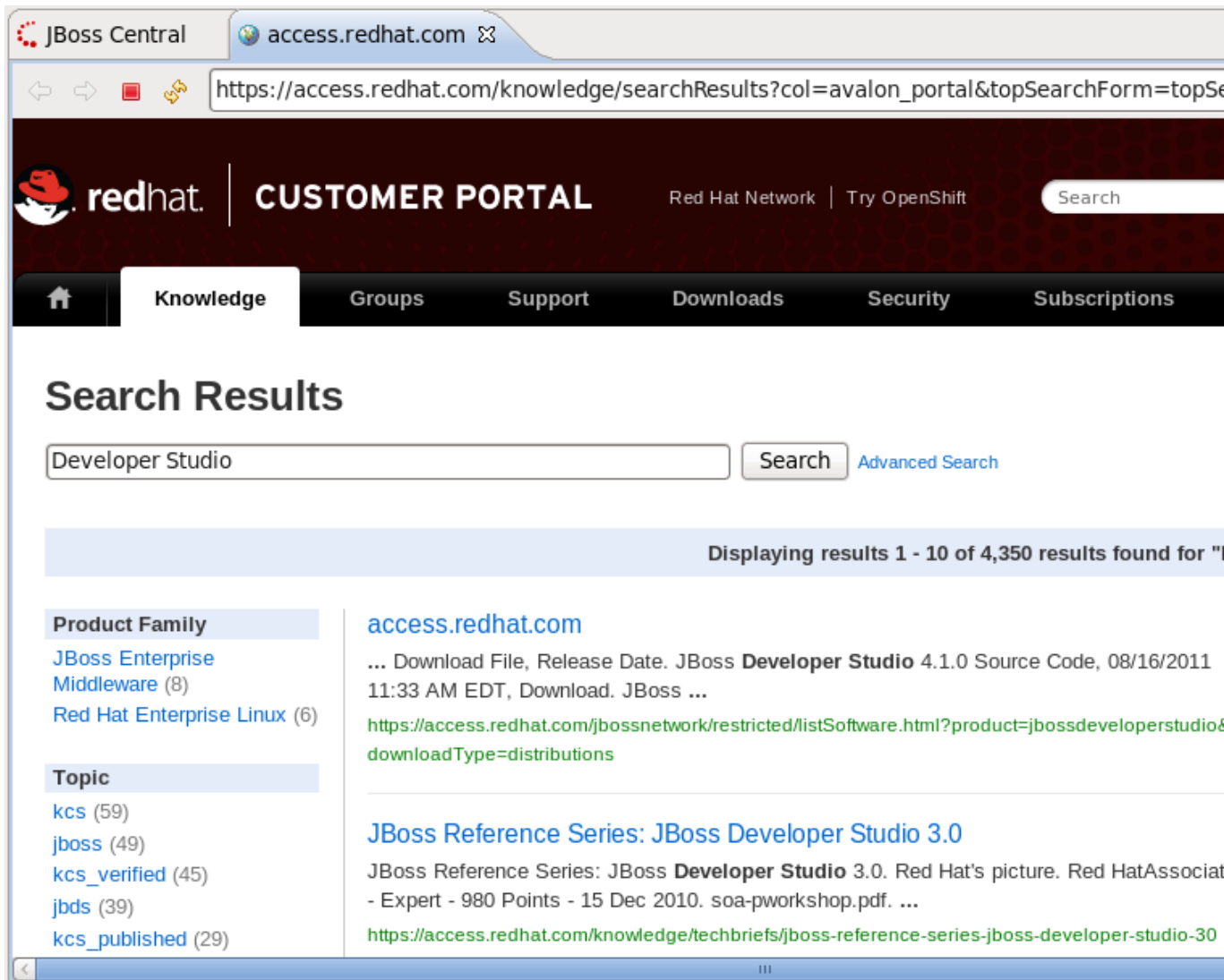


Figure 2.3. Search Results

From the **Create Projects** section you can create a **Dynamic Web Project**, **OpenShift Express Application**, **Java EE Web Project**, **Java EE Project**, **HTML5 Project**, **Spring MVC Project**, **RichFaces Project**, **GWT Web Project**, or any one of many **Project Examples**. To access a complete list of projects you can create, click on the window icon at the top-right of the **Create Projects** section



Note

All wizards generate Maven based projects, except for the **Dynamic Web Project** wizard.

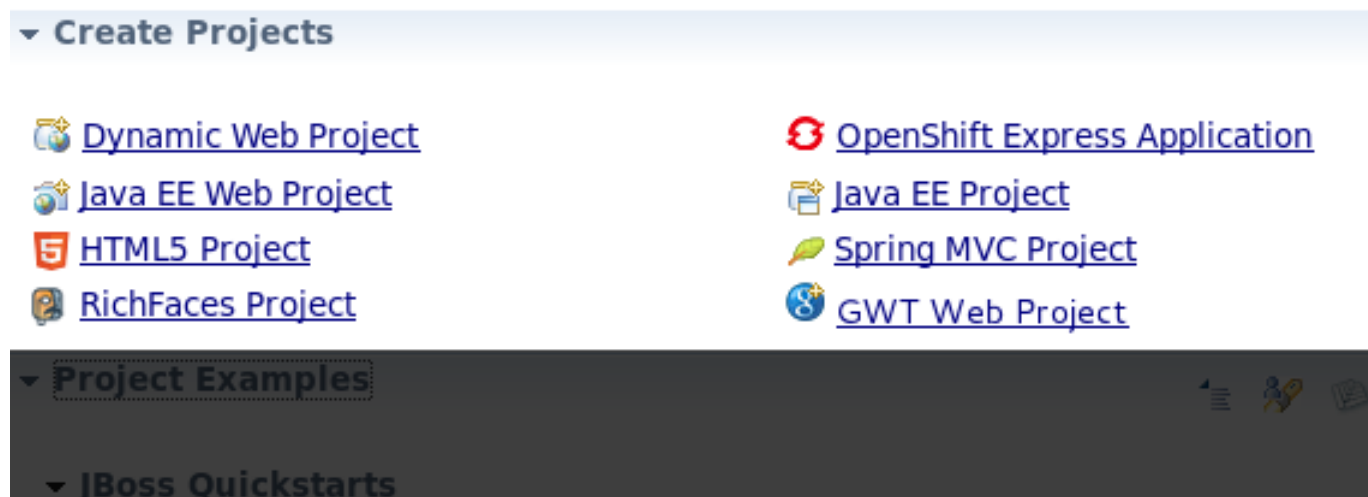


Figure 2.4. Creating a Project

GWT (Google Web Toolkit) Web Project creation. To create a GWT web project, the latest m2e-wtp and Google plug-ins have to be installed. If these plug-ins have not been previously installed, the first page of the GWT web project wizard will provide you with the ability to install the plug-ins.

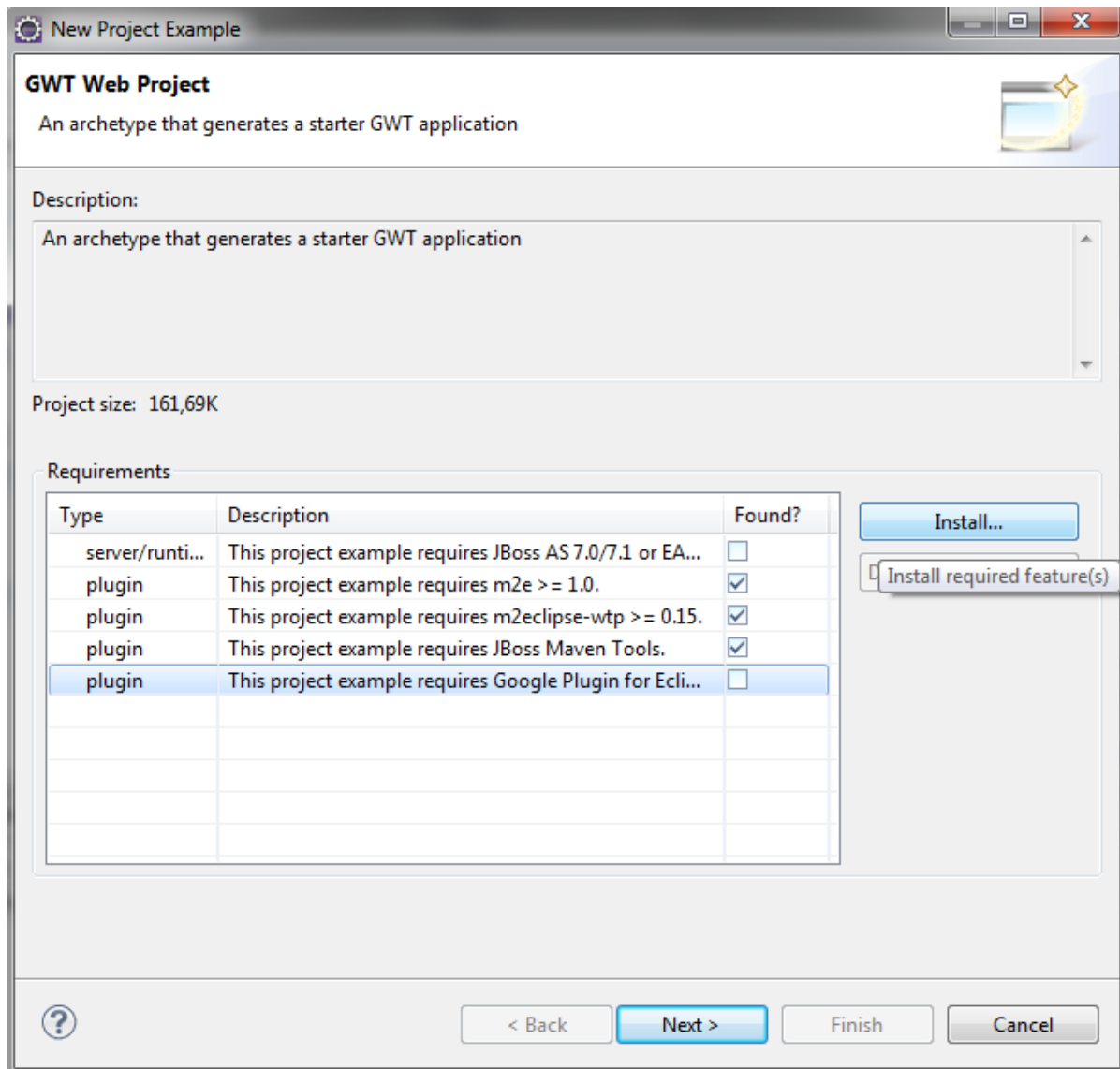


Figure 2.5. GWT Web Project creation: plug-in page

Under the **Project Examples** section you can expand and view **JBoss Quickstarts**. Each quickstart is an example to assist first time users.

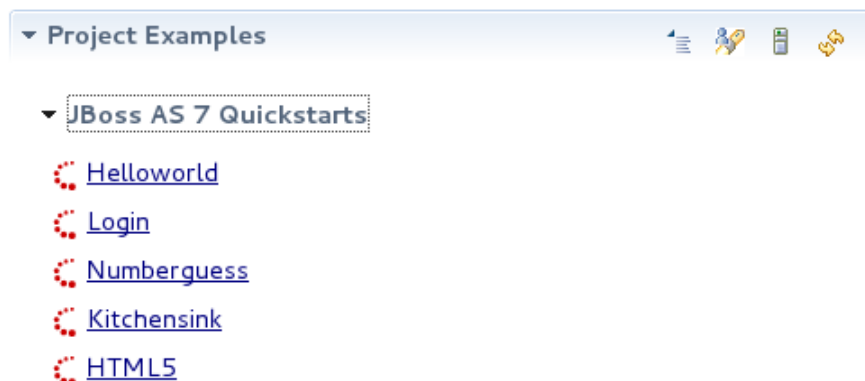


Figure 2.6. Project Examples

For an in-depth look into the installation of project examples go to [Section 2.3, “Project Example Installation”](#).

You can also download other examples and install and set runtime preferences through the **Project Examples** section by using the five buttons at the top-right. The first button



launches a **New Project Examples** wizard. Here you can search and download project examples to assist you with getting started.

Project Example

Import Project Example

☐ Show experimental sites

☐ Show runtime sites

Site: All

Targeted runtime: All

Projects:

type filter text

- JBoss Maven Archetypes
- Seam
- Portlet
- ESB for JBoss Enterprise SOA Platform 4.3
- ESB for JBoss Enterprise SOA Platform 5.0
- Teiid Designer

Description:

Project name:

Project size:

URL:

? < Back Next > Cancel Finish

Figure 2.7. New Project Examples wizard

By clicking on the second button



you will be taken directly to the **JBoss Tools Runtime Detection** dialog within **Preferences**.

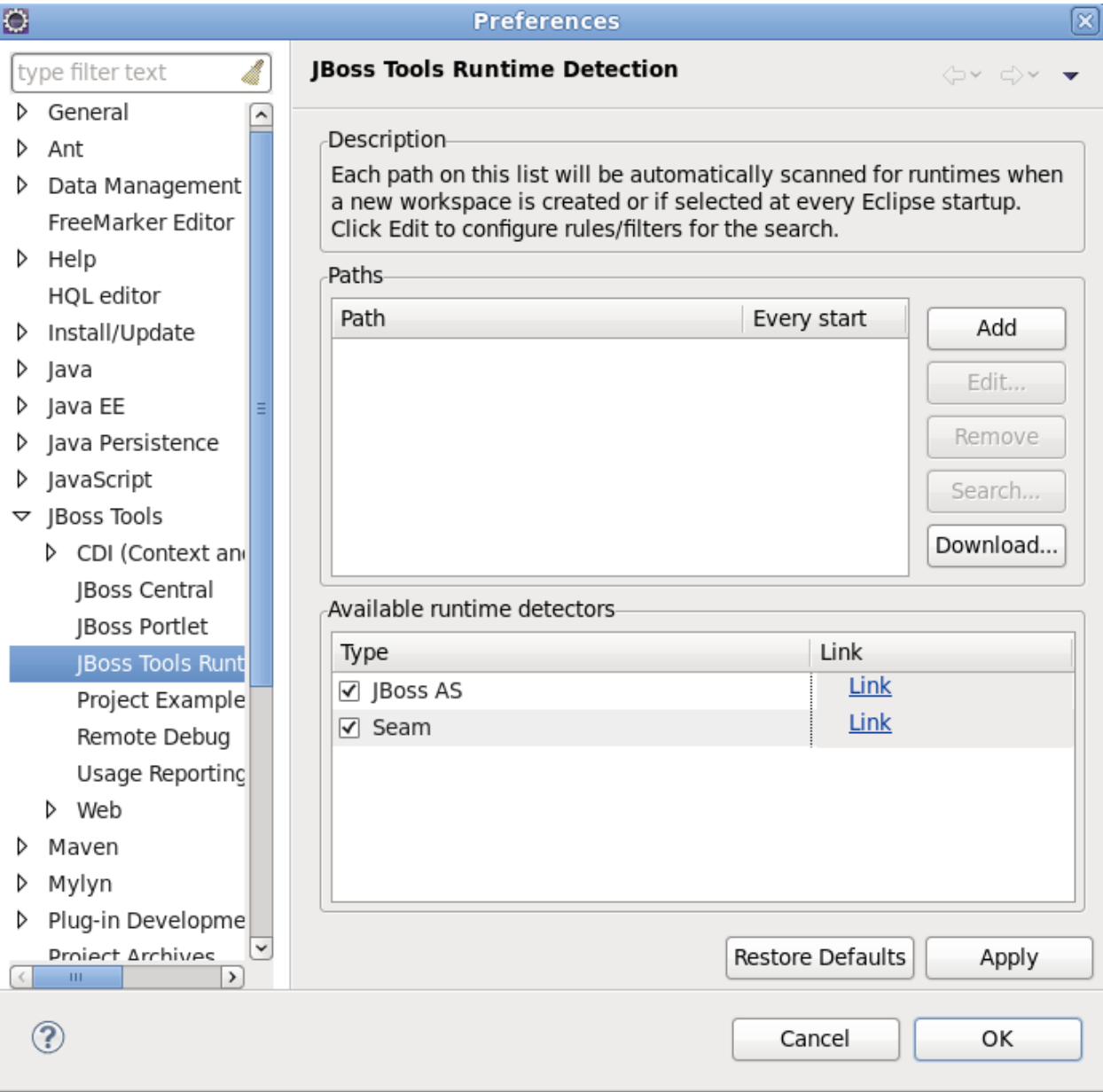


Figure 2.8. JBoss Tools Runtime Detection

The

third



button takes you directly to the **Server Runtime Environments** preferences dialog.

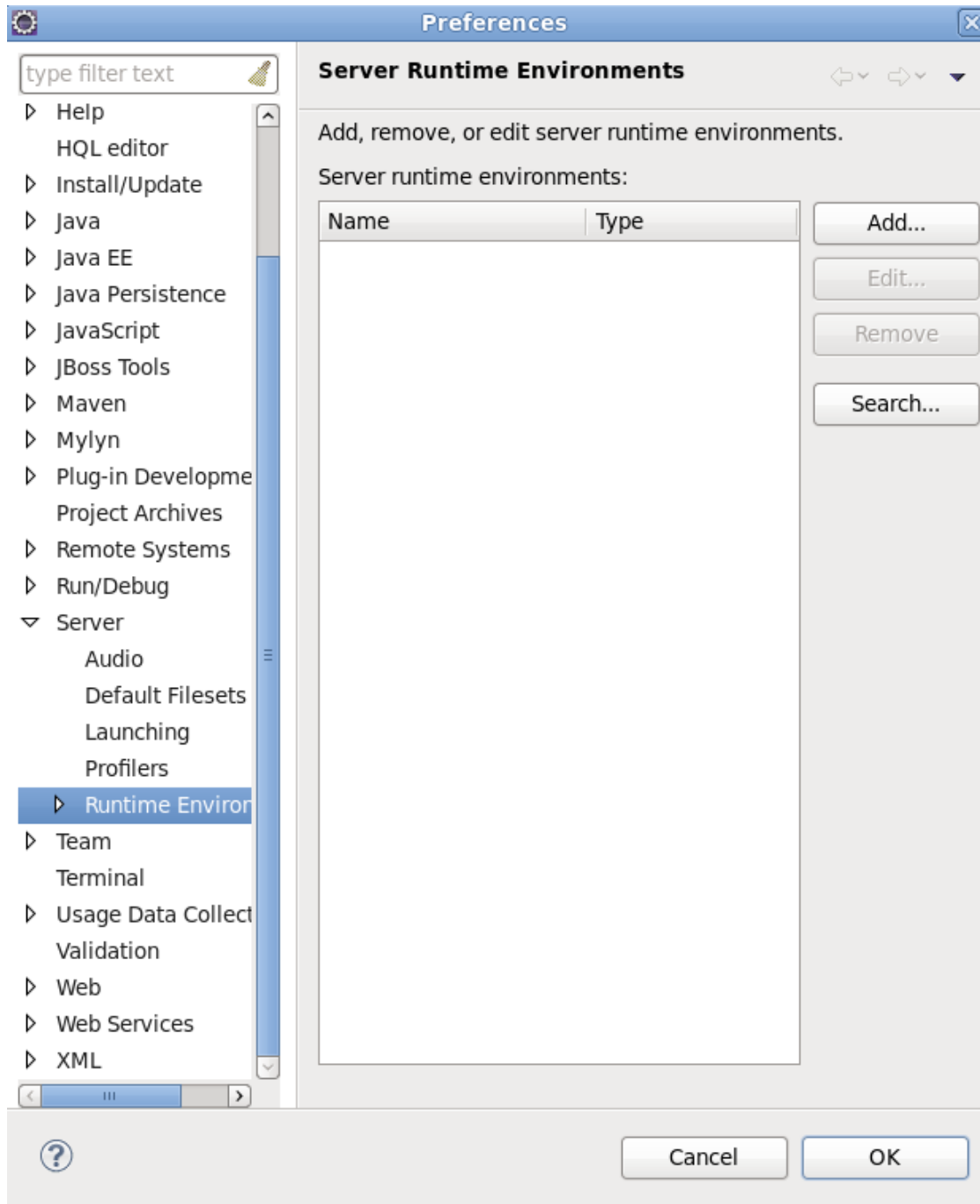


Figure 2.9. Server Runtime Environments preferences

The [Add...](#) button



refreshes the **Project Examples** list.

The **Documentation** section of JBoss Central contains links to help materials such as reference guides, user forums and documentation concerning new features.



Figure 2.10. Documentation links

If you do not wish to see JBoss Central at every startup, you can deselect the checkbox **Show on Startup** in the **Settings** section.

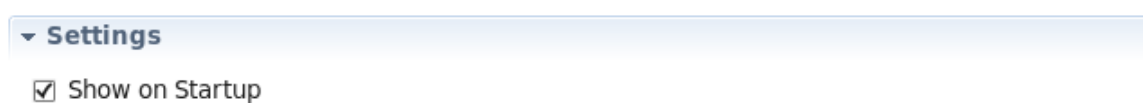
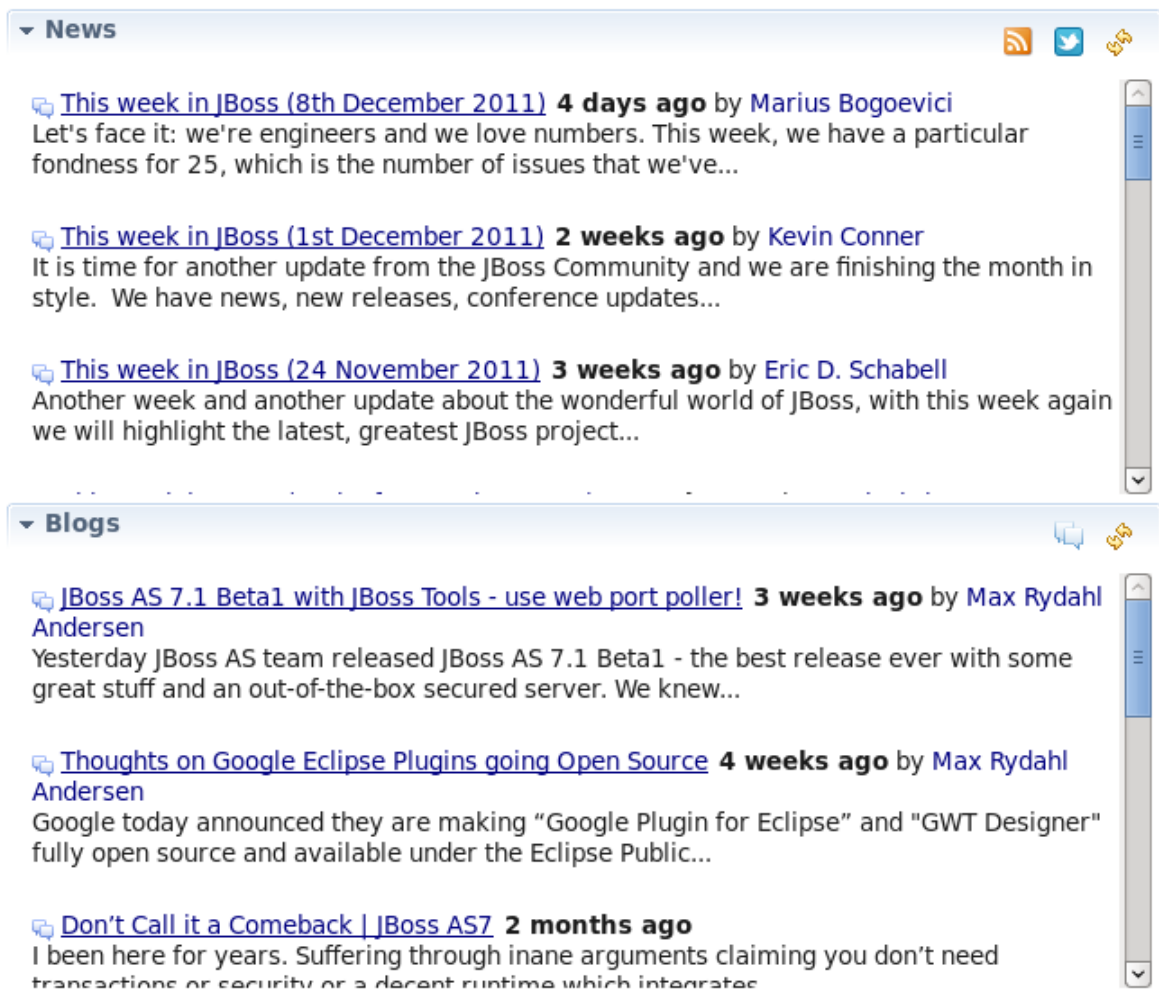


Figure 2.11. Settings section

The **News** and **Blogs** sections are updated automatically, bringing you the latest information concerning JBoss technologies.



The screenshot shows two sections of the JBoss Central interface. The top section, titled 'News', contains three entries: 'This week in JBoss (8th December 2011) 4 days ago' by Marius Bogoevici, 'This week in JBoss (1st December 2011) 2 weeks ago' by Kevin Conner, and 'This week in JBoss (24 November 2011) 3 weeks ago' by Eric D. Schabell. The bottom section, titled 'Blogs', contains three entries: 'JBoss AS 7.1 Beta1 with JBoss Tools - use web port poller!' 3 weeks ago by Max Rydahl Andersen, 'Thoughts on Google Eclipse Plugins going Open Source' 4 weeks ago by Max Rydahl Andersen, and 'Don't Call it a Comeback | JBoss AS7' 2 months ago. Each entry includes a brief text snippet. The interface features a sidebar on the right with navigation icons and a vertical scrollbar.

News

[This week in JBoss \(8th December 2011\)](#) **4 days ago** by [Marius Bogoevici](#)
Let's face it: we're engineers and we love numbers. This week, we have a particular fondness for 25, which is the number of issues that we've...

[This week in JBoss \(1st December 2011\)](#) **2 weeks ago** by [Kevin Conner](#)
It is time for another update from the JBoss Community and we are finishing the month in style. We have news, new releases, conference updates...

[This week in JBoss \(24 November 2011\)](#) **3 weeks ago** by [Eric D. Schabell](#)
Another week and another update about the wonderful world of JBoss, with this week again we will highlight the latest, greatest JBoss project...

Blogs

[JBoss AS 7.1 Beta1 with JBoss Tools - use web port poller!](#) **3 weeks ago** by [Max Rydahl Andersen](#)
Yesterday JBoss AS team released JBoss AS 7.1 Beta1 - the best release ever with some great stuff and an out-of-the-box secured server. We knew...

[Thoughts on Google Eclipse Plugins going Open Source](#) **4 weeks ago** by [Max Rydahl Andersen](#)
Google today announced they are making "Google Plugin for Eclipse" and "GWT Designer" fully open source and available under the Eclipse Public...

[Don't Call it a Comeback | JBoss AS7](#) **2 months ago**
I been here for years. Suffering through inane arguments claiming you don't need transactions or security or a decent runtime which integrates

Figure 2.12. News and Blogs sections



Note

After using JBoss Central once, the current content will be available offline. This includes project example archives and descriptors, maven artifacts, and news and blog feeds.

2.2. Software installation and updates from within JBoss Central

The **Software/Update** tab at the bottom of JBoss Central switches to a screen where you can install and update your software.

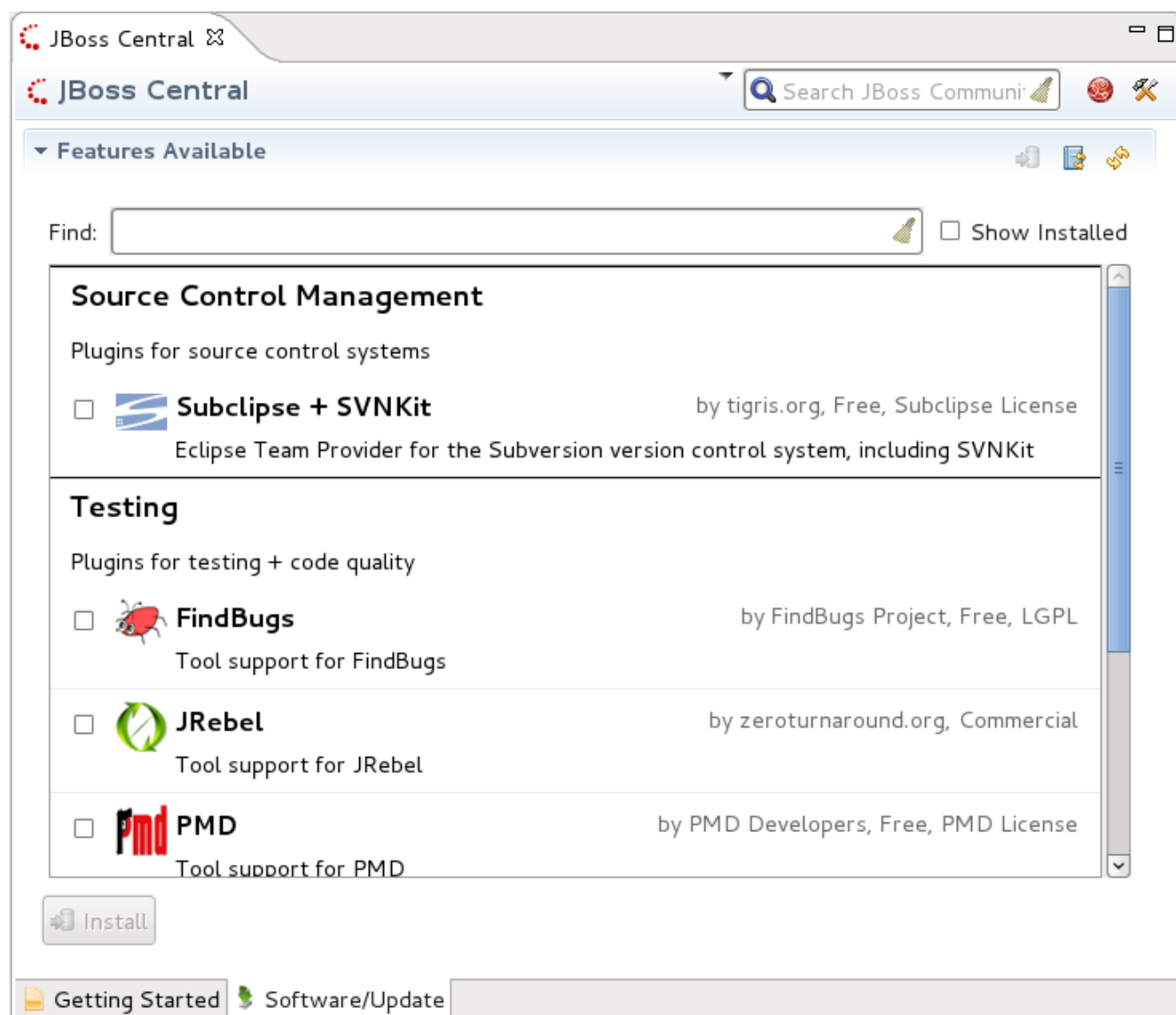


Figure 2.13. Software/Update tab

To install or update software, select the checkbox beside one or more components.

Source Control Management

Plugins for source control systems

- ☒ **Subclipse + SVNKit** by tigris.org, Free, Subclipse License
 Eclipse Team Provider for the Subversion version control system, including SVNKit

Figure 2.14. Software/Update tab

Once you have selected all the components to update or install, click the



button at the bottom left of the tab window. A screen will then appear offering you the option of deselecting a part of the component if you wish.

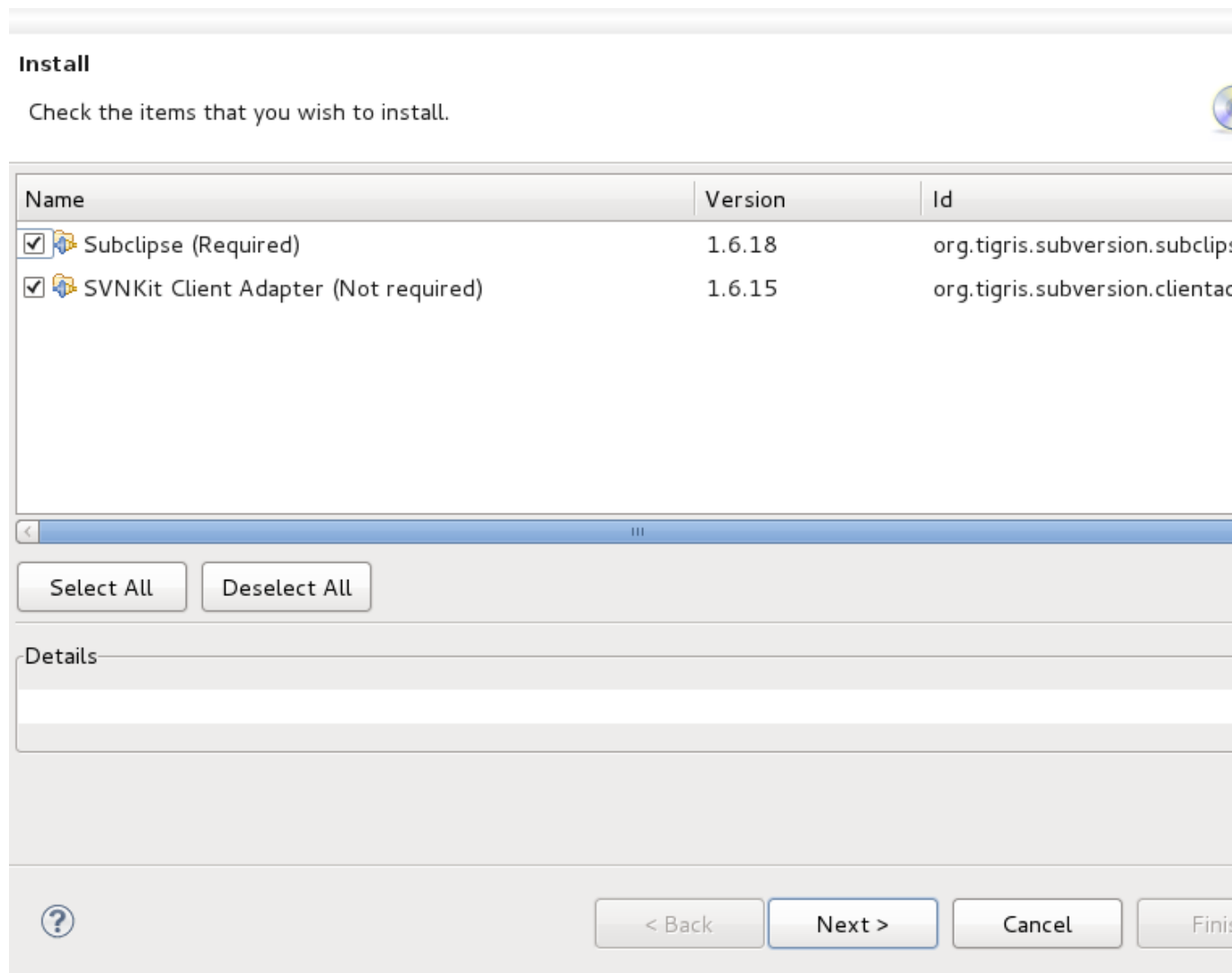


Figure 2.15. Component selection

You will then be asked to confirm the selection of components to install. To accept the selection press the **Next** button. You can change the components by pressing the **Back** button.

Install Details

Review the items to be installed.

Name	Version	Id
 Subclipse (Required)	1.6.18	org.tigris.subversion.subclips
 SVNKit Client Adapter (Not required)	1.6.15	org.tigris.subversion.clientad

Size: Unknown

Details



Figure 2.16. Component verification

After pressing the **Next** button you will need to **accept the terms of the license agreement** for the associated software being installed. To do this select the corresponding radio button.

Click **Finish** to begin installation.

Review Licenses

Licenses must be reviewed and accepted before the software can be installed.

Licenses:	License text:
▶ Subclipse Software User Agreement	Subclipse Software User Agreement 11th April, 2006 Subclipse is licensed under the terms of the Eclipse Public License v1.0. http://www.eclipse.org/legal/epl-v10.html Applicable Licenses Subclipse is built upon a number of other open source

☒ I accept the terms of the license agreement
☐ I do not accept the terms of the license agreement



Figure 2.17. Licenses review

As the installation process takes place you can choose to have further details displayed or run the installation in the background.

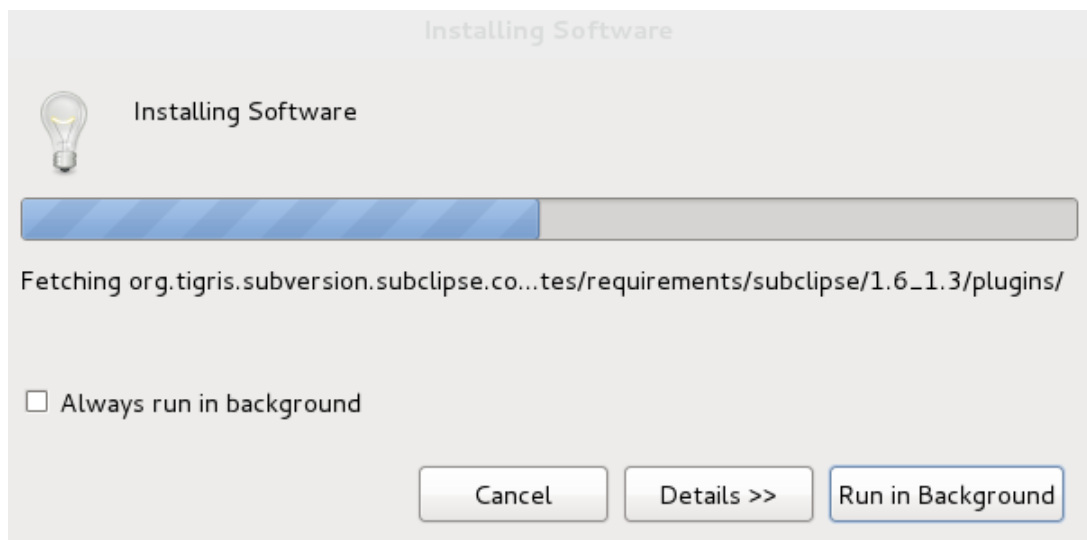


Figure 2.18. Licenses review

After the software has been downloaded, you will be asked to verify the installation of the components if the contents are unsigned. Install unsigned third-party components with discretion.

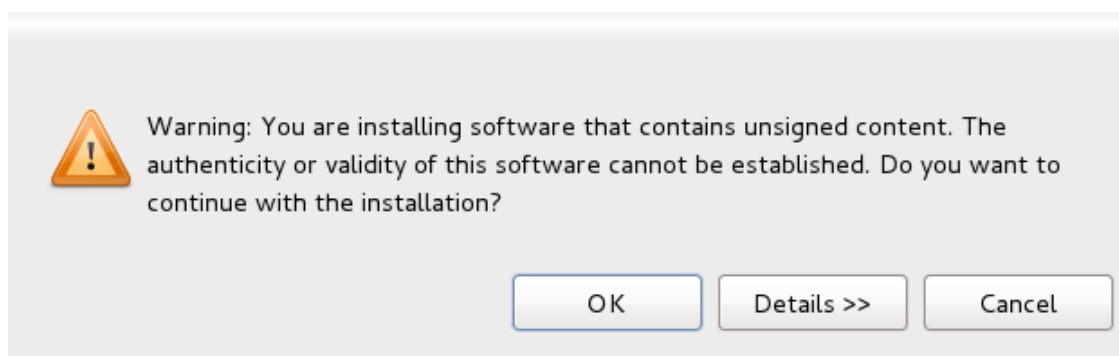


Figure 2.19. Installing unsigned content

Once installation is complete, restart JBoss Developer Studio by pressing the **Restart Now** button that appears.

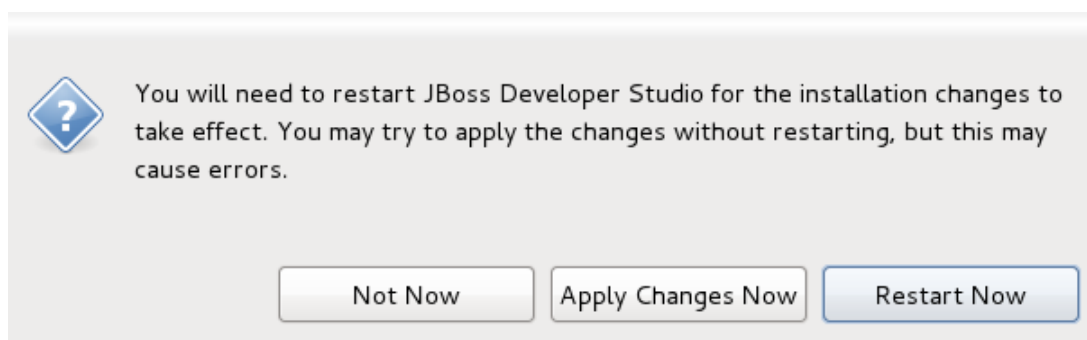


Figure 2.20.

2.3. Project Example Installation

To install an example from within JBoss Central, navigate to the **Project Examples** section and select an example from the list.

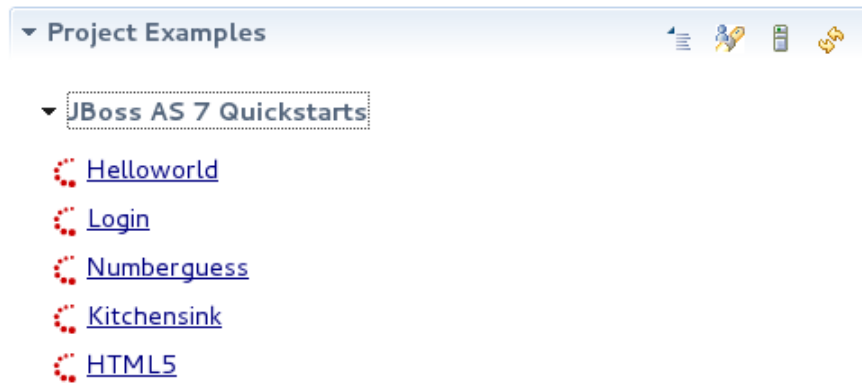


Figure 2.21. Project Examples

This section will guide you through the installation of the **Helloworld** example.

Click on the **Helloworld** item in the **Quickstarts** list.

An installation wizard for the example will open and automatically search for required components on your system. If a particular component is not found (a plug-in or a required version of the JBoss server), the component will be automatically downloaded for you during example installation.

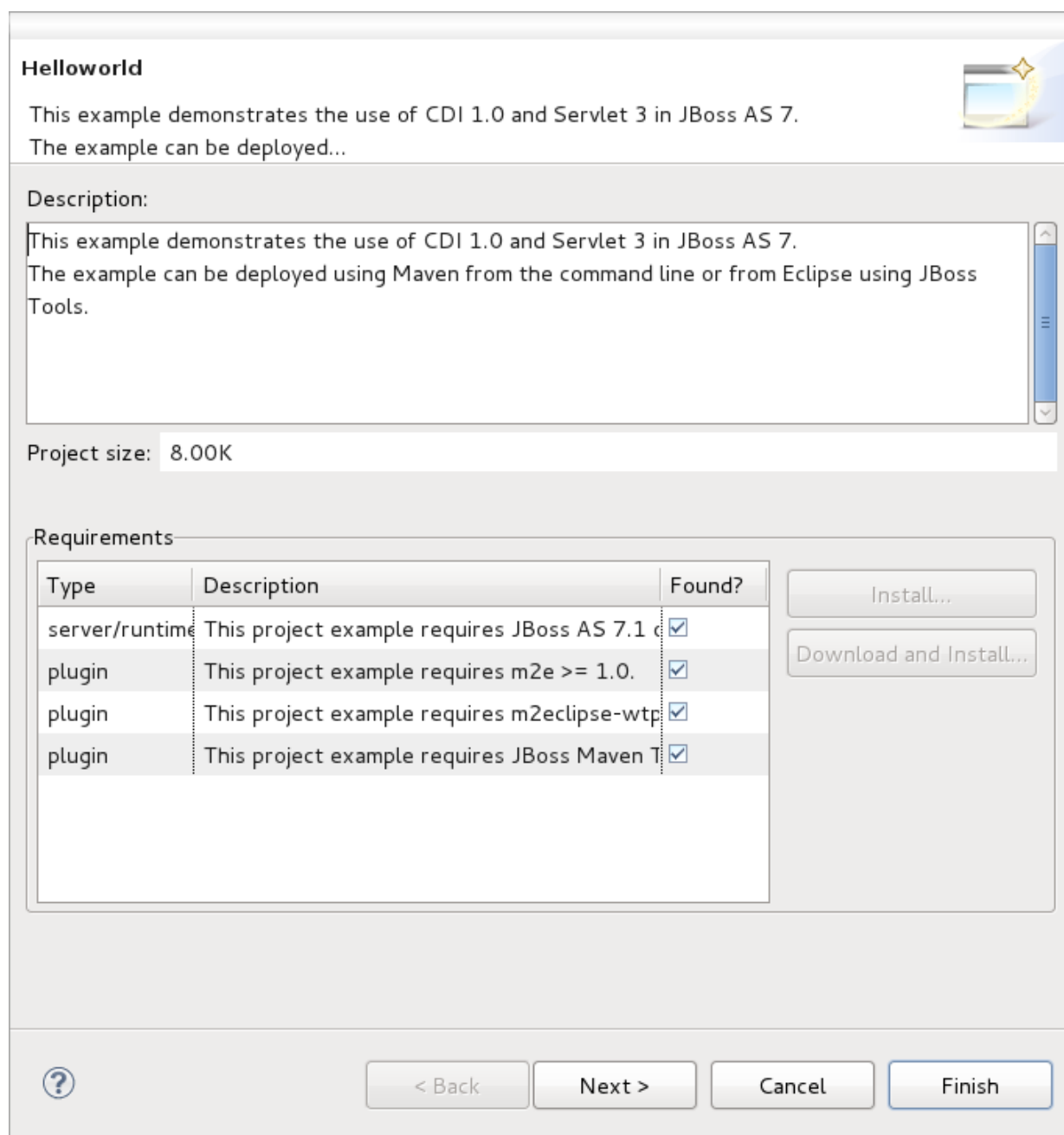
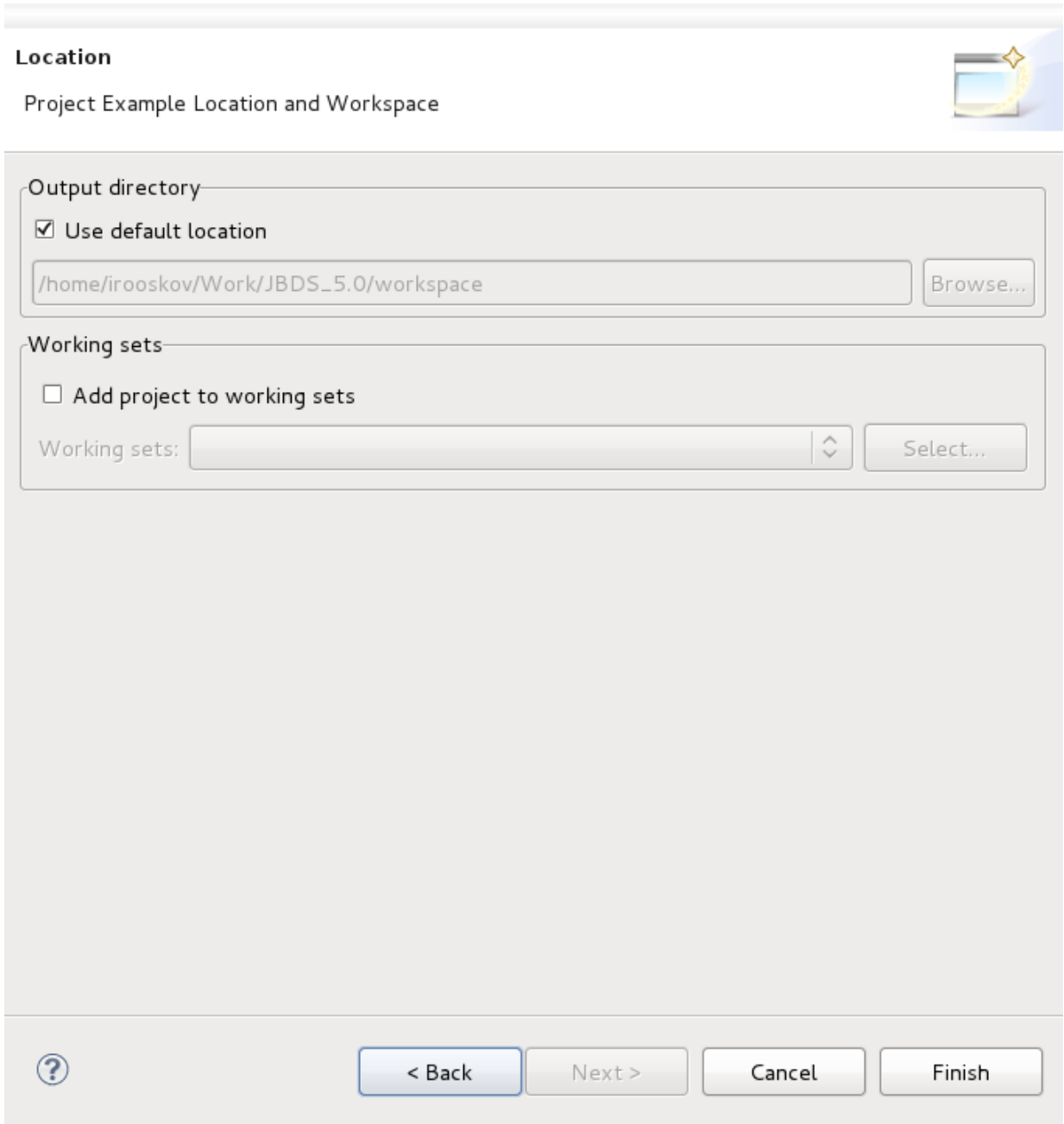


Figure 2.22. Project Examples

Click the **Next** button.

You will be asked to define the install location for the example. Your current workspace will be selected by default.



The dialog box is titled "Location" and "Project Example Location and Workspace". It features a "Location" icon in the top right corner. The "Output directory" section has a checked "Use default location" checkbox and a text field containing "/home/irooskov/Work/JBDS_5.0/workspace", with a "Browse..." button to its right. The "Working sets" section has an unchecked "Add project to working sets" checkbox and a "Working sets:" label followed by a list box and a "Select..." button. At the bottom, there is a help icon, a "< Back" button, a "Next >" button, a "Cancel" button, and a "Finish" button.

Location

Project Example Location and Workspace

Output directory

☒ Use default location

/home/irooskov/Work/JBDS_5.0/workspace Browse...

Working sets

☐ Add project to working sets

Working sets: Select...

? < Back Next > Cancel Finish

Figure 2.23. Project Examples

Click the **Finish** button to begin installation of the example.

Once the example has been installed it will appear in the list of projects within your **Project Explorer**.

JBoss Perspective

The JBoss perspective is designed to incorporate the views most often used by developers using JBoss, while also changing standard menu items to reflect what a JBoss developer would want and need.

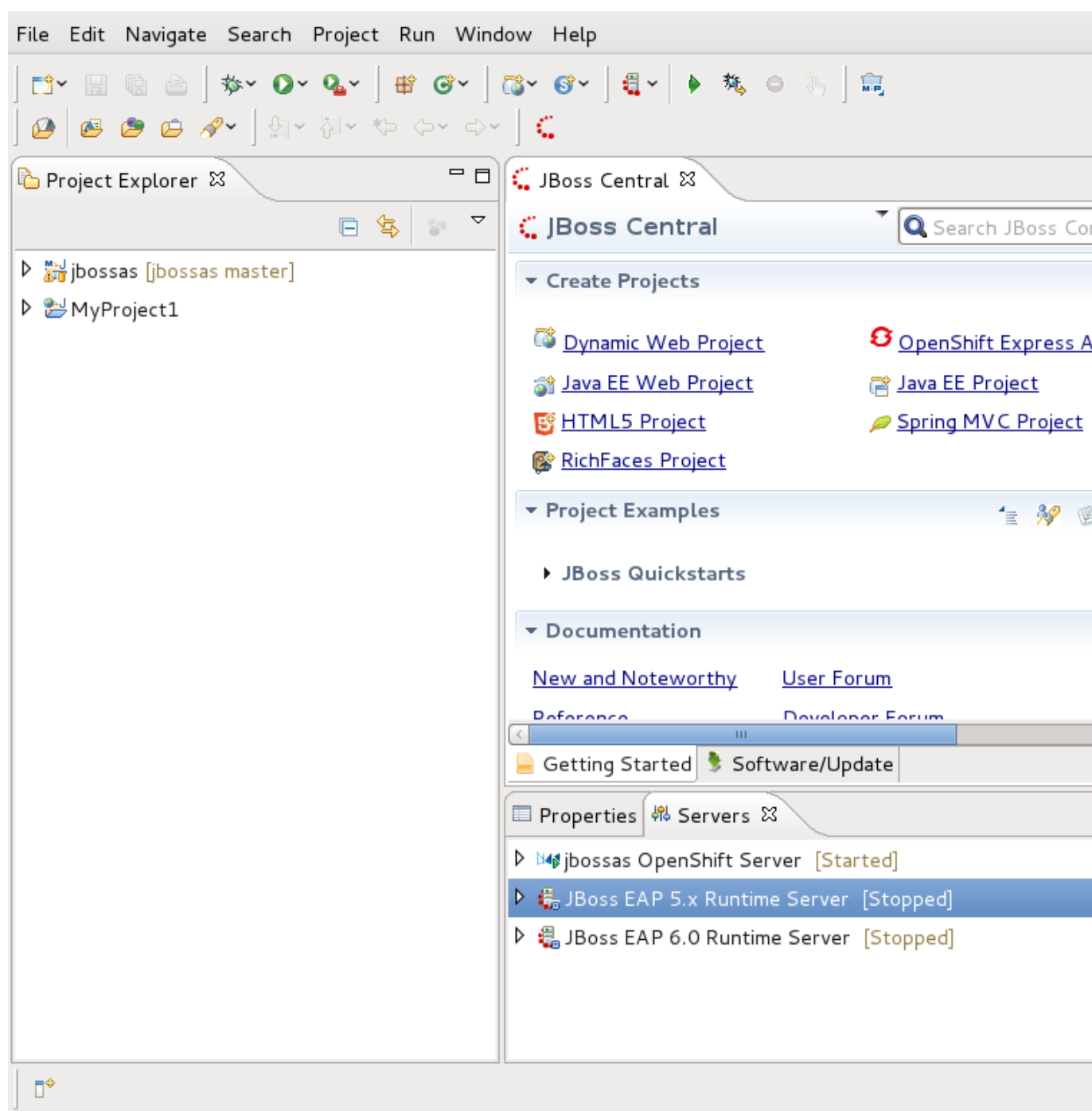


Figure 3.1. JBoss Perspective

The JBoss perspective views include: **Project Explorer**, **Outline**, **Palette**, **Properties** and **Servers**.

Certain menus also see a change in available items. These menus include **File** → **New**, **Window** → **Show View** and the context menu for a project.

A selection of the menu items for **File** → **New** can be seen in [Figure 3.2, “JBoss Perspective: FileNew”](#).

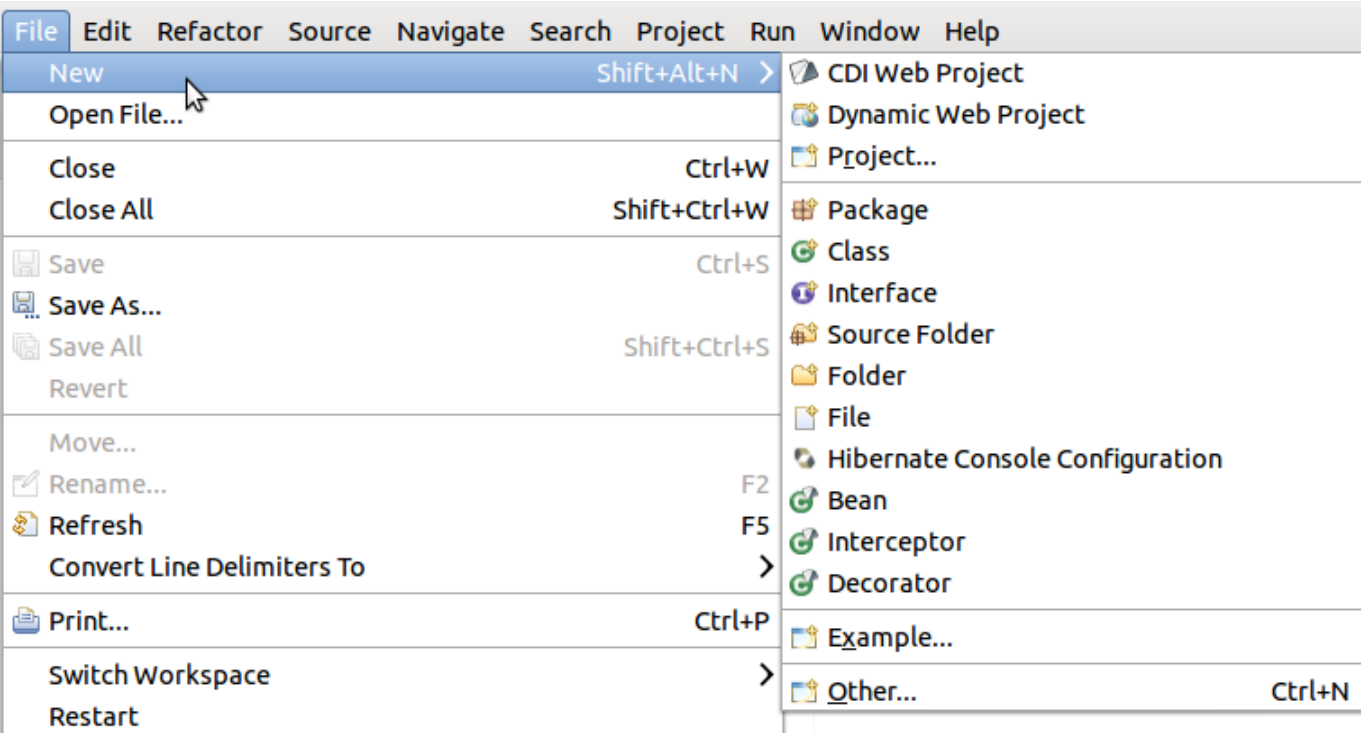


Figure 3.2. JBoss Perspective: File → New

A selection of the menu items for **Window** → **Show View** can be seen in [Figure 3.3, “JBoss Perspective: WindowShow View”](#).

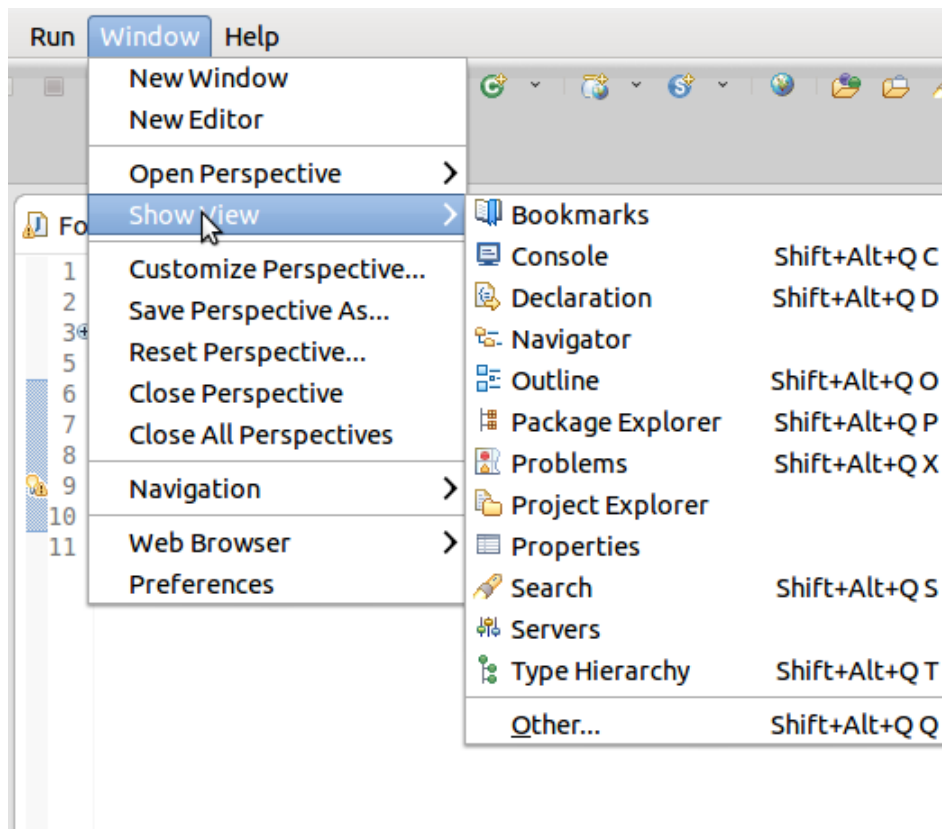


Figure 3.3. JBoss Perspective: Window → Show View

A selection of the menu items for the context menu of a project can be seen in [Figure 3.4, “JBoss Perspective: Context MenuNew”](#).

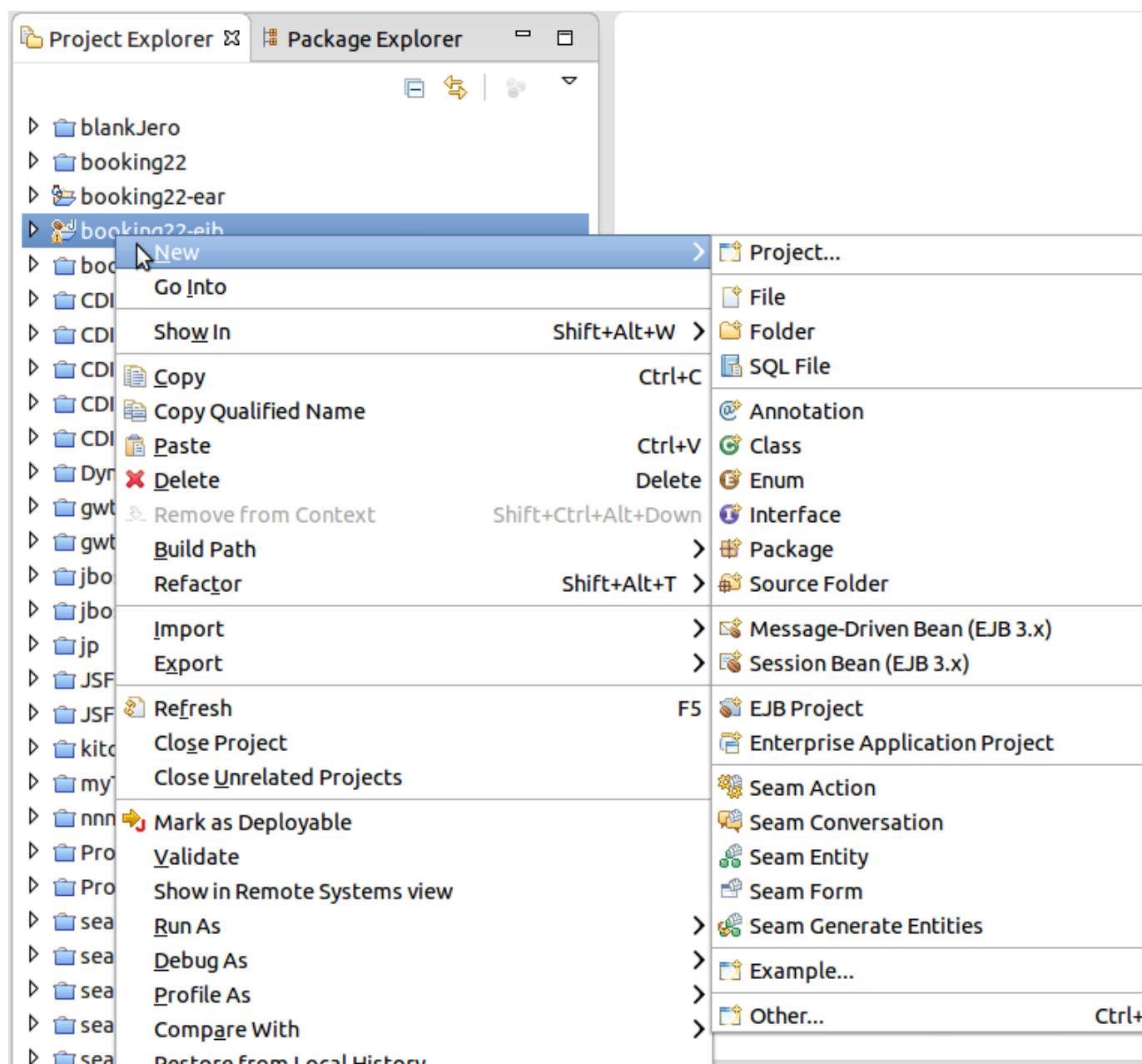


Figure 3.4. JBoss Perspective: Context Menu → New

Setting up a JBoss runtime and managing the server

Although JBoss Developer Studio works closely with JBoss Enterprise Application Platform 6 we do not ultimately tie you to any particular server for deployment. There are some servers that Studio supports directly (via the bundled Eclipse WTP plug-ins). In this chapter we discuss how to manage a JBoss server.



Note

This chapter assumes you have a JBoss application server installed on your system. If you do not, consult the installation instructions that accompanied your server.

4.1. Adding and configuring a JBoss server runtime

- Select the Servers view by navigating to **Window** → **Show View** → **Other** → **Server** → **Servers**.
- Click the **new server wizard** text in the **Servers** view or if this is not your first server, right-click anywhere in the view and select **New** → **Server**.
- Select the server option that matches your installed server.

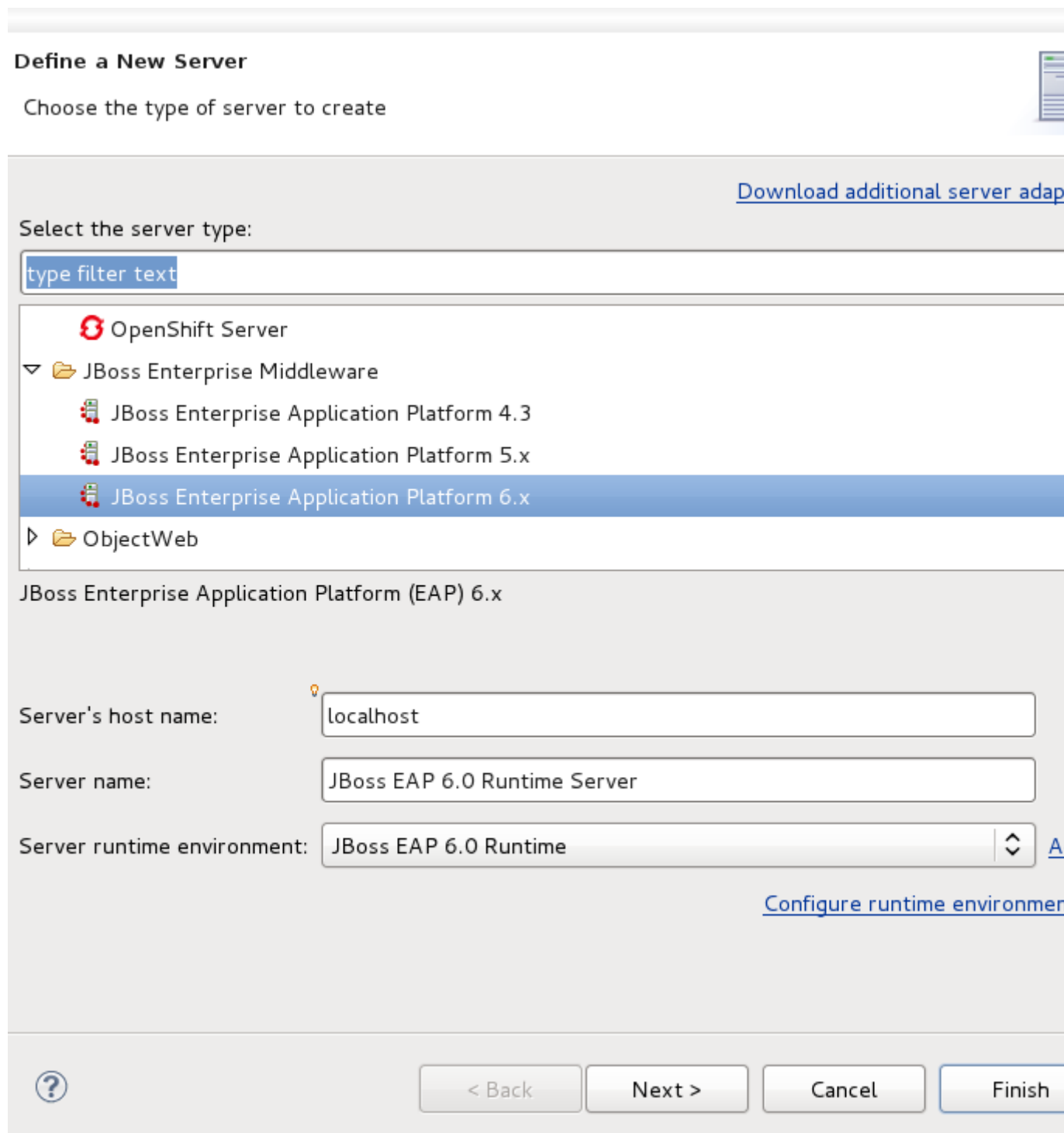


Figure 4.1. Selecting Server Name and Server Type

- To setup the new runtime, click the **Next** button.
- In the next dialog verify the specified information and set local or remote server information.

Create a new JBoss Server

JBoss Enterprise Application Platform 6.0



A JBoss Server manages starting and stopping instances of JBoss.

It manages command line arguments and keeps track of which modules have been deployed.

Runtime Information

If the runtime information below is incorrect, please press back, Installed Runtimes..., and then Add to create a new runtime from a different location.

Home Directory /home/irooskov/Work/JBT_3/Installed/Servers/jboss-eap-6.0-ER6

Execution Environment Java Platform, Standard Edition 6.0

JRE Default JRE for JavaSE-1.6

Server Behaviour

- ☐ Server is externally managed. Assume server is started.
- ☐ Listen on all interfaces to allow remote web connections
- ☐ Expose your management port as the server's hostname

Local



< Back

Next >

Cancel

Finish

Figure 4.2. JBoss Runtime Summary

- Lastly, a screen will appear that will allow you to modify projects that are to be configured for the server. Click the **Finish** button.

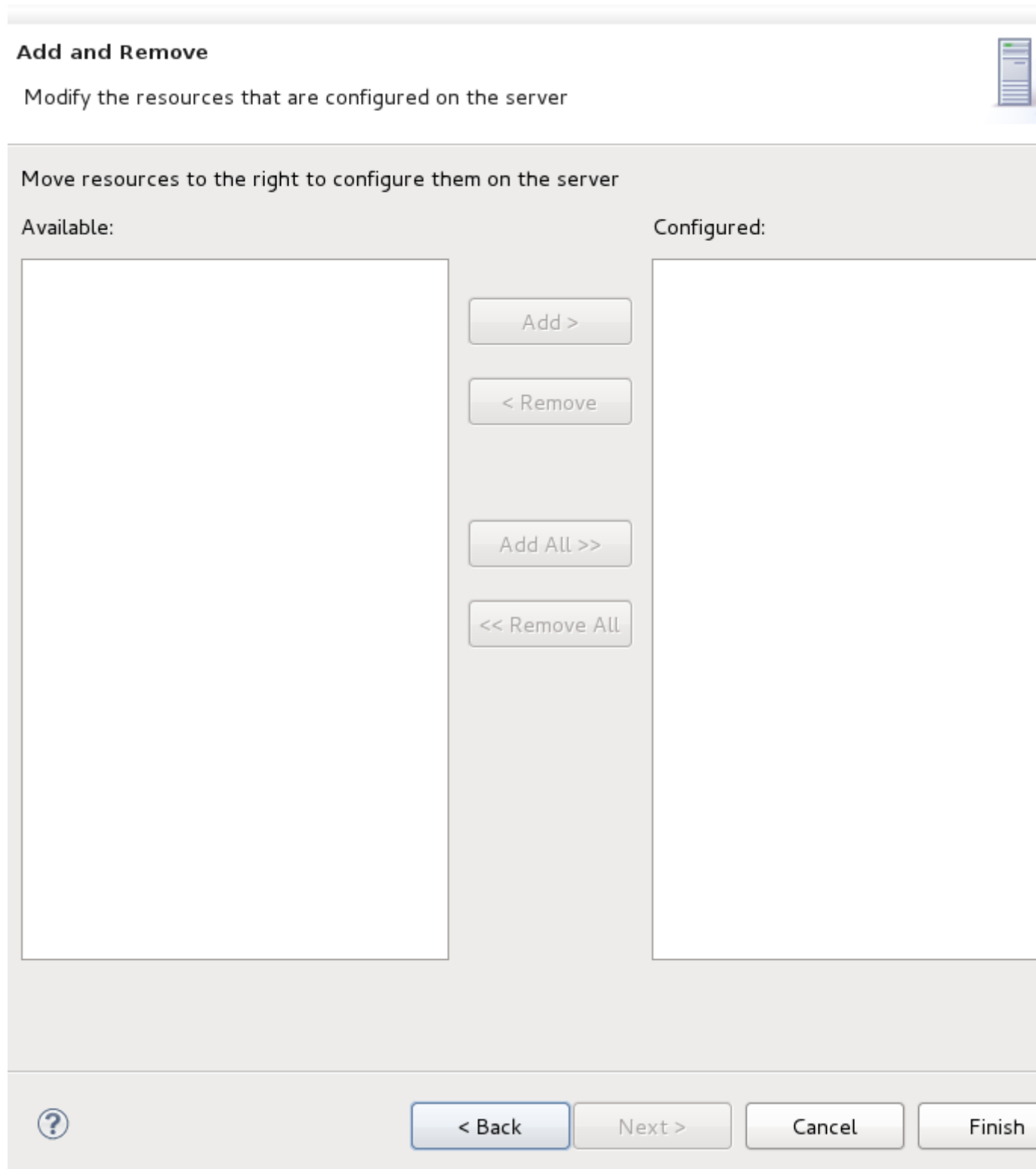


Figure 4.3. Configuring Projects

A new JBoss Server should now appear in the Servers view.

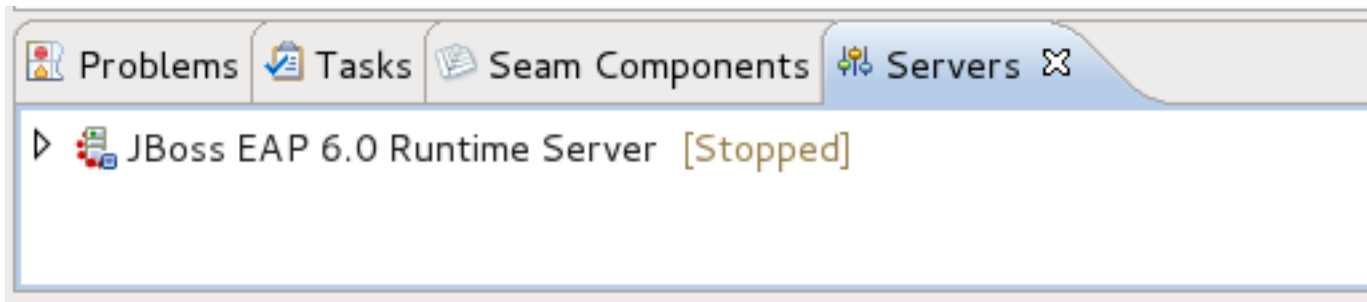


Figure 4.4. New JBoss Server

4.2. Starting JBoss Server

Starting JBoss Server is quite simple. JBoss Developer Studio allows you to control its behavior with the help of a special toolbar, where you could start it in a regular or debug mode, stop it or restart it.

- To launch the server click the green-with-white-arrow icon in the Servers view or right click server name in this view and click the **Start** button. If this view is not open, select **Window** → **Show View** → **Other** → **Server** → **Servers**

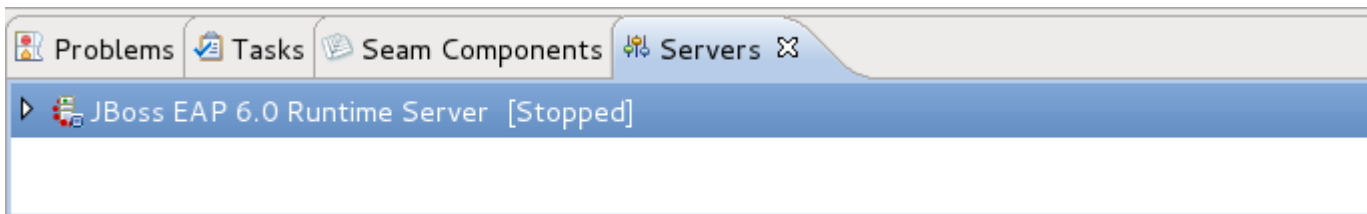


Figure 4.5. Starting from Icon

While launching, server output is written to the Console view:

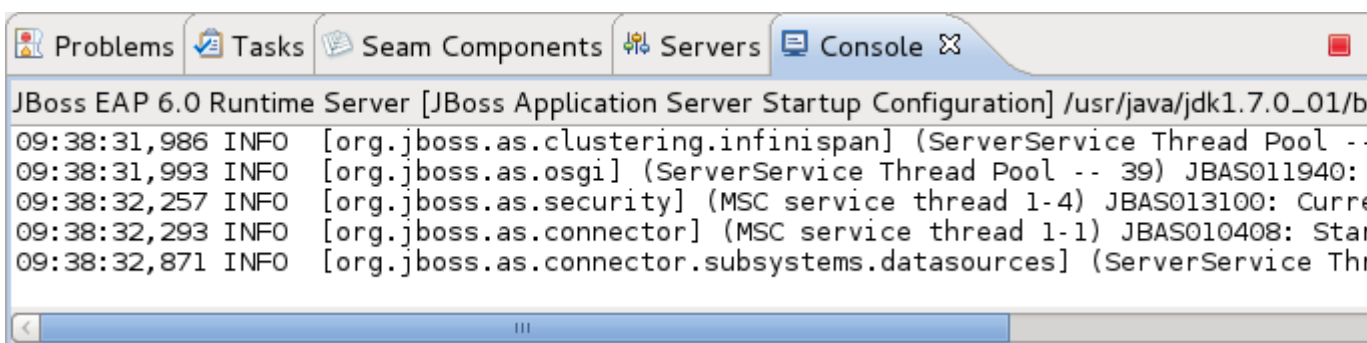


Figure 4.6. Console Output

When the server is started you should see *Started* in the square brackets right next its name in the Servers view.

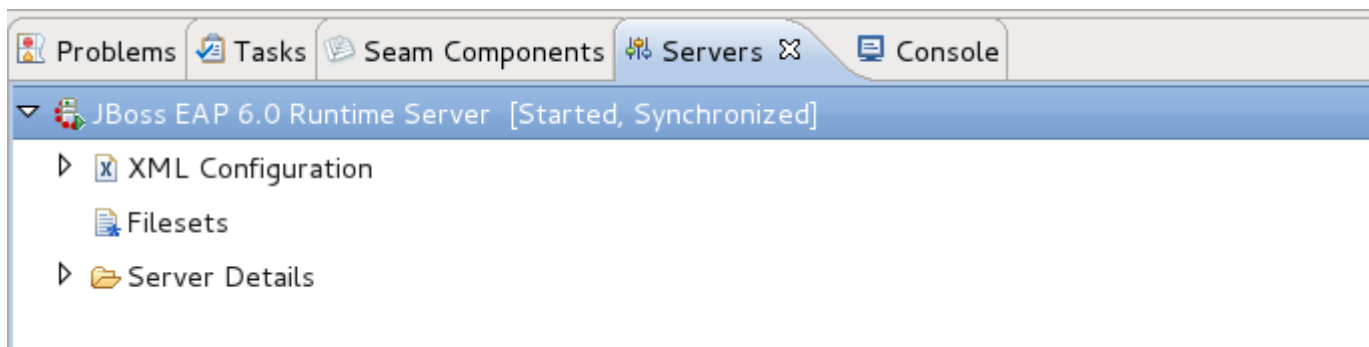


Figure 4.7. Server is Started

4.3. Stopping the JBoss Server

To stop the server, click the **Stop** button icon in Servers or right click the server name and press **Stop**.

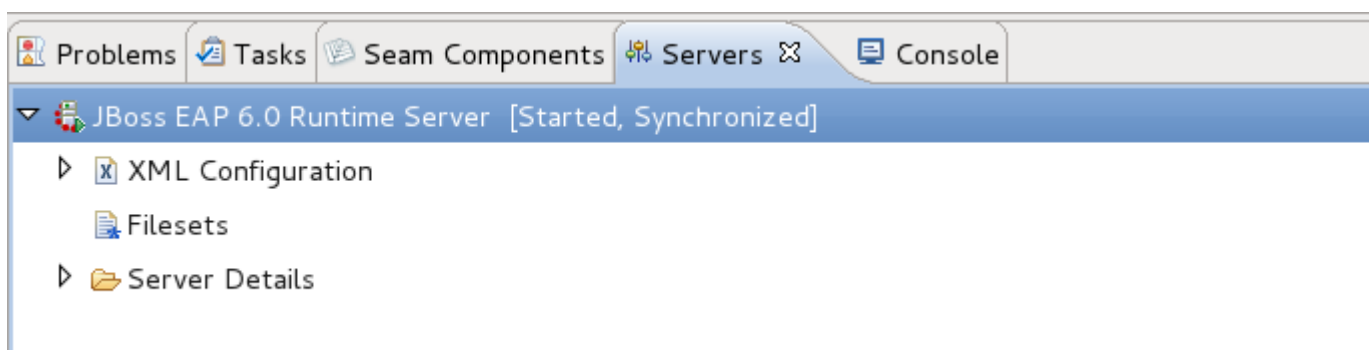
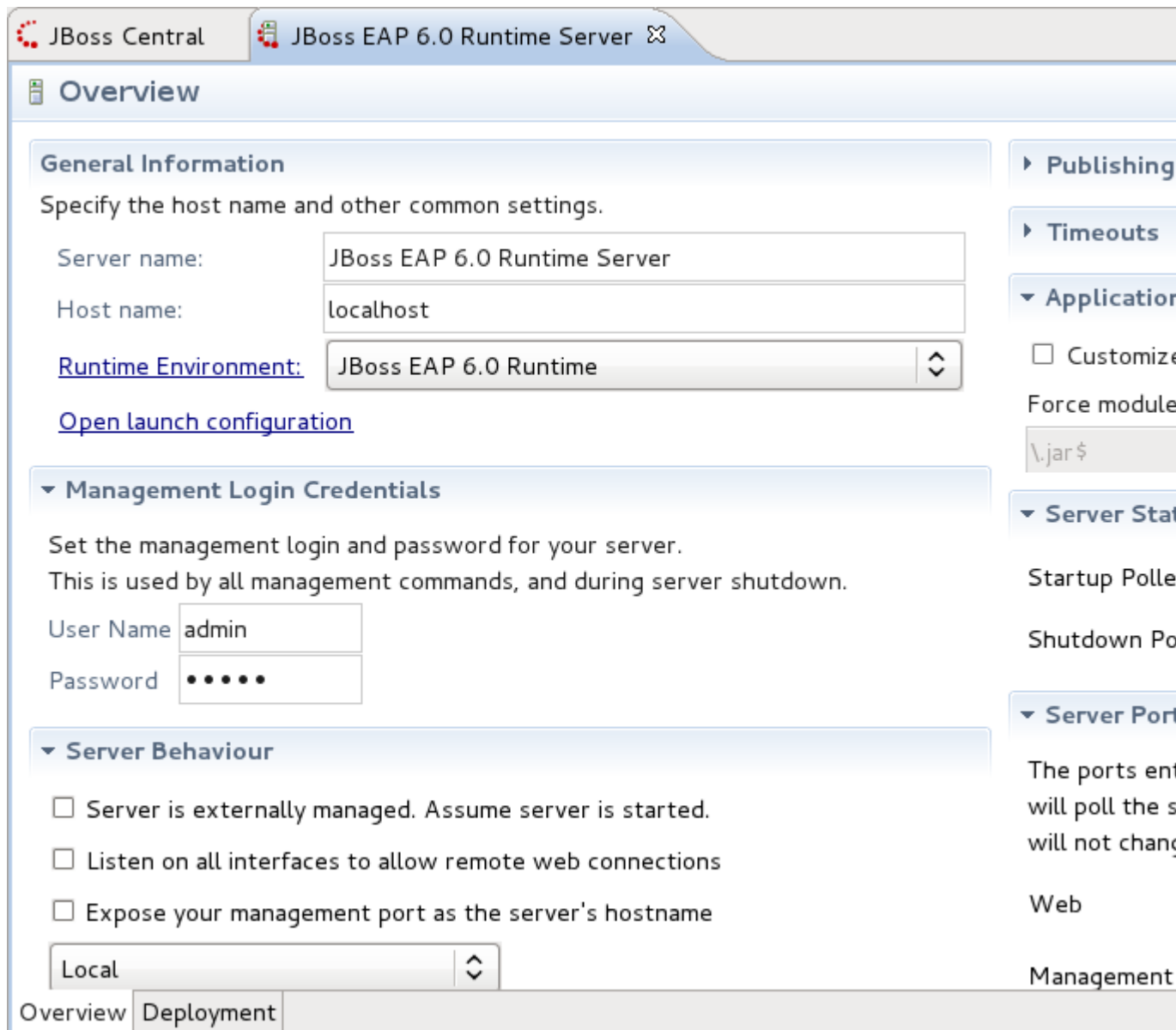


Figure 4.8. Stopping Server

When the server is stopped you will see *Stopped* in the square brackets next to its name.

4.4. Server Container Preferences

You can control how JBoss Developer Studio interacts with server containers in the Server editor. Double-click the server to open it in the editor.



The image shows the 'Overview' tab of the JBoss Central configuration page for a JBoss EAP 6.0 Runtime Server. The page is divided into several sections: 'General Information', 'Management Login Credentials', and 'Server Behaviour'. The 'General Information' section includes fields for 'Server name' (JBoss EAP 6.0 Runtime Server), 'Host name' (localhost), and 'Runtime Environment' (JBoss EAP 6.0 Runtime). There is a link to 'Open launch configuration'. The 'Management Login Credentials' section includes fields for 'User Name' (admin) and 'Password' (masked with dots). The 'Server Behaviour' section includes three checkboxes: 'Server is externally managed. Assume server is started.', 'Listen on all interfaces to allow remote web connections', and 'Expose your management port as the server's hostname'. Below these checkboxes is a dropdown menu set to 'Local'. On the right side, there are expandable sections for 'Publishing', 'Timeouts', 'Application' (with a 'Customize' checkbox and 'Force module' text), 'Server Status' (with 'Startup Poller' and 'Shutdown Poller' text), and 'Server Ports' (with text about ports and a 'Web Management' link).

JBoss Central JBoss EAP 6.0 Runtime Server

Overview

General Information

Specify the host name and other common settings.

Server name: JBoss EAP 6.0 Runtime Server

Host name: localhost

Runtime Environment: JBoss EAP 6.0 Runtime

[Open launch configuration](#)

Management Login Credentials

Set the management login and password for your server.
This is used by all management commands, and during server shutdown.

User Name: admin

Password: •••••

Server Behaviour

☐ Server is externally managed. Assume server is started.

☐ Listen on all interfaces to allow remote web connections

☐ Expose your management port as the server's hostname

Local

Overview Deployment

Publishing

Timeouts

Application

☐ Customize

Force module

\.jar\$

Server Status

Startup Poller

Shutdown Poller

Server Ports

The ports entered will poll the server and will not change.

Web Management

Figure 4.9. Server Overview

Here you can specify some common settings: host name, server name, runtime as well as settings related to publishing, timeouts and server ports.

Developing a simple JSP web application

In this chapter you'll find out how to create a simple JSP application using JBoss Developer Studio. The application will show a classic "Hello World!" on the page.

We'll assume that you have already launched JBoss Developer Studio and also that the Web Development perspective is the current perspective. If not, make it active by selecting **Window** → **Open Perspective** → **Web Development** from the menu bar or by selecting **Window** → **Open Perspective** → **Other...** from the menu bar and then selecting Web Development from the Select Perspective dialog box.

5.1. Setting Up the Project

We are going to start by creating a Dynamic Web Project with a minimal structure, i.e. with just required facets. Thus this section will perform you all necessary steps on how to do this.

- Go to the menu bar and select **File** → **New** → **Other...**
- Select **Web** → **Dynamic Web Project** in the New Project dialog box
- Click the **Next** button
- Enter "jspHello" as a project name
- Then select *Minimal Configuration* from the list of possible configurations and click the **Finish** button.

New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location

☒ Use default location

Location:

Target runtime

Dynamic web module version

Configuration

The most conservative starting point. Only the required facets are installed. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name:

Working sets

☐ Add project to working sets

Working sets:

Figure 5.1. Create New Web Project

The *jspHello* node should appear in the upper-left Package Explorer view.

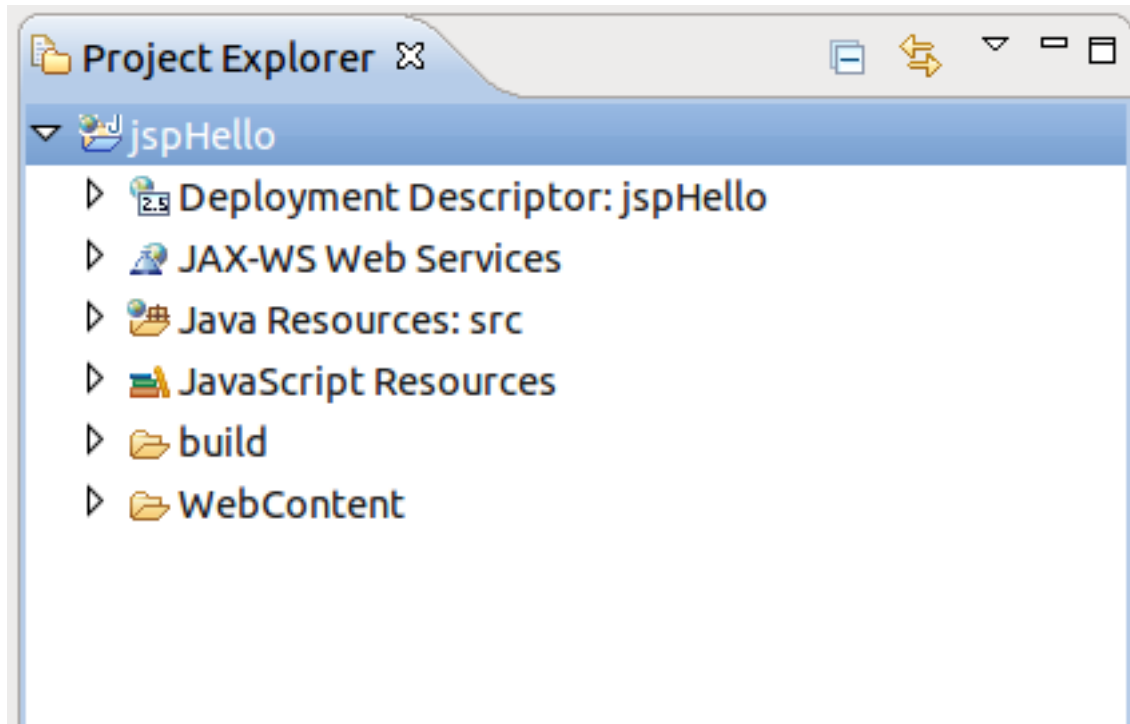


Figure 5.2. New Web Project

5.2. Creating JSP Page

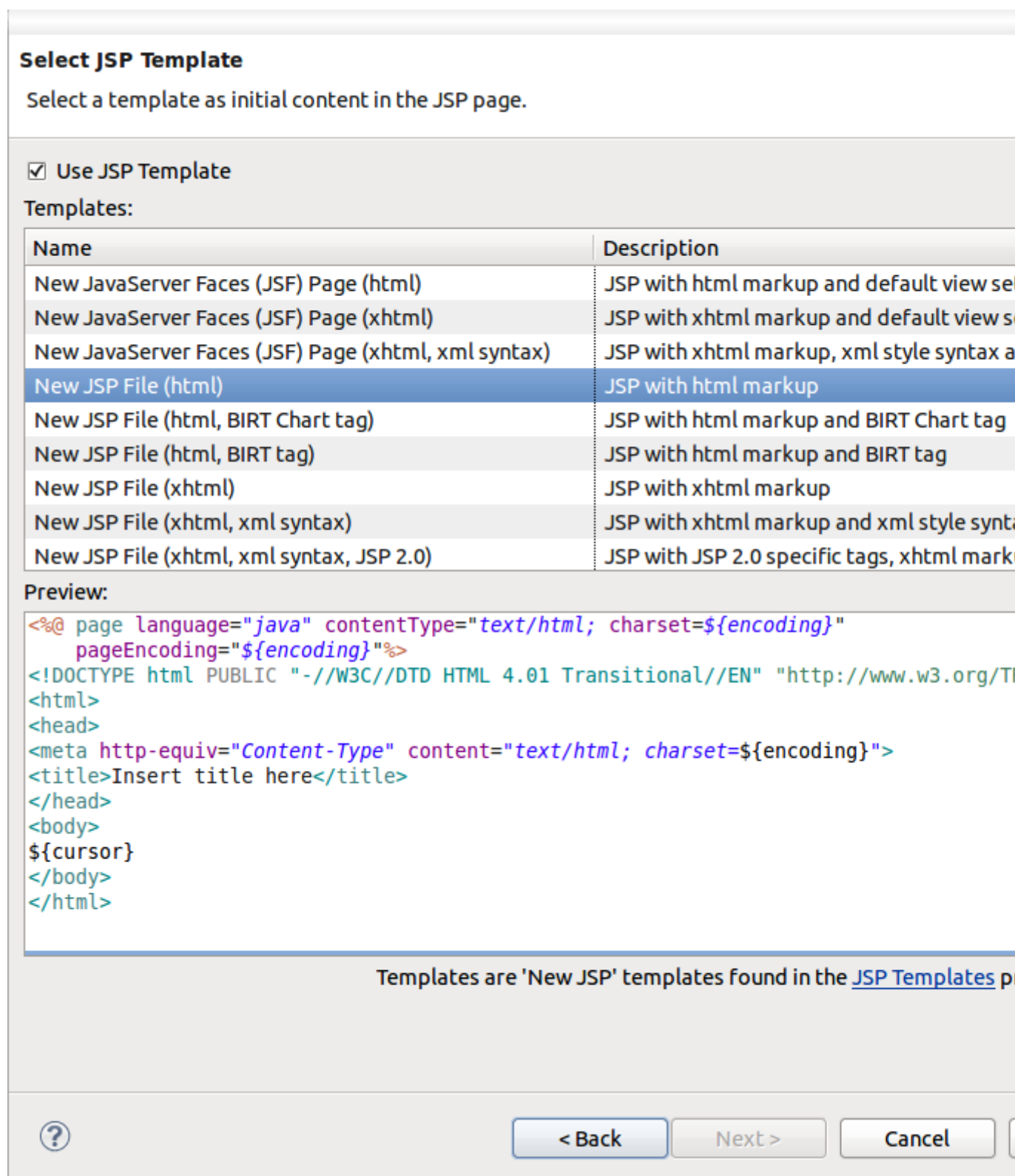
This section covers all the points how to create, edit and then preview JSP page.

In our simple application we need to create only one JSP page which displays a *"Hello World!"* message.

- Right click the webContent folder and select **New** → **JSP**.
- Type `hello.jsp` for a file name and click the **Next** button.

In the next window you can choose a template for your JSP page and see its preview.

- Select *New JSP File (xhtml)* template and click the **Finish** button.

**Figure 5.3. Create JSP Page**

Our `hello.jsp` page will now appear in the Project Explorer view.

5.2.1. Editing a JSP Page

Let's now make a little change so that a JSP page displays *"Hello World!"* message.

- Insert this line inside the `<body>` `</body>` tag:

```
<% System.out.println("Hello World!"); %>
```

Notice that content assist functionality is always available when you are typing:

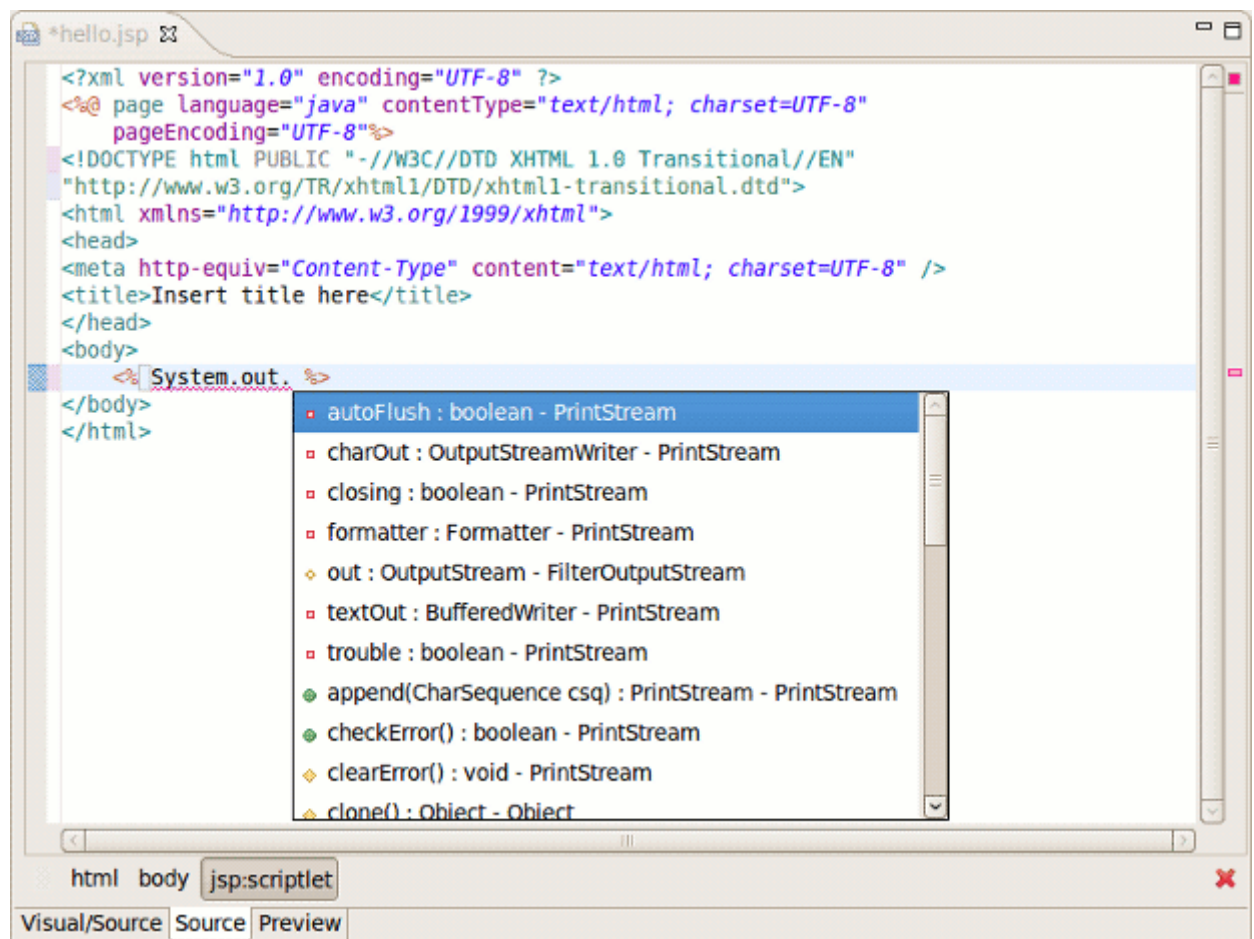


Figure 5.4. Content Assist in JSP Page

After changes made your `hello.jsp` page should look like this:

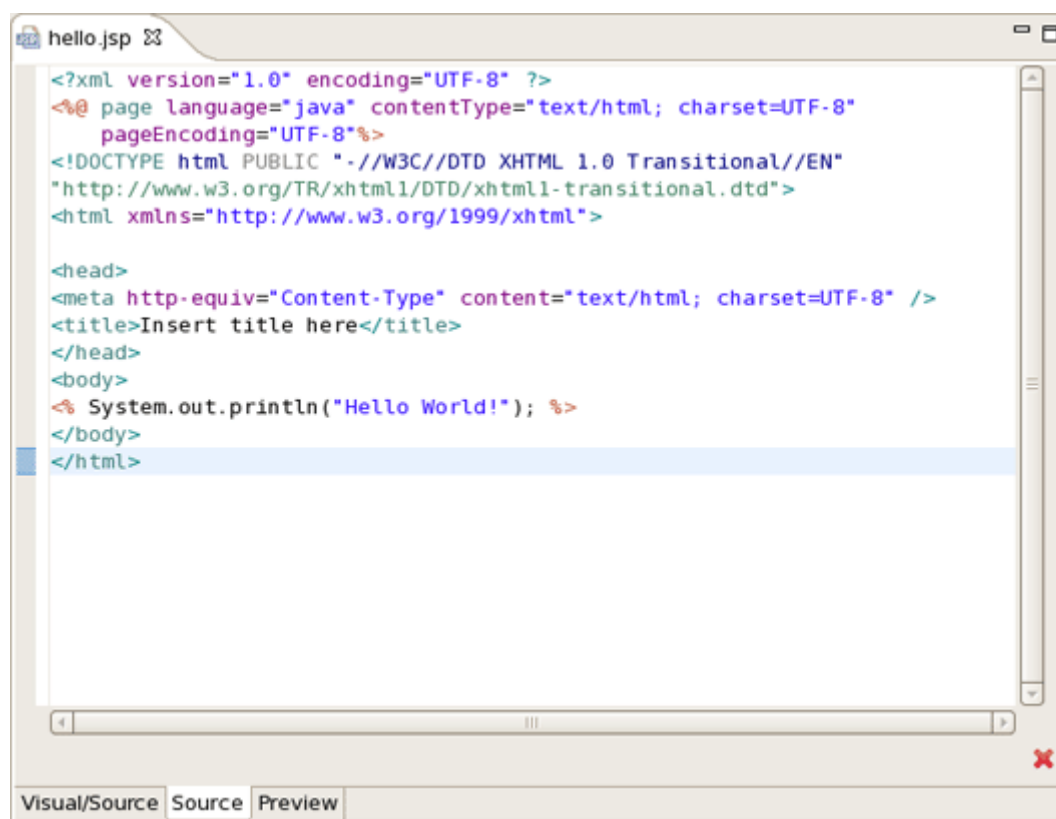


Figure 5.5. Hello.jsp Page Source

This line will actually output *"Hello World!"* message in the Console. To make the message displayed in the Browser, just replace this line with the simple *Hello World!*.

5.2.2. web.xml file

When you are creating web project the wizard creates the `web.xml` file for you automatically. The `web.xml` file editor provided by JBoss Developer Studio is available in two modes: Tree and Source.

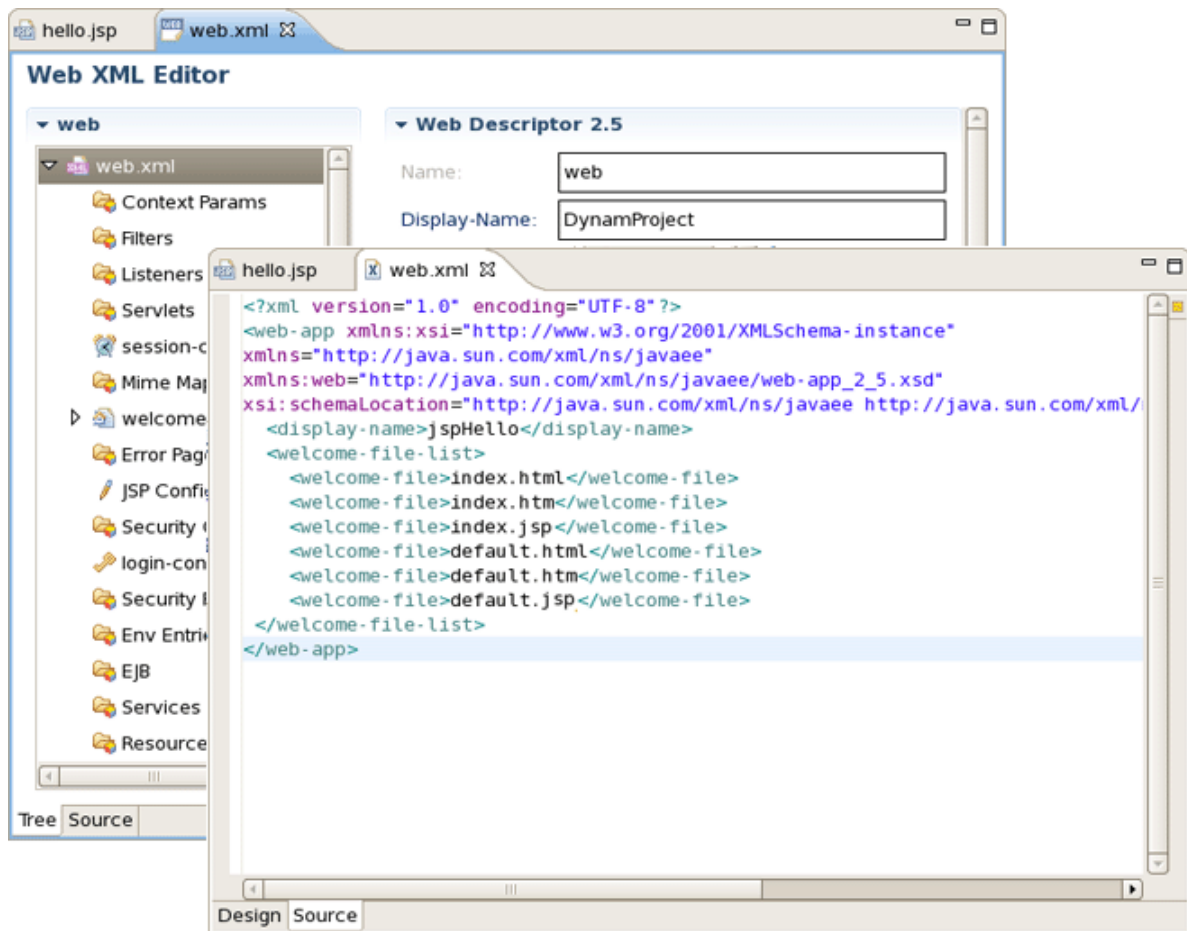


Figure 5.6. Web.xml in Design and Source Mode

Both modes are fully synchronized. Let's add a mapping to our `hello.jsp` page in the `web.xml` file.

- Switch to the Source tab.
- Add the next code into `<welcome-file-list>` :

```
<welcome-file>hello.jsp</welcome-file>
```

If you go back to Tree tab you will see that the changes made in the Source tab are automatically reflected.

Actually you do not really need to do any configurations right now.

5.2.3. Deploying the project

Writing ant scripts and managing the packaging process can be quite a complicated and time consuming task for even the most trivial web applications. However, JBoss Developer Studio relieves you of this burden. All you need is to start your JBoss Server and launch your application in your favorite browser.

You can also create a JAR archive with JBoss Developer Studio's Archive Tools and export it to any web server.

5.2.3.1. JAR Config

Project archives managing is available through the Project Archives view.

- Select **Window** → **Show view** → **Other** → **JBoss Tools** → **Project archives** from the menu bar
- Select a project in Package Explorer you want to be archived

In the Project Archives view you will see the that the project is now listed:

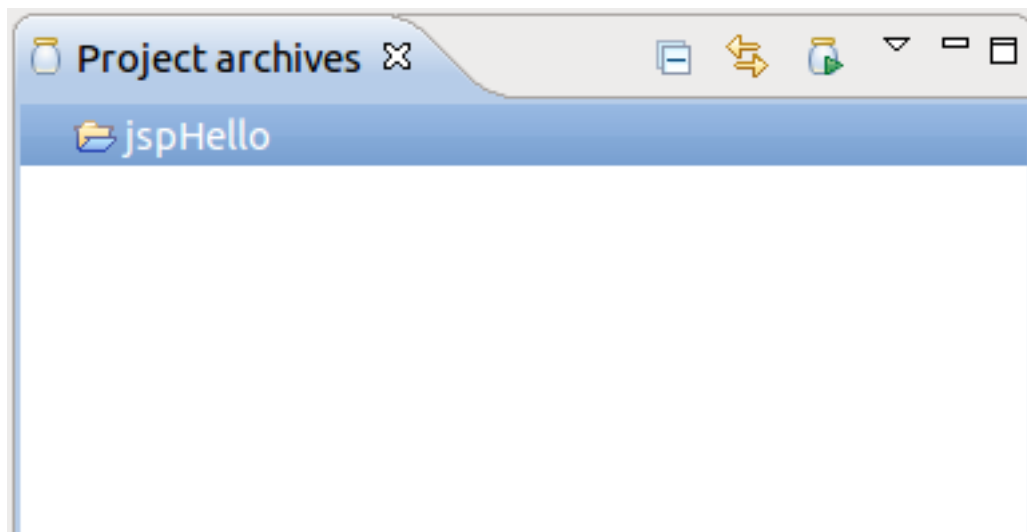


Figure 5.7. Project Archives

Right click on the project and select the JAR type of archive.

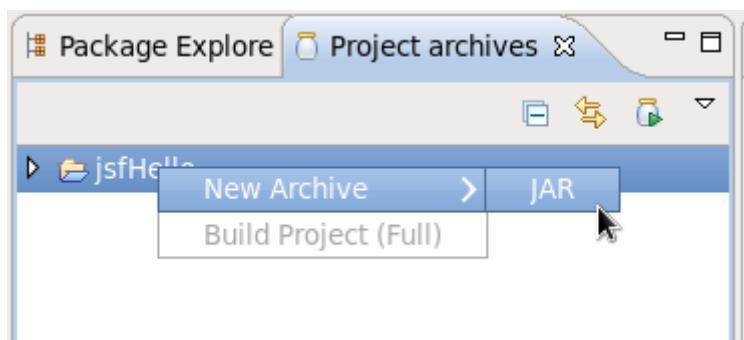


Figure 5.8. Project Archives

In the New JAR dialog you can see automatically selected default values.

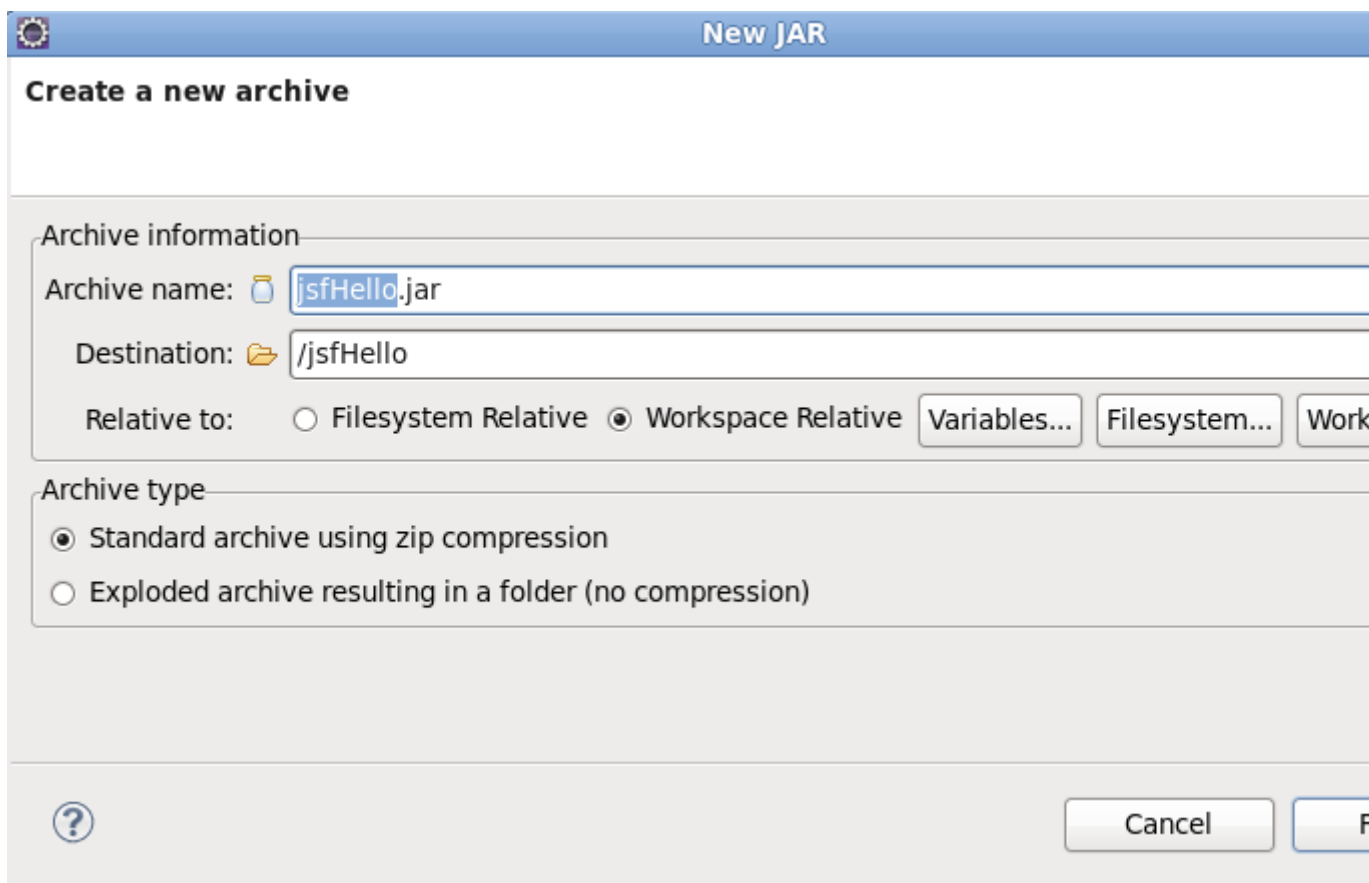


Figure 5.9. New JAR Archive

- Click the **Finish** button. The **.JAR** file will appear in Package Explorer and also in Project Archives view as structure tree:

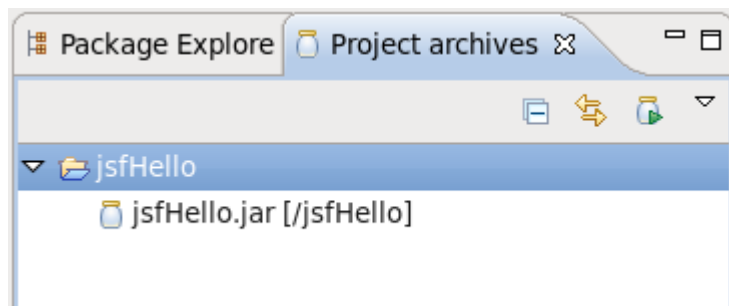


Figure 5.10. Archive is Created

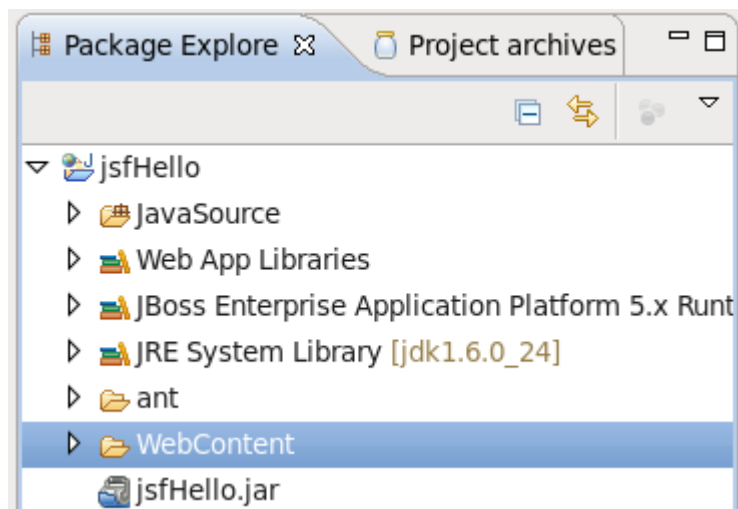


Figure 5.11. Archive in Package Explorer

Using the Project Archives view you can rebuild the archive:

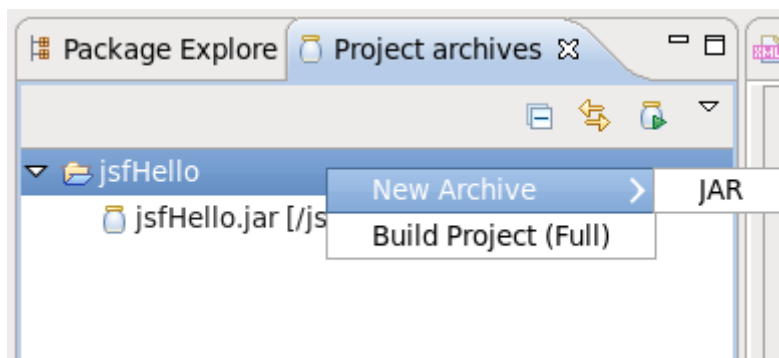


Figure 5.12. Configure Archive

5.2.3.2. Auto redeploy

When you create a web application and register it on the JBoss Server as it is automatically deployed into the `/deploy` directory of the server. JBoss Developer Studio's auto-redeploy feature

ensures you do not need to restart the server. Any changes made in the application in exploded format will trigger a redeployment on the server.

You can also use the "Finger touch" button for a quick restart of the project without restarting the server:



Figure 5.13. Finger Touch button

The "Finger" touches descriptors dependent on project (i.e. web.xml for WAR, application.xml for EAR, jboss-esb.xml in ESB projects).

5.2.4. JSP Page Preview

JBoss Developer Studio comes with JSP design-time preview features. When designing JSP pages you can easily preview how they will look during runtime. You can even attach your stylesheet to the Preview.

- Make a little change to hello.jsp page, e.g. put this code snippet:

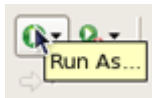
```
<%= new java.util.Date() %>
```

- Click the **Save** button.
- Switch to Preview page by clicking the Preview tab at the bottom of the page. You will see how the page will look at runtime.

5.2.5. Launch JSP Project

Now launch the project onto a JBoss server:

- Start a JBoss Server from the Servers view by clicking the Start the server icon ().
- Click the **Run** icon or right click your project folder and select **Run As** → **Run on Server**. If you haven't made any changes in the web.xml file or cleared it out you can launch the application by right clicking the hello.jsp page and selecting **Run on the Server**(



).

You should see the next page in a Browser :



Figure 5.14. Running Project

You have learnt how to organize a Dynamic Web Project with a minimal configuration, add new elements to it (in our case it is a JSP page), deploy, and run it on a JBoss Server from within JBoss Developer Studio.

Rapid Application Development of a JSF application

In this chapter you will learn how to create a JSF application being based on the Rapid Application Development (RAD) philosophy. We will create the familiar Guess Number application. The game is played according to the following rules. You are asked to guess a number between 0 and 100. If the guess is correct, a success page is displayed with a link to play again. If the guess is incorrect, a message is printed notifying that a smaller or a larger number should be entered and the game continues.

You will now learn how to create such an application from scratch, along the way demonstrating the powerful features included in JBoss Developer Studio such as project templating, Visual Page Editor, code completion and others. You will design the JSF application and then run the application from inside JBoss Developer Studio using a JBoss server.

6.1. Setting up the project

First, you should create a JSF 1.2 project using an integrated JBoss Developer Studio's new project wizard and predefined templates. Follow the next steps:

- In the Web Projects view (if it is not open select **Window** → **Show View** → **Others** → **JBoss Tools Web** → **Web Projects**) click **Create New JSF Project** button.

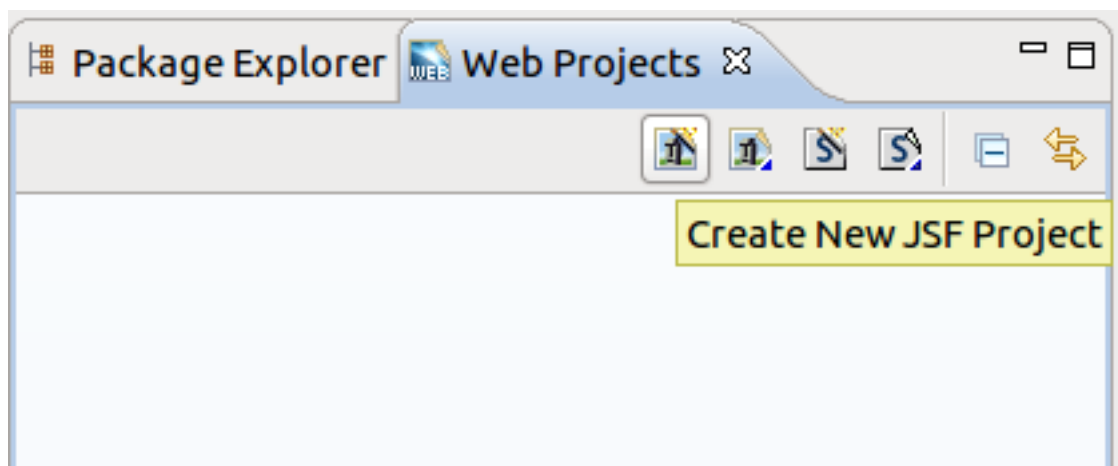


Figure 6.1. Create New JSF Project

- Enter GuessNumber as a project name, in JSF Environment drop down list choose JSF 1.2
- Leave everything else as it is and click the **Finish** button

Our project will appear in the Project Explorer and Web Projects views. As you can see JBoss Developer Studio has created the entire skeleton for the project with all required libraries, `faces-config.xml` file and `web.xml` file.

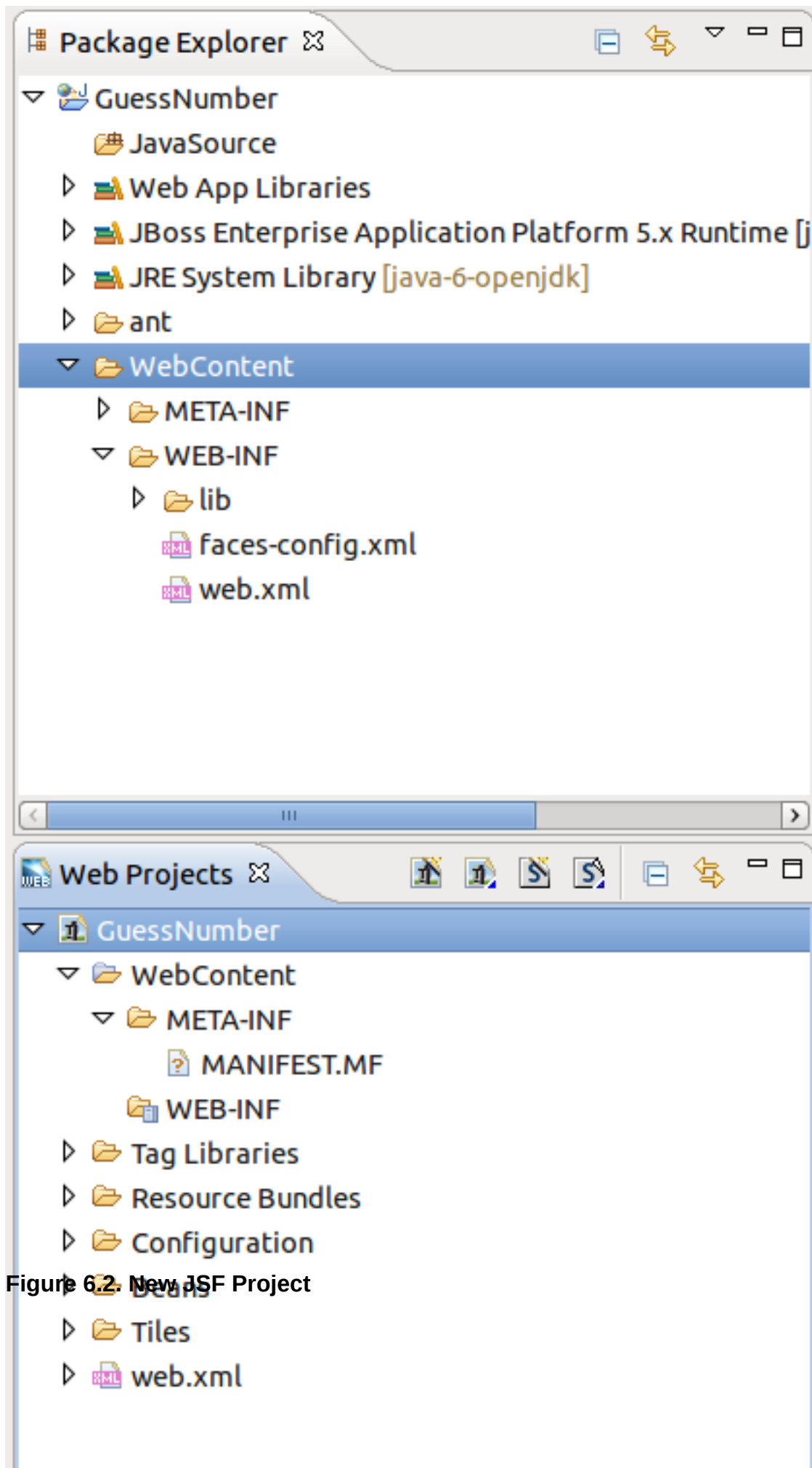


Figure 6.2. New JSF Project

As the project has been set up, new JSP pages should now be created.

6.2. Creating JSP Pages

Here, we are going to add two pages to our application. The first page is called `inputnumber.jsp`. It prompts you to enter a number. If the guess is incorrect, the same page will be redisplayed with a message indicating whether a smaller or a larger number should be tried. The second page is called `success.jsp`. This page will be shown after you guess the number correctly. From this page you also have the option to play the game again.

Steps for adding two pages to your application:

- First a folder called `pages` needs to be created under the `webContent` folder. To do this right click on the `webContent` folder in the Package Explorer view and select **New** → **Folder**. Set the **Folder Name** to `pages` and click the **Finish** button.

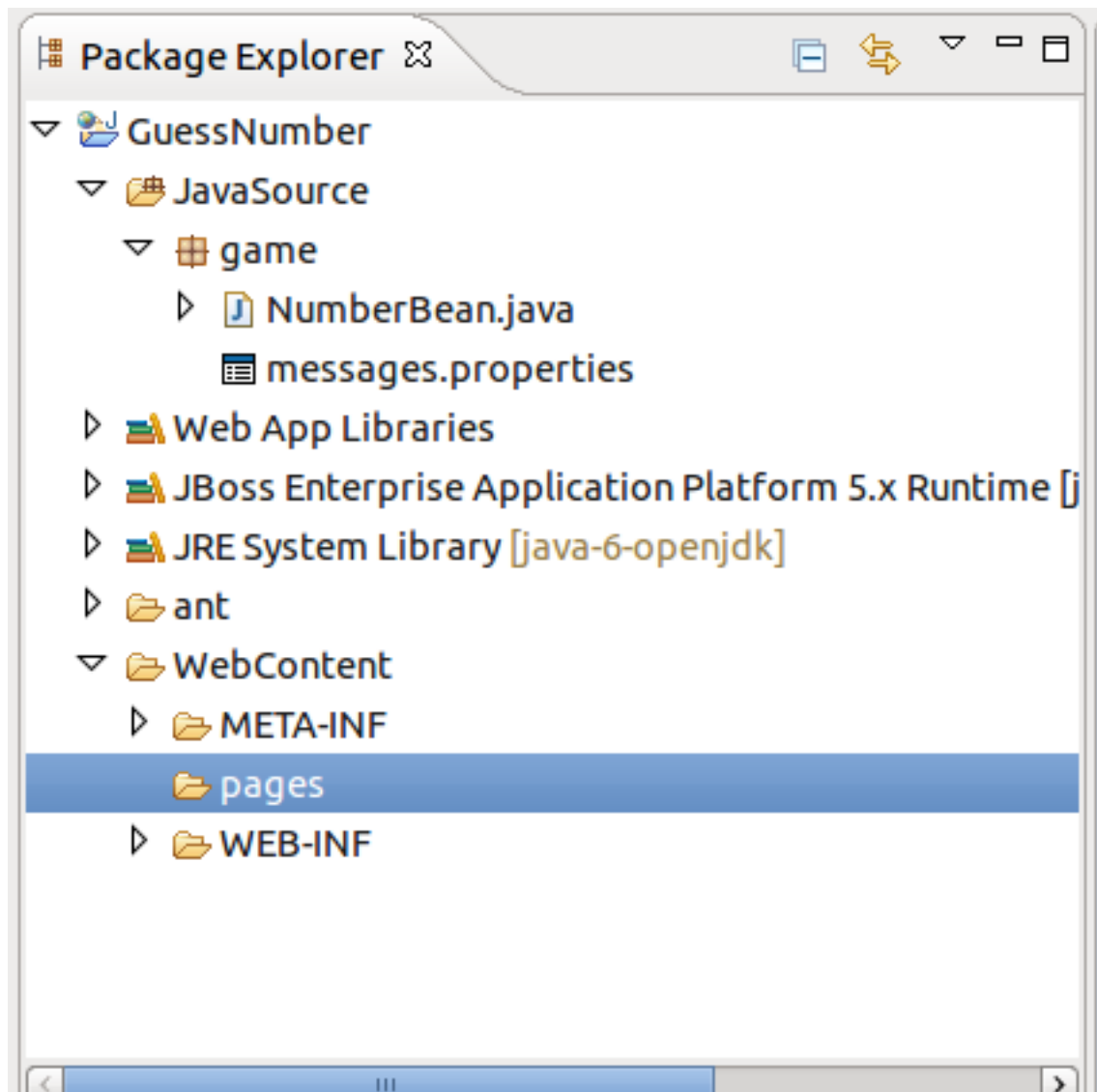


Figure 6.3. Create pages folder

- Open the `faces-config.xml` file.
- Right click anywhere on the diagram mode
- From the context menu select **New View**

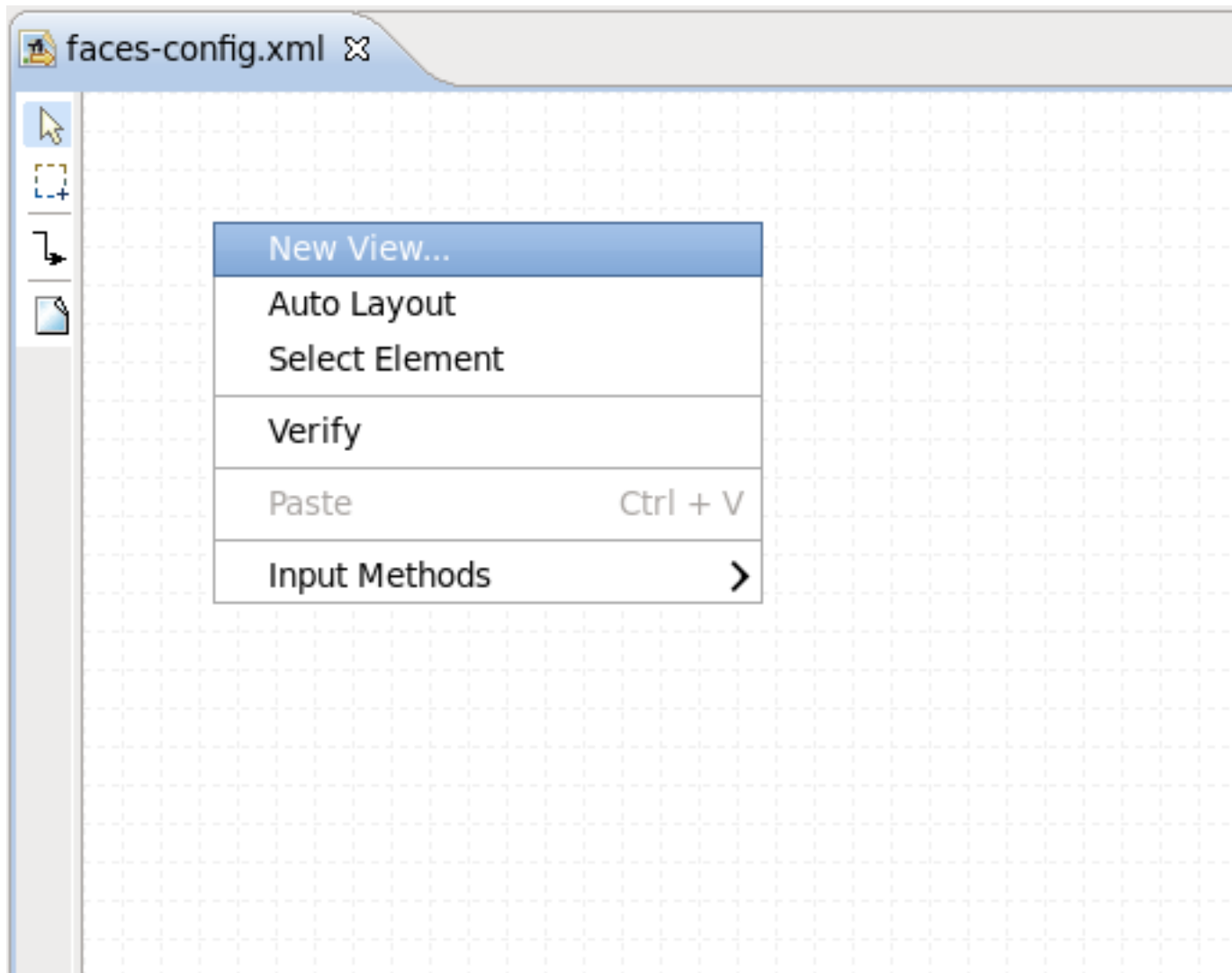


Figure 6.4. Create New View

- Type *pages/inputnumber* as the value for the *From View ID* field
- Leave everything else as is and click the **Finish** button
- In the same way create another JSF view. Type *pages/success* as the value for *From View ID*
- Select **File** → **Save**

On the diagram you will see two created views.

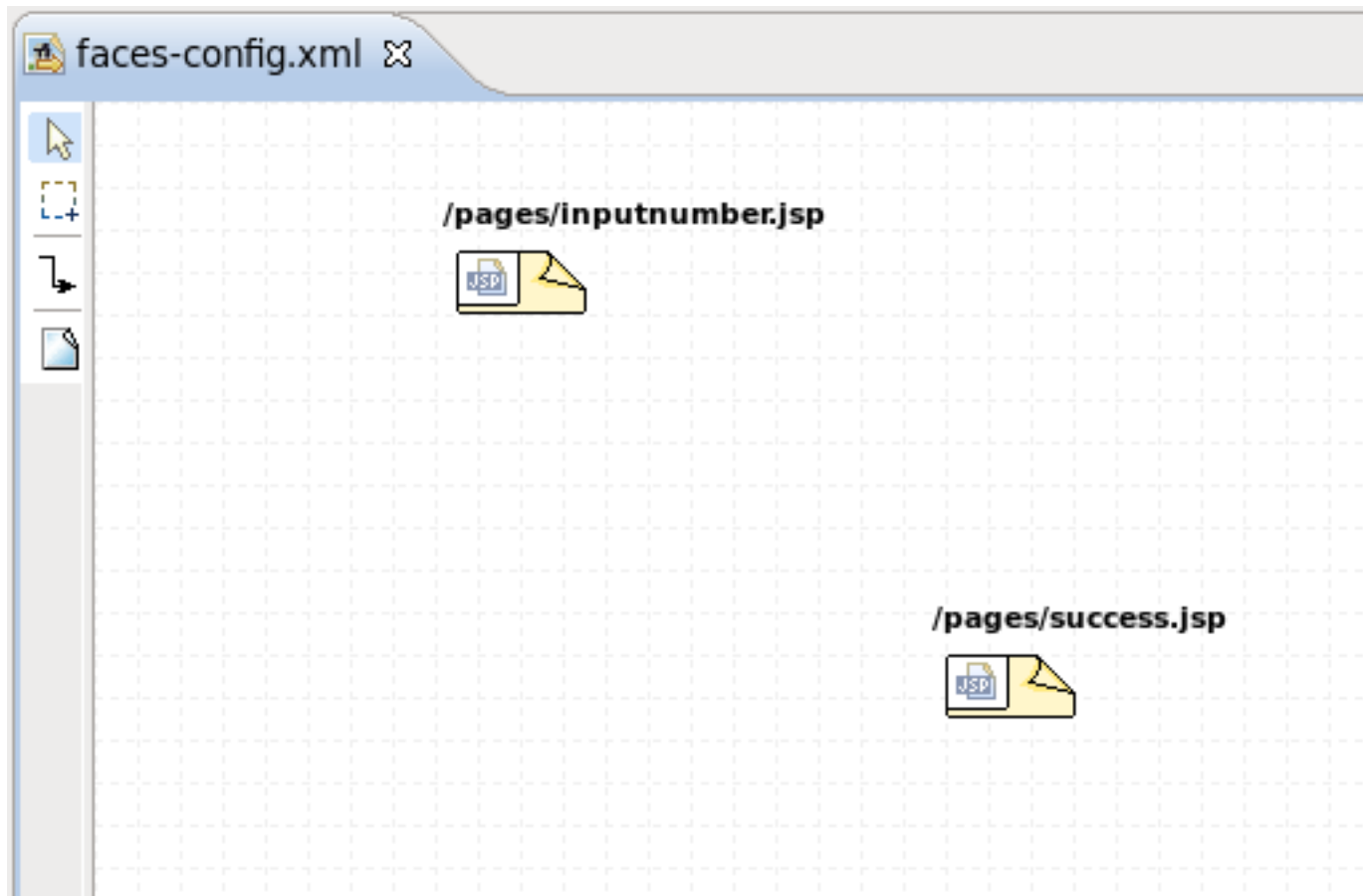


Figure 6.5. New Views

6.3. Creating Transition between two views

Then, we should create connection between JSP pages.

- In the diagram, select the **Create New Connection** icon third from the top along the upper left side of the diagram to get an arrow cursor with a two-pronged plug at the arrow's bottom

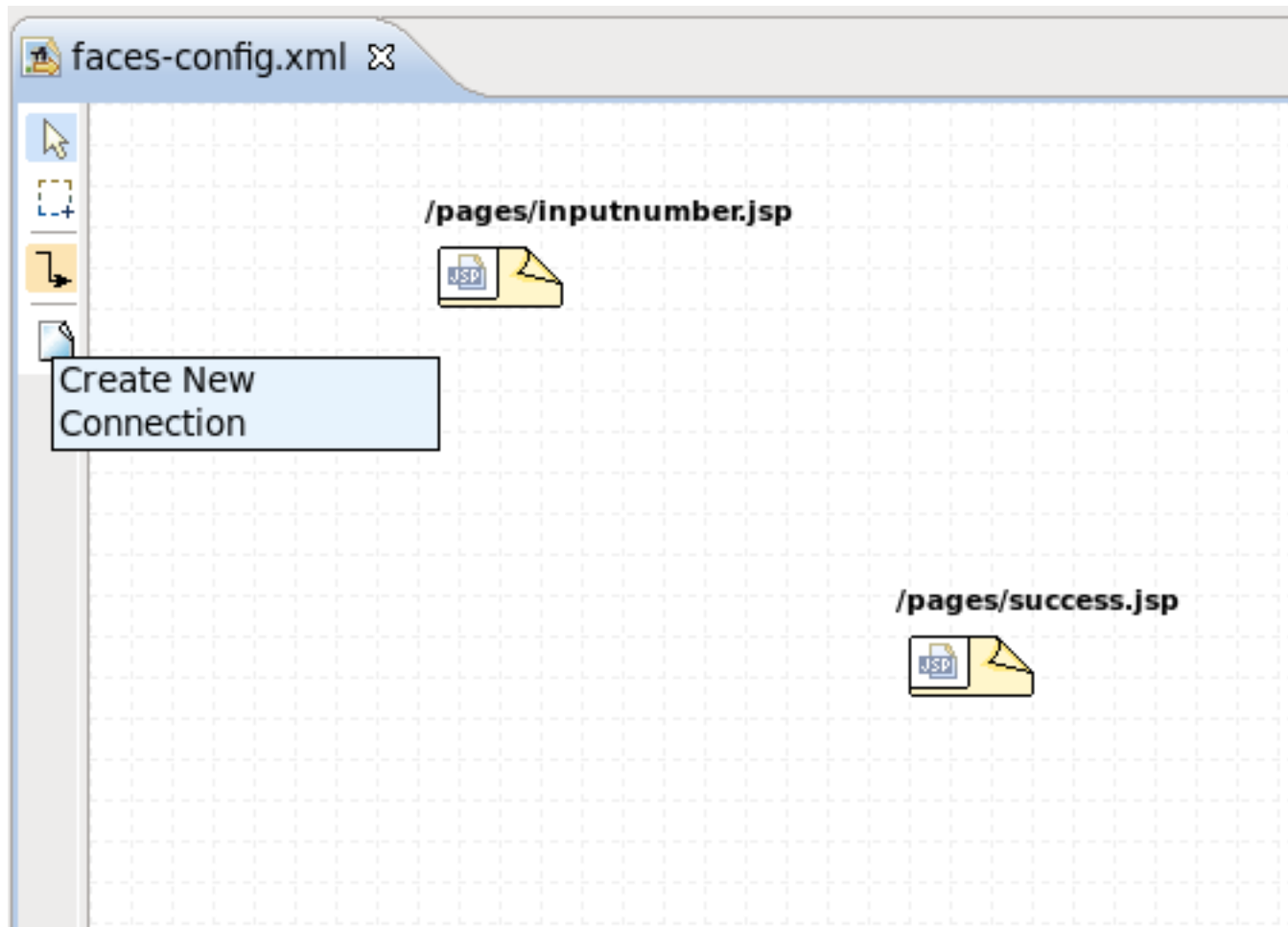


Figure 6.6. Create Connection

- Click on the *pages/inputnumber* page icon and then click on the *pages/success* page icon

A transition should appear between the two icons of views.

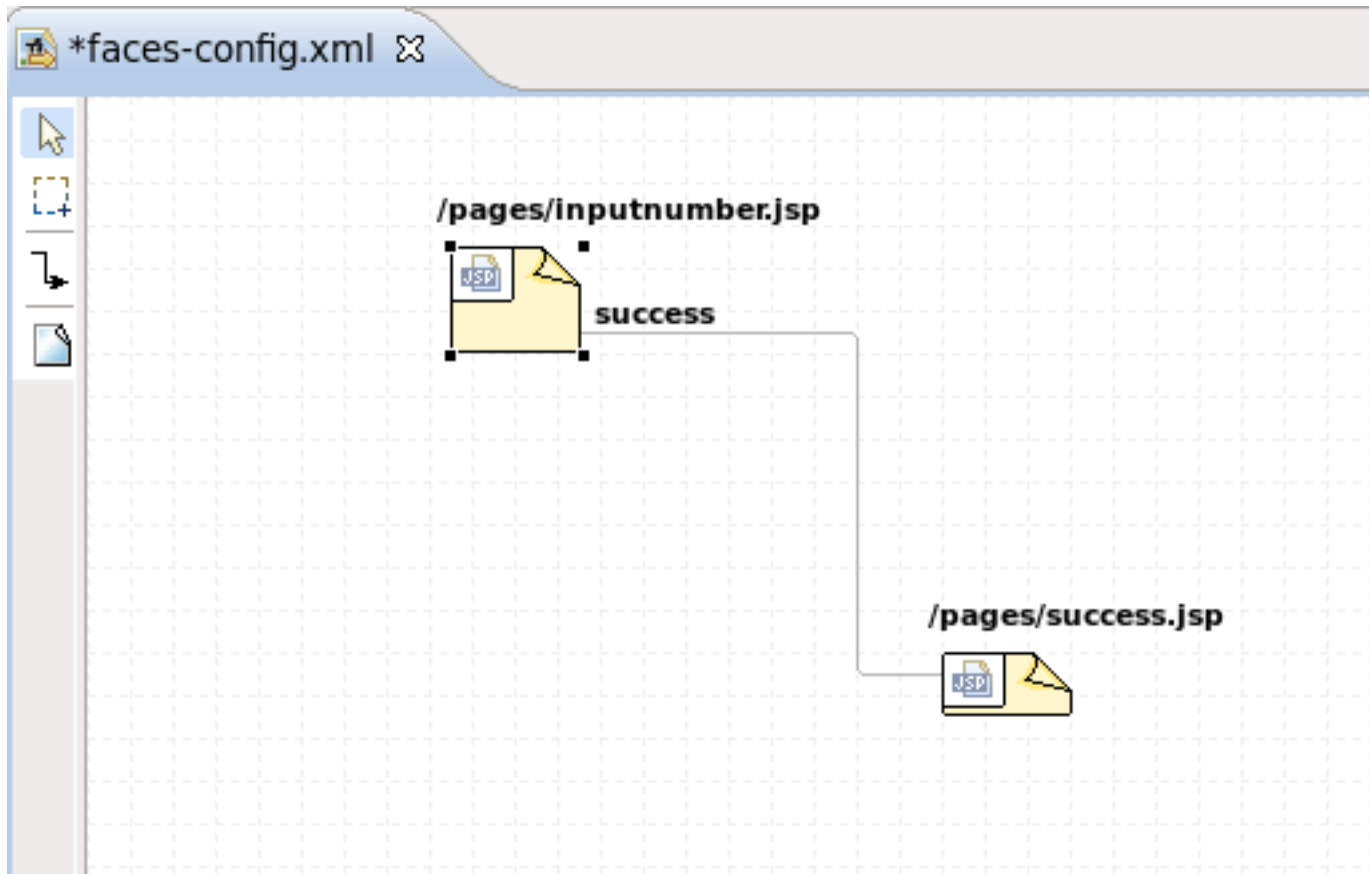


Figure 6.7. Created Connection

- Select **File** → **Save** from the menu bar

6.4. Creating Resource File

A resource file is a file with a `.properties` extension for collecting text messages in one central place. JBoss Developer Studio allows you to create quickly a resource file. The messages stored in a resource file can be displayed to you on a Web page during application execution.

With resource file you don't hard code anything into the JSP pages. It also makes it easier to translate your application to other languages. All you have to do is to translate all your messages to the other language and save them in a new properties file with a name that ends with the appropriate ISO-639 language code.

It is a good idea to keep your resources inside the `JavaSource` folder, where you keep your `.java` files. Every time you build the project, all `.properties` files will then be copied to the `classes` folder by default.

- Right click the `JavaSource` folder and select **New** → **Folder**

- Enter `game` as the Folder name and click the **Finish** button

Your resource file and java bean will be stored in this folder.

- Right click on the `game` folder and select **New** → **Properties File**
- Type `messages` as the value for "name" attribute and click the **Finish** button

JBoss Developer Studio will automatically open `messages.properties` file for editing.

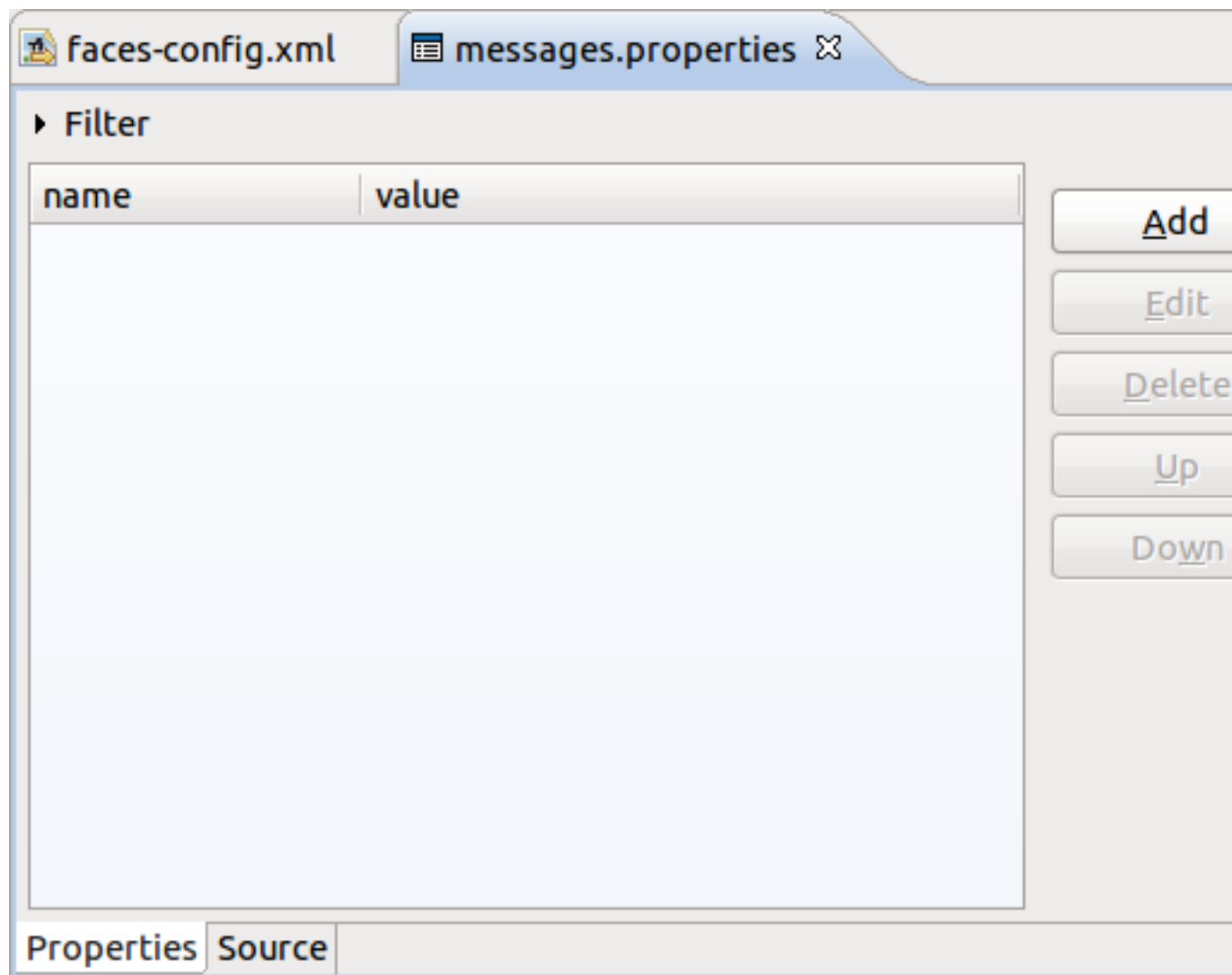


Figure 6.8. Messages.properties File

- Click the **Add** button for adding new attribute to your resource file
- Enter `how_to_play` for the "name" and `Please pick a number between 0 and 100.` for the value

- Click the **Finish** button
- Add the following properties using the same process:

```
makeguess_button=Make Guess
trayagain_button=Play Again?
success_text=How cool.. You have guessed the number, {0} is correct!
tryagain_smaller=Oops..incorrect guess. Please try a smaller number.
tryagain_bigger=Oops..incorrect guess. Please try a bigger number.
```

- Select **File** → **Save** from the menu bar

Your .properties file should now look like follows:

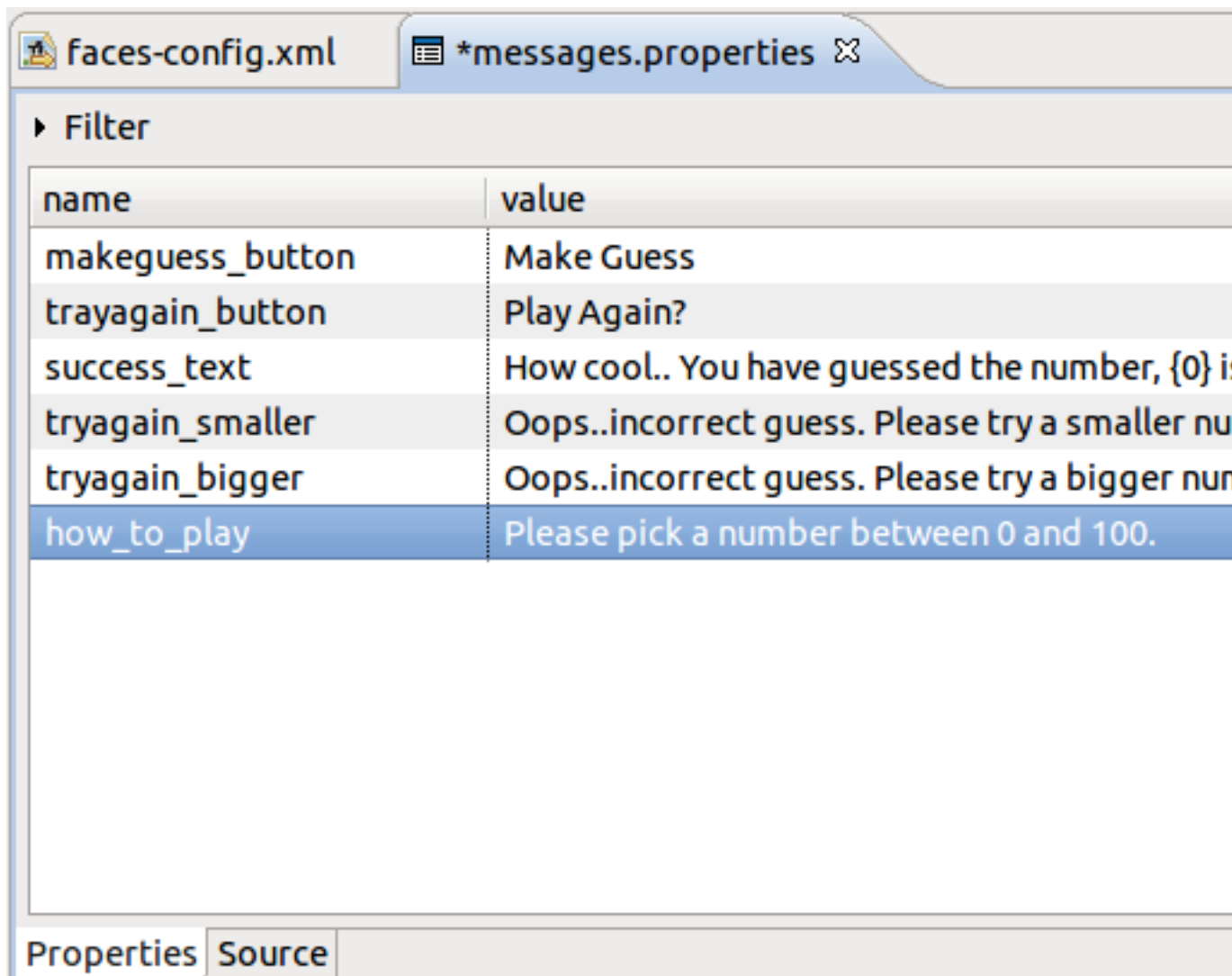


Figure 6.9. Properties are Added

The **Up** and **Down** buttons allow you to move the attributes in the list. To delete the attribute, select it and press the **Delete** button.

If you want to change a value or a name of your attribute, select it and then click the **Edit** button.

If the .properties file is rather big and there are a lot of entries in it, you can use filtering and regular expressions narrow down the list. The Filter and Regular Expressions Search is implemented by an expandable panel, closed by default:

When "Expression" is not selected (as by default), filter is case insensitive. When "Expression" is selected, filter uses regular expressions which are case sensitive

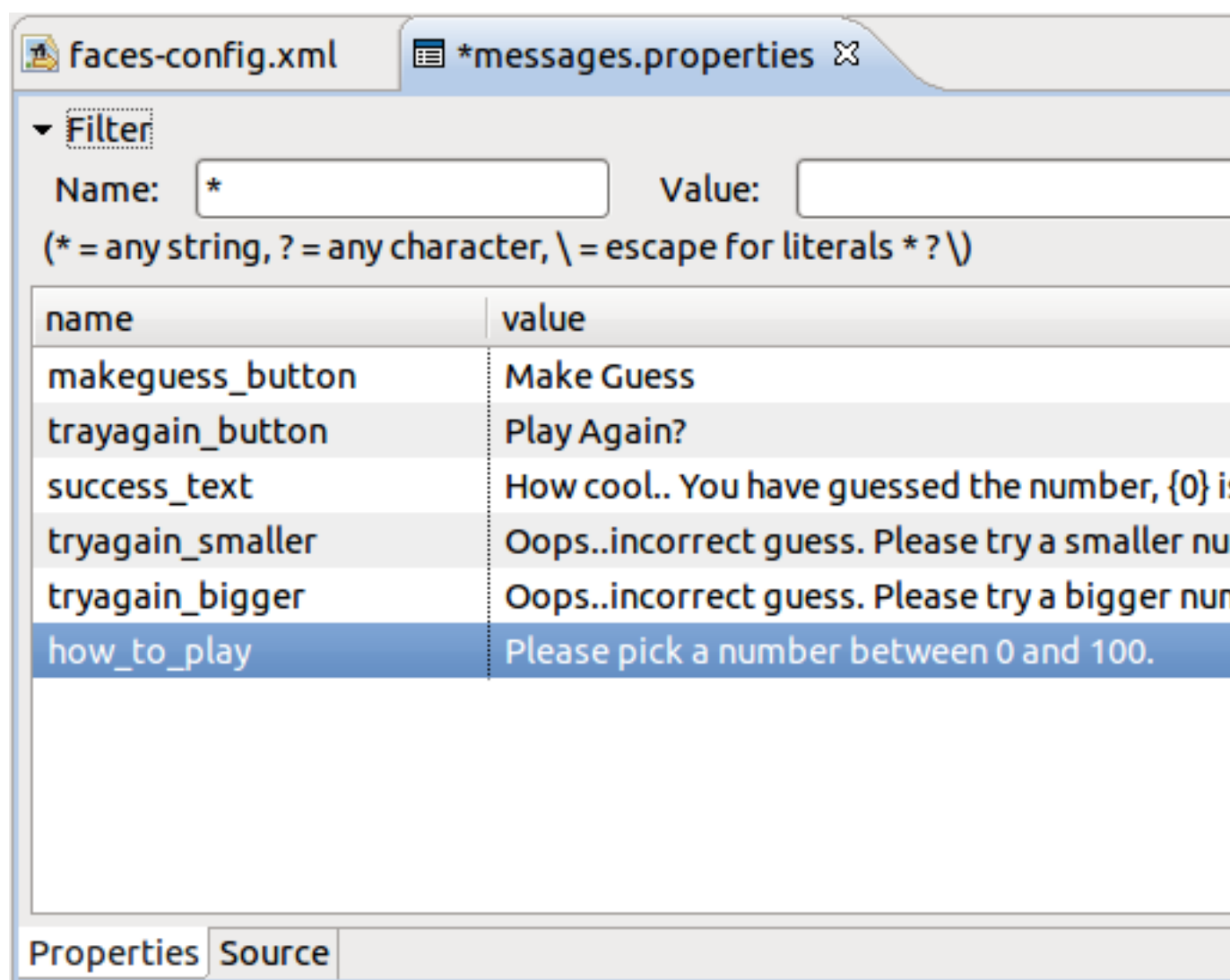


Figure 6.10. Filter and Regular Expressions Search Panel

Enter the characters that should be searched for in the entries to the 'name' or 'value' input fields accordingly. The filtered results will be displayed in the table below:

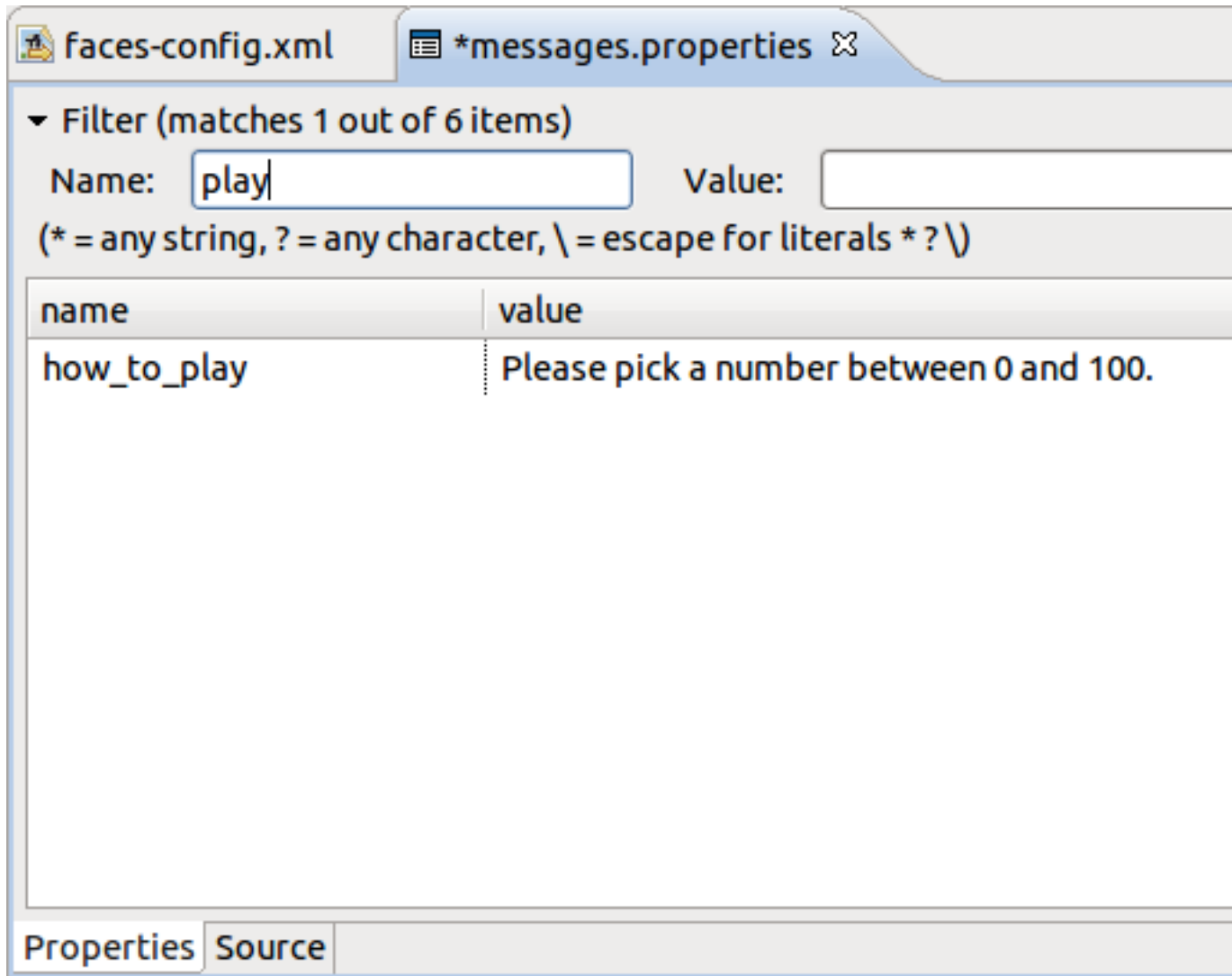


Figure 6.11. Filter results

When using regular expressions please note, that regular expression syntax does not use "*" for any characters and "?" for any one character. It's necessary to use "." for any one character and ".*" for any characters. Symbols "*" and "?" are used to show that the preceding token is not required, for example, "a.a" matches "aba" but not "aa", while "a.a?" or "a.*a" matches both; besides "a.*a" matches "abcda".

To find the exact match, use sequences \A and \z in expression. For example, expression "\Adate \z" matches only string "date"; expression "\Adate" matches "date" and "dateline", expression "date\z" matches "date" and "Begin date", and expression "date" matches all of them.

6.5. Creating a Java Bean

In this section you'll learn how to create a Java bean that will hold business logic of our application.

- Right click the game folder

- Select **New** → **Class**
- Type *NumberBean* for bean name

A java bean is created.

- Declare the variable of your entered number:

```
Integer userNumber;
```

JBoss Developer Studio allows for quick generation of getters and setters for java bean.

- Right click the *NumberBean.java* file in the Package Explorer view
- Select **Source** → **Generate Getters and Setters...**
- Check *userNumber* box and click the **OK** button

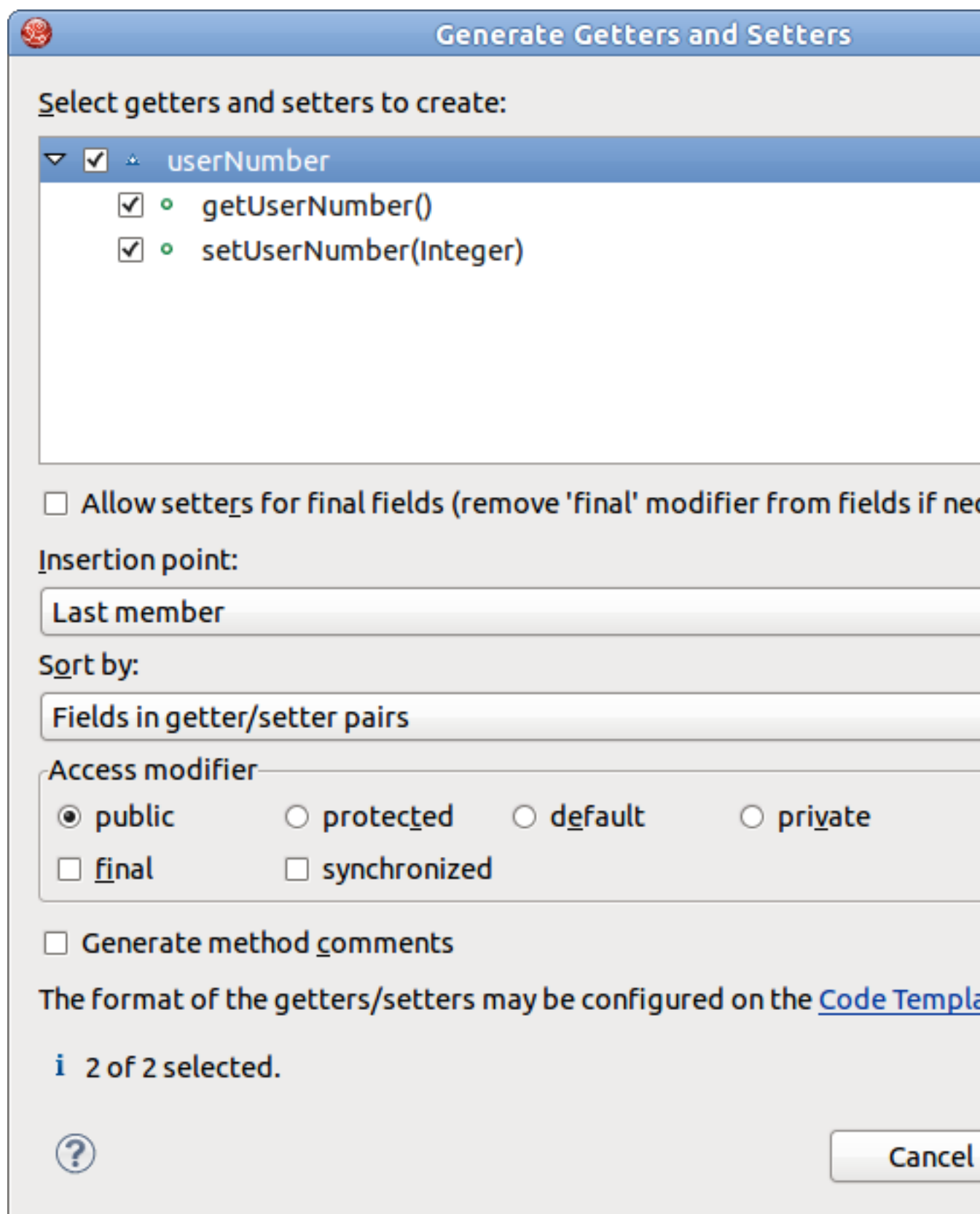


Figure 6.12. Generate Getters and Setters

- Add the declaration of the second variable

```
int randomNumber;
```

- .. other bean methods:

```
public NumberBean ()
{
    randomNumber = (int)(Math.random()*100);
    System.out.println ( "Random number: "+randomNumber);
}
public String playagain ()
{
    FacesContext context = FacesContext.getCurrentInstance();
    HttpSession session =
        (HttpSession) context.getExternalContext().getSession(false);
    session.invalidate();
    return "playagain";
}
public String checkGuess ()
{
    // if guessed, return 'success' for navigation
    if ( userNumber.intValue() == randomNumber )
    {
        return "success";
    }
    else
    {
        FacesContext context = FacesContext.getCurrentInstance();
        ResourceBundle bundle = ResourceBundle.getBundle("game.messages",
            context.getViewRoot().getLocale());
        String msg = "";
        // if number bigger, get appropriate message
        if ( userNumber.intValue() > randomNumber )
            msg = bundle.getString("tryagain_smaller");
        else // if number smaller, get appropriate message
            msg = bundle.getString("tryagain_bigger");
        // add message to be displayed on the page via <h:messages> tag
        context.addMessage (null, new FacesMessage(msg));
        // return 'tryagain' for navigation
        return "tryagain";
    }
}
```

- And the import declarations:

```
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpSession;
import javax.faces.application.FacesMessage;
import java.util.ResourceBundle;
```

The Java Bean contains the following code:

```
package game;

import javax.faces.context.FacesContext;
import javax.servlet.http.HttpSession;
import javax.faces.application.FacesMessage;
import java.util.ResourceBundle;

public class NumberBean
{
    Integer userNumber;
    int randomNumber; // random number generated by application

    public Integer getUserNumber ()
    {
        return userNumber;
    }
    public void setUserNumber (Integer value)
    {
        this.userNumber = value;
    }

    // constructor, generates random number
    public NumberBean ()
    {
        randomNumber = (int)(Math.random()*100);
        System.out.println (
            "Random number: " + randomNumber);
    }

    public String playagain ()
    {
        FacesContext context = FacesContext.getCurrentInstance();
        HttpSession session =
            (HttpSession) context.getExternalContext().getSession(false);
        session.invalidate();
        return "playagain";
    }

    // check if user guessed the number
    public String checkGuess ()
```

```
{
    // if guessed, return 'success' for navigation
    if ( userNumber.intValue() == randomNumber )
    {
        return "success";
    }
    // incorrect guess
    else
    {
        // get a reference to properties file to retrieve messages
        FacesContext context = FacesContext.getCurrentInstance();
        ResourceBundle bundle =
            ResourceBundle.getBundle("game.messages",
                context.getViewRoot().getLocale());
        String msg = "";
        // if number is bigger, get appropriate message
        if ( userNumber.intValue() > randomNumber )
            msg = bundle.getString("tryagain_smaller");
        else // if number smaller, get appropriate message
            msg = bundle.getString("tryagain_bigger");

        // add message to be displayed on the page via <h:messages> tag
        context.addMessage (null, new FacesMessage(msg));
        // return 'tryagain' for navigation
        return "tryagain";
    }
}
}
```

6.6. Editing faces-config.xml File

In this section you will learn about the `faces-config.xml` file.

This file holds two navigation rules and defines the backing bean used.

- Open the `faces-config.xml` file in a source mode
- Here we will add one more navigation rule and a managed bean declaration, so that the content of the file looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
    version="1.2"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xi="http://www.w3.org/2001/XInclude"
```

```

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-facesconfig_1_2_.xsd">

<navigation-rule>
  <from-view-id>*</from-view-id>
  <navigation-case>
    <from-outcome>playagain</from-outcome>
    <to-view-id>/pages/inputnumber.jsp</to-view-id>
  </navigation-case>
</navigation-rule>

<navigation-rule>
  <from-view-id>/pages/inputnumber.jsp</from-view-id>
  <navigation-case>
    <from-outcome>success</from-outcome>
    <to-view-id>/pages/success.jsp</to-view-id>
  </navigation-case>
</navigation-rule>

<managed-bean>
  <managed-bean-name>NumberBean</managed-bean-name>
  <managed-bean-class>game.NumberBean</managed-bean-class>
  <managed-bean-scope>session</managed-bean-scope>
</managed-bean>

</faces-config>

```

The first navigation rule states that from any page (* stands for any page) an outcome of playagain will take you to the /pages/inputnumber.jsp file. Outcome values are returned from backing bean methods in this example. The second navigation rule states that if you are at the page /pages/inputnumber.jsp, and the outcome is success, then navigate to the /pages/success.jsp page.

6.7. Editing the JSP View Files

Now, we will continue editing the JSP files for our two "views" using the Visual Page Editor.

6.7.1. Editing inputnumber.jsp page

First, edit the inputnumber.jsp file.

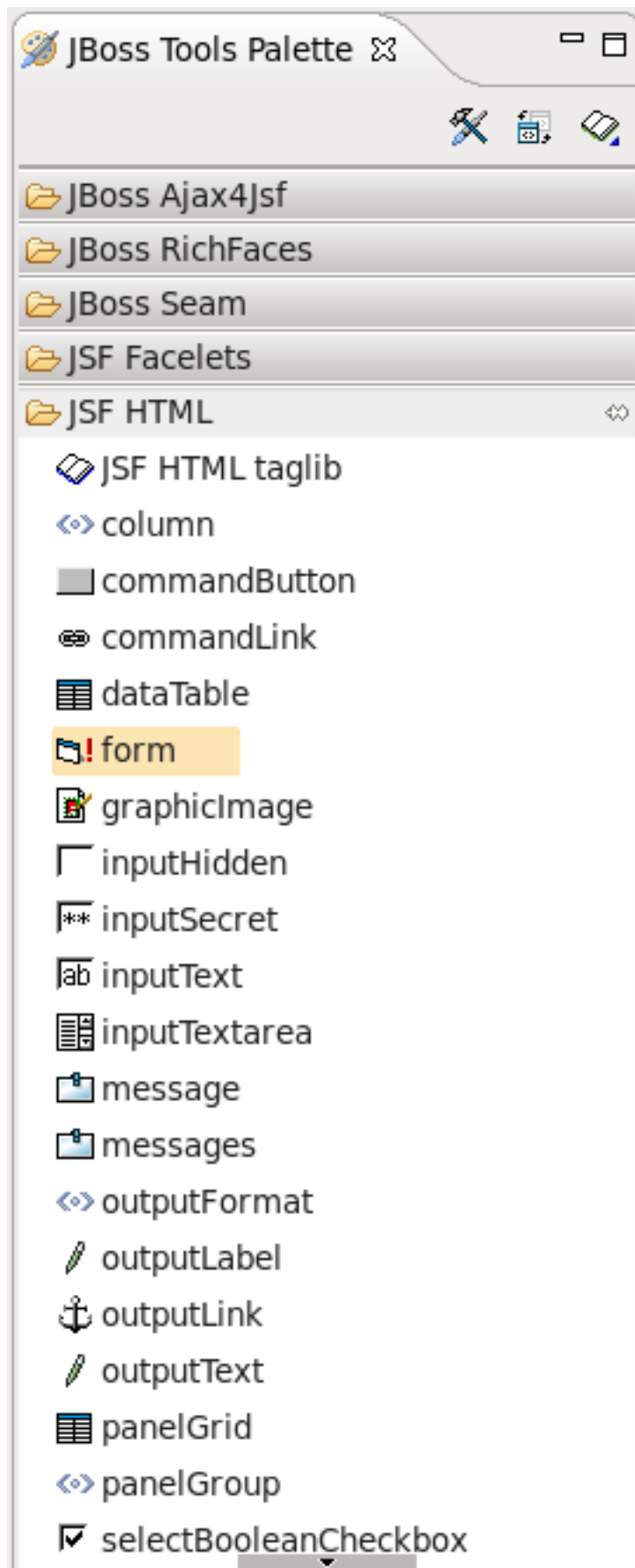
On this page we will have an output text component displaying a message, a text field for user's number entering and a button for input submission.

- Open the inputnumber.jsp file by double-clicking on the /pages/inputnumber.jsp icon

The Visual Page Editor will open in a screen split between source code along the top and a WYSIWIG view along the bottom. You can see that some JSF code will have already been generated since we chose a template when creating the page.

At the beginning it's necessary to create a `<h:form>` component that will hold the other components.

- Place the mouse cursor inside the `<f:view></f:view>` tag
- Go to JBoss Tools Palette and expand JSF HTML folder by selecting it
- Click on the `<h:form>` tag



- In the Insert Tag dialog select the *id* field and click on the second column. A blinking cursor will appear in a input text field inviting to enter a value of id

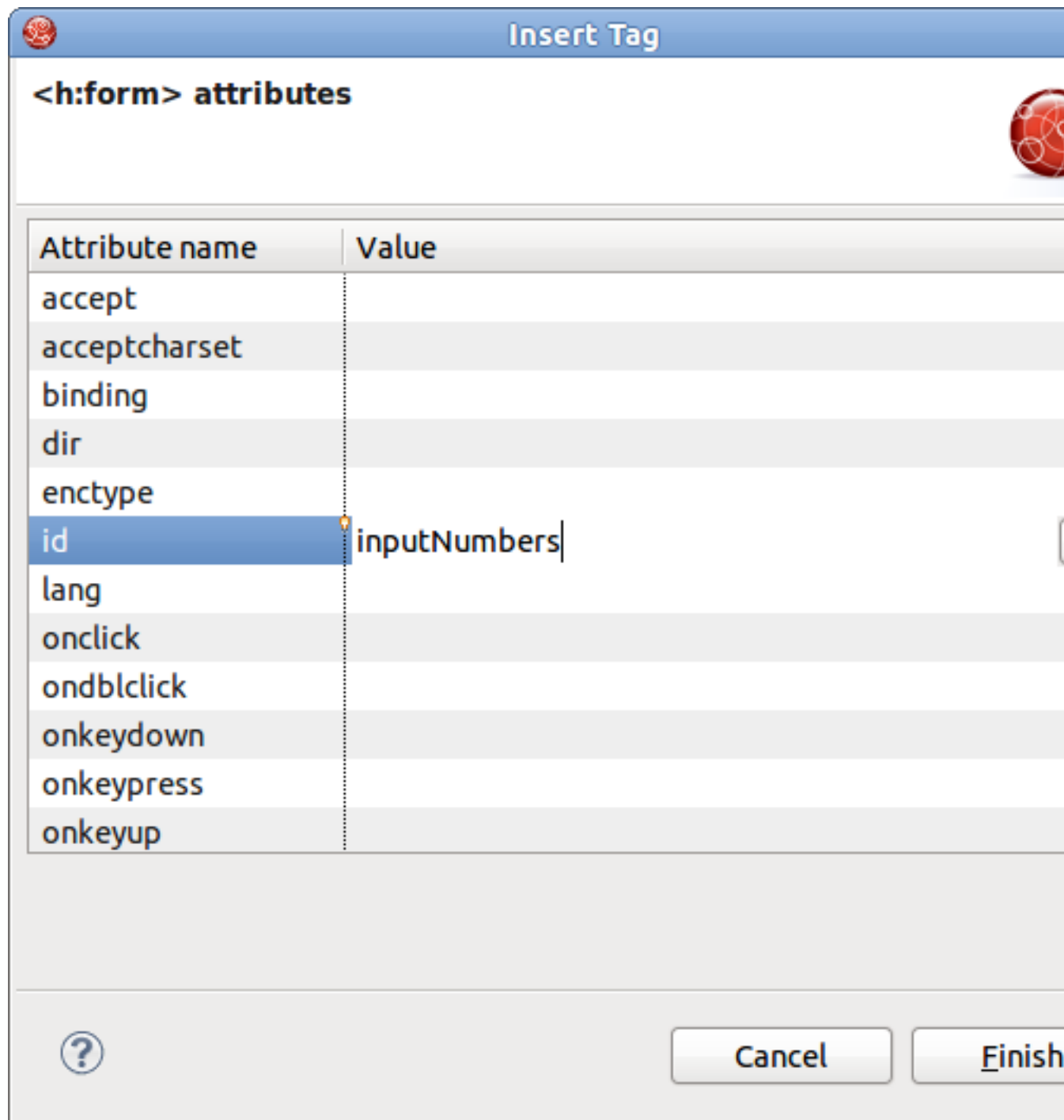


Figure 6.14. Define Id of Form

- Enter *inputNumbers* and click the **Finish** button

In source view you can see the declaration of a form.

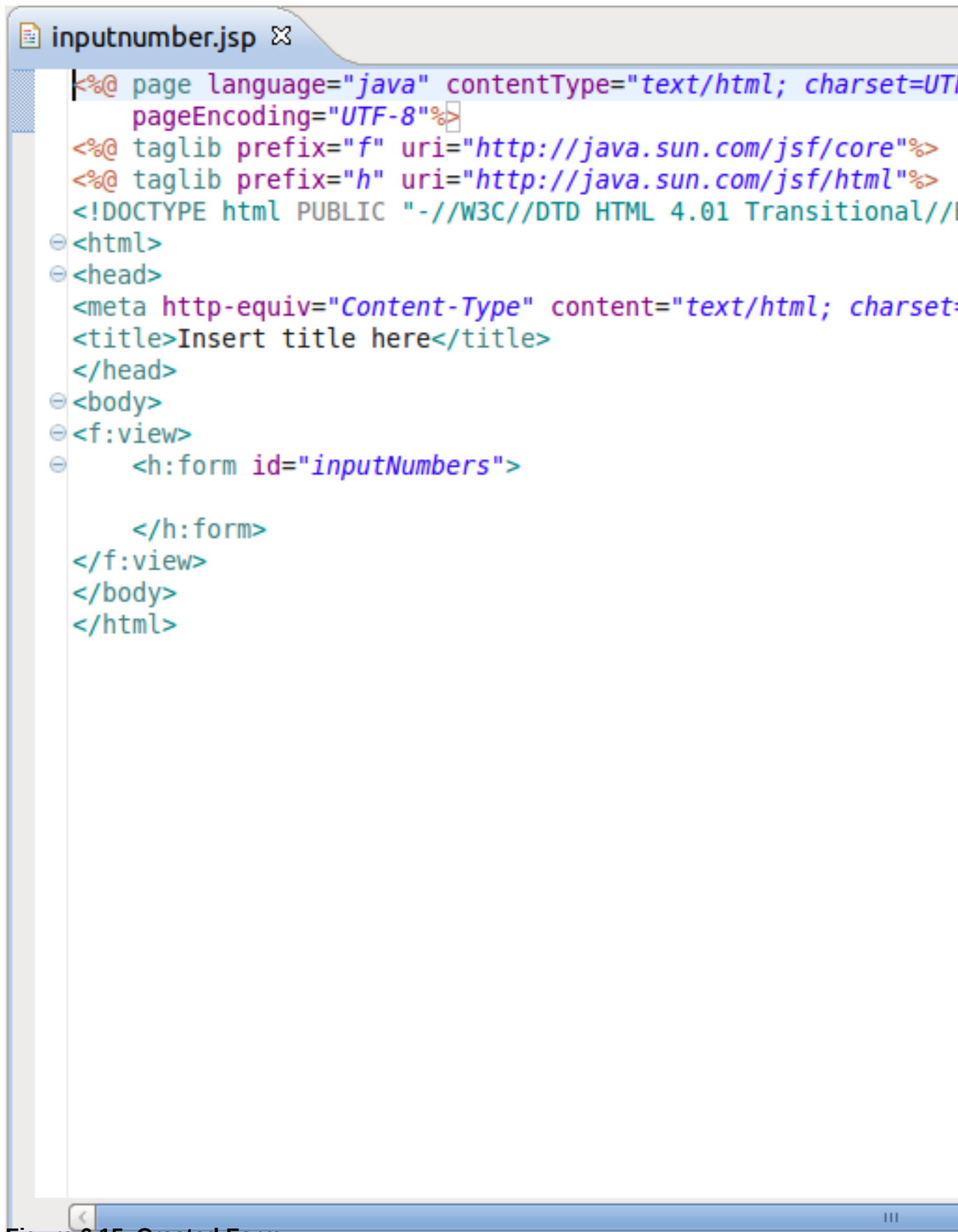


Figure 6.15. Created Form

First let's declare the properties file in the `inputnumber.jsp` page using the `loadBundle` JSF tag.

- Add this declaration on the top of a page, right after the first two lines:

```
<f:loadBundle basename="game.messages" var="msg"/>
```

As always JBoss Developer Studio provides code assist:

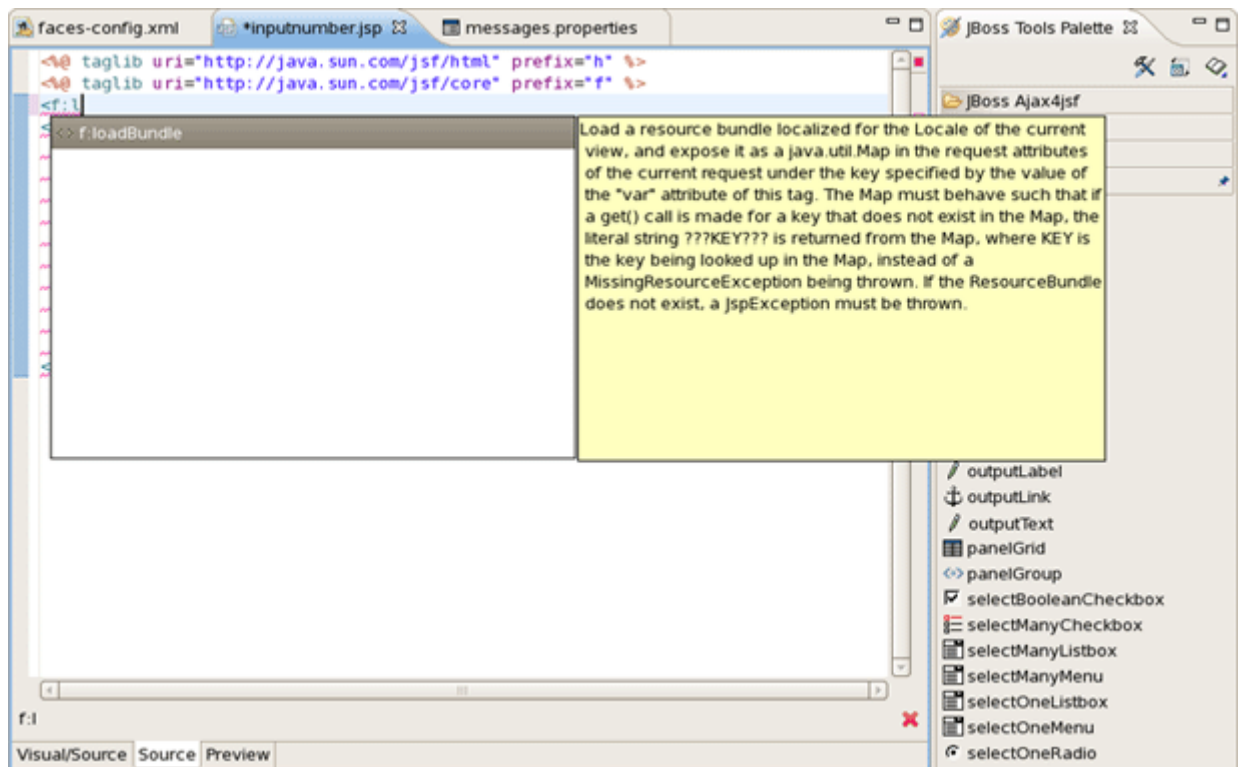


Figure 6.16. Code Assist

- Switch to Visual tab, where it is possible to work with the editor through a WYSIWYG interface
- Click the `outputText` item from the **JSF HTML** group in the **JBoss Tools Palette** view, drag the cursor over to the editor, and drop it inside the blue box in the editor
- Select the second column in the value row.
- Click the ... button next to the value field

JBoss Developer Studio will display a list of possible values:

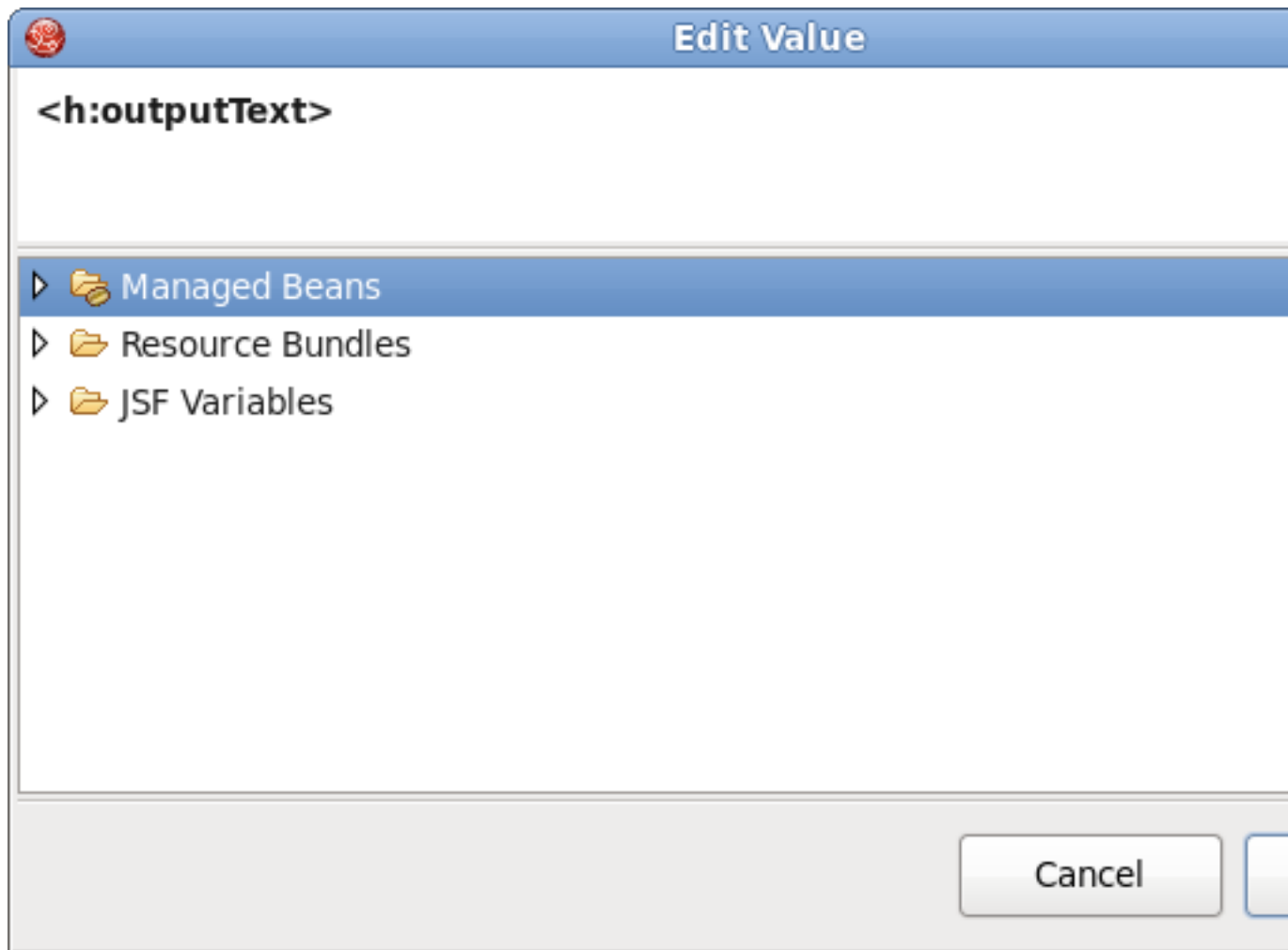


Figure 6.17. Choose Value

- Expand **Resource Bundles** → **msg**
- Select the *how_to_play* value and click the **OK** button. Then click the **Finish** button.

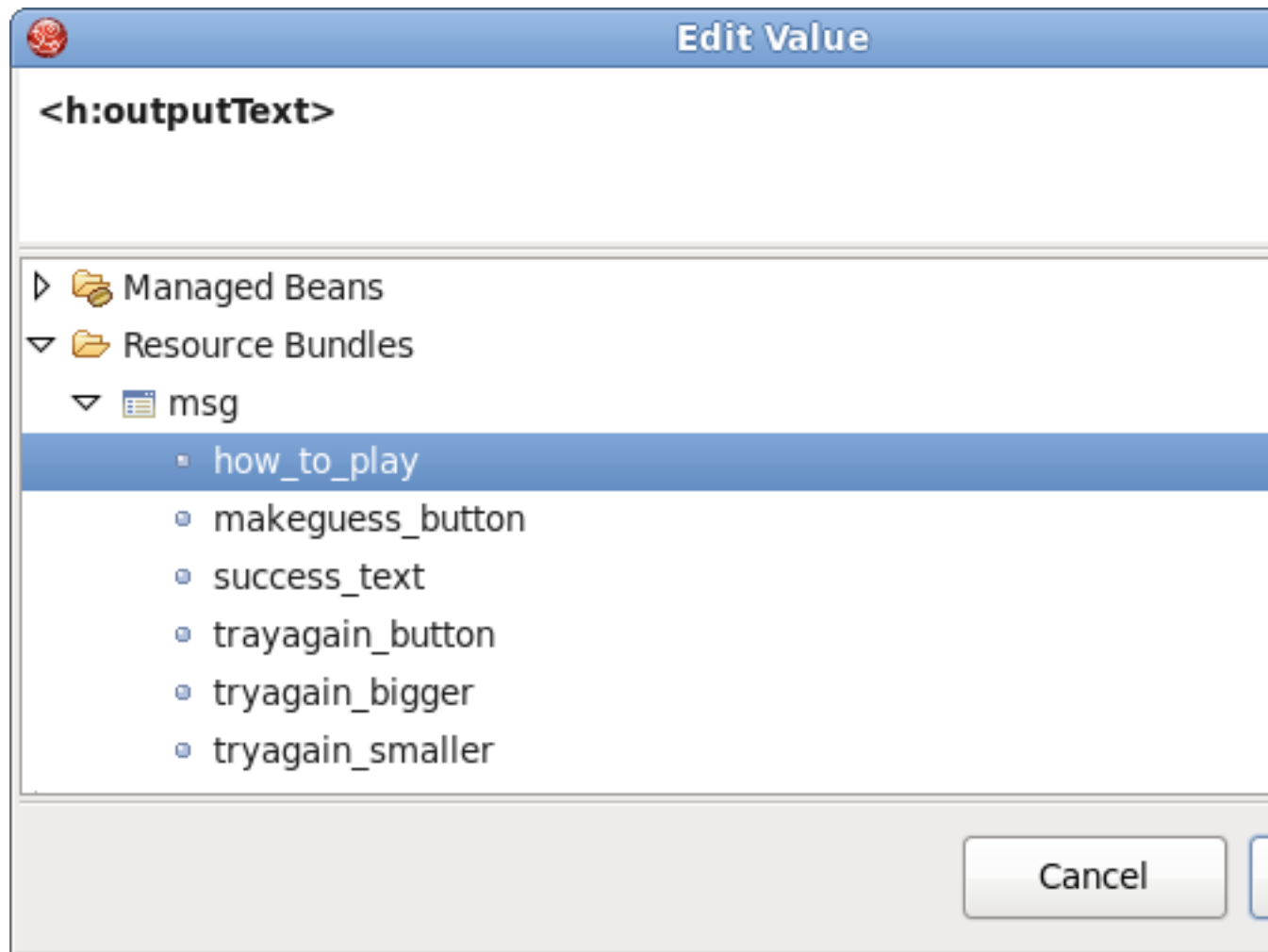


Figure 6.18. Selecting Value

The text will appear on the page:

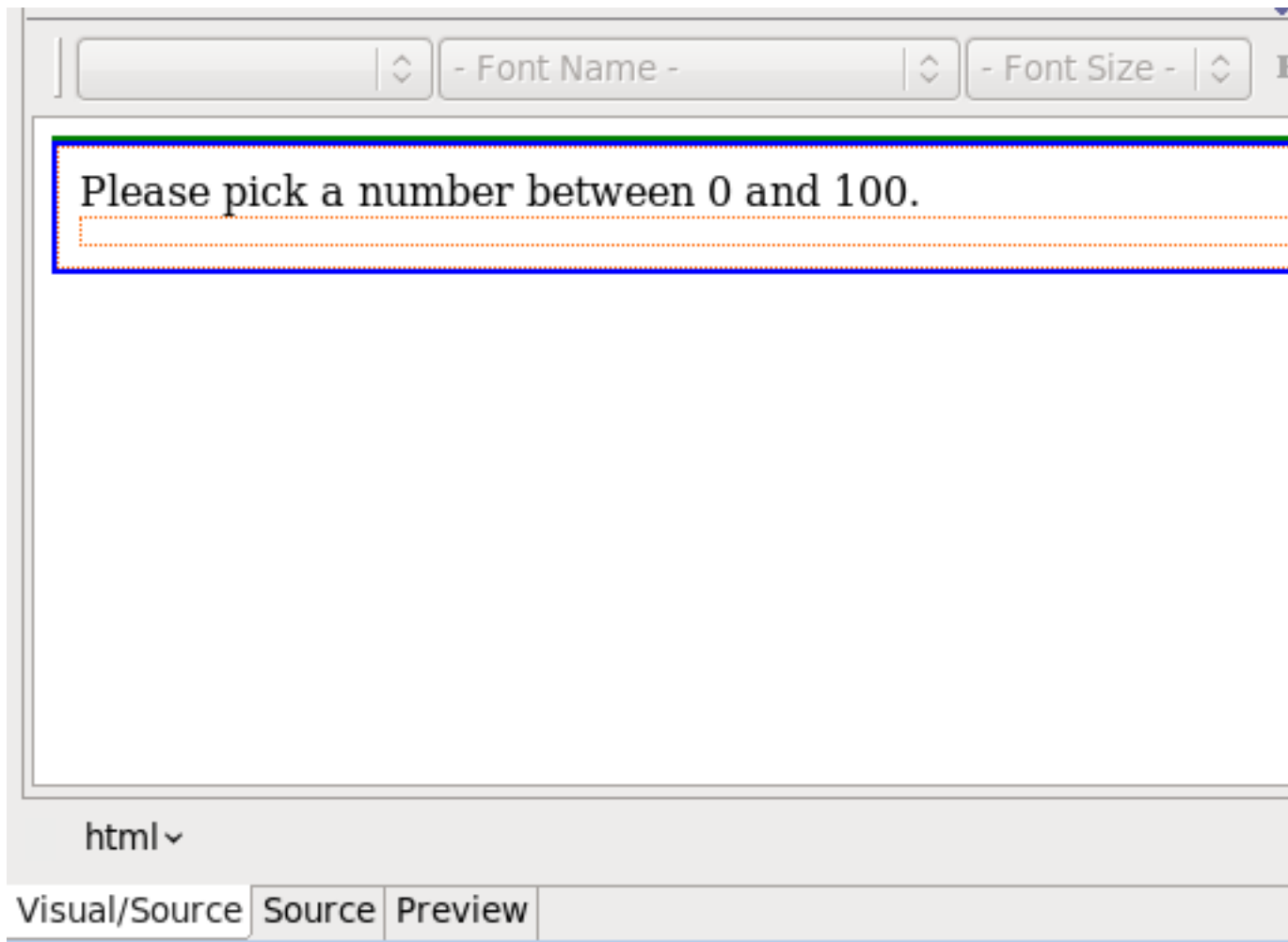


Figure 6.19. Created OutputText Component

- Switch to Source mode and insert a `<br/>` tag after the `<h:outputText>` component to make a new line
- Click the **Save** button
- On the Palette click on *inputText*, drag the cursor over to the editor, and drop it inside the editor after the text
- Select the *value* row and click in the second column
- Click the ... button next to the value field
- Expand **Managed Beans** → **NumberBean**
- Select *userNumber* value and click the **OK** button
- Select the *Advanced* tab

- Select the *id* row and click in the second column
- Type *userNumber* in the text field
- Select the *required* row and click in the second column
- Click ... button next to the value field
- Expand *Enumeration* and select *true* as a value

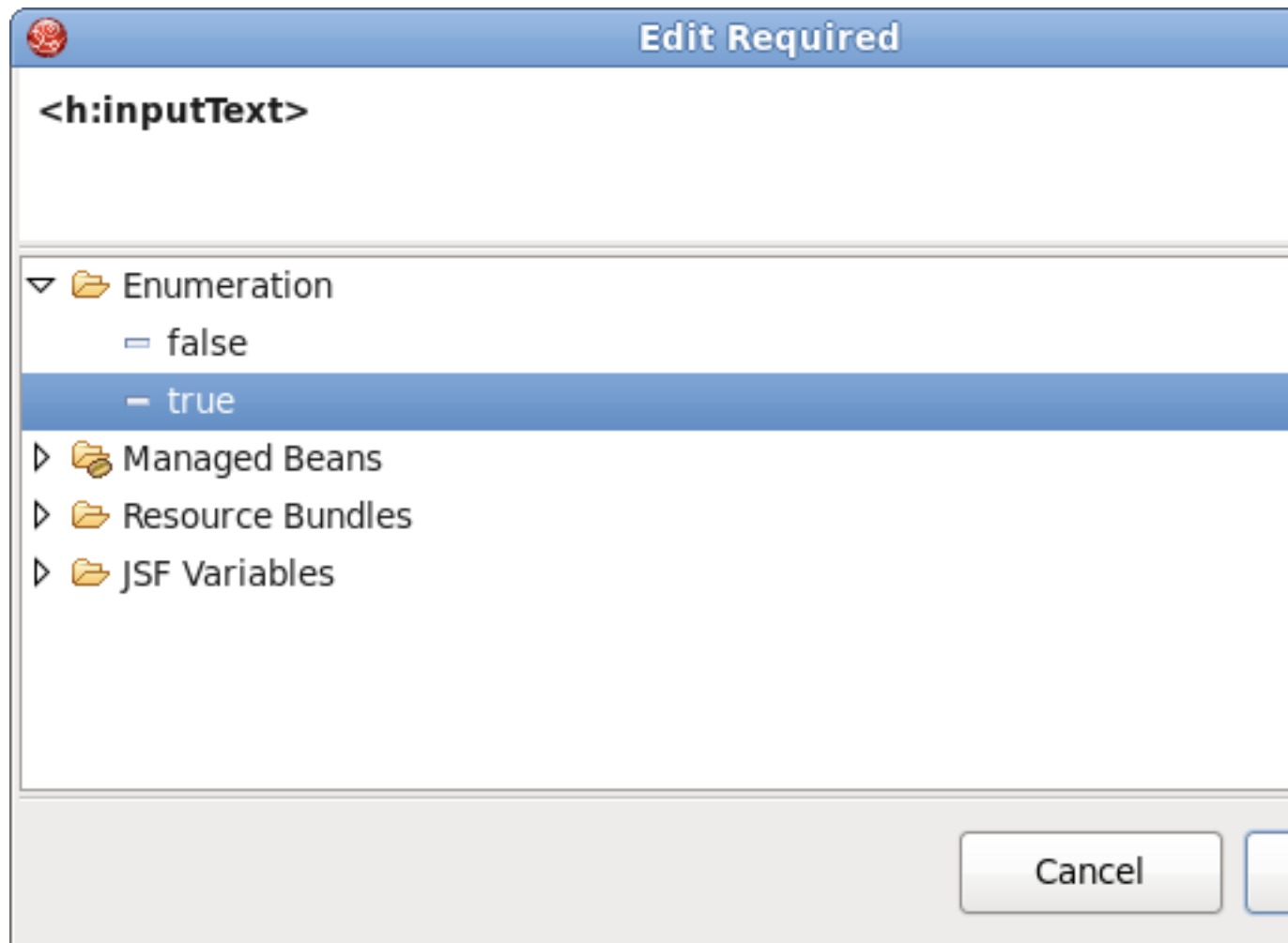


Figure 6.20. Add "required" Attribute

- Click the **OK** button, then click the **Finish** button
- Go to Source mode
- Add the validation attribute to `<f:validateLongRange>` for user input validation

```
<h:inputText id="userNumber" value="#{NumberBean.userNumber}" required="true">
```

```
<f:validateLongRange minimum="0" maximum="100"/>
</h:inputText>
```

- Click the **Save** button
- Again select *Visual* mode
- On the Palette, click on *commandButton*, drag the cursor over to the editor, and drop it inside the editor after the *inputText* component.
- In the editing dialog select the *value* row and click on the second column
- Click the ... button next to the value field
- Expand **Resource Bundles** → **msg** and select *makeguess_button* as a value
- Click the **OK** button
- Select the *action* row and click in the second column
- Type `#{NumberBean.checkGuess}` in the text field
- Click the **Finish** button
- In Source mode add `<br/>` tags between the `<outputText>`, `<inputText>` and `<commandButton>` components to place them on different lines

inputnumber.jsp page should look like this:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="f" uri="http://java.sun.com/jsf/core"%>
<%@ taglib prefix="h" uri="http://java.sun.com/jsf/html"%>
<f:loadBundle basename="game.messages" var="msg"/>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<f:view>
<h:form id="inputNumbers">
<h:outputText value="#{msg.how_to_play}"/>
<br/>
<h:messages style="color: blue" />
```

```
<br/>
                <h:inputText          id="userNumber"          required="true"
value="#{NumberBean.userNumber}">
    <f:validateLongRange minimum="0" maximum="100" />
</h:inputText>
<br/>
<br/>
                <h:commandButton      action="#{NumberBean.checkGuess}"
value="#{msg.makeguess_button}"/>
    </h:form>
</f:view>
</body>
</html>
```

6.7.2. Editing success.jsp page

We now edit the `success.jsp` page in the same way as we just edited the `inputnumber.jsp` file. The code for the `success.jsp` page should look like the following:

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<f:loadBundle basename="game.messages" var="msg"/>

<html>
<head>
    <title></title>
</head>
<body>
    <f:view>
        <h:form id="result">
            <h:outputFormat value="#{msg.success_text}">
                <f:param value="#{NumberBean.userNumber}" />
            </h:outputFormat>
            <br />
            <br />
            <h:commandButton value="#{msg.trayagain_button}"
                action="#{NumberBean.playagain}" />
        </h:form>
    </f:view>
</body>
</html>
```

Again you can use code assist provided by JBoss Developer Studio when editing jsp page:

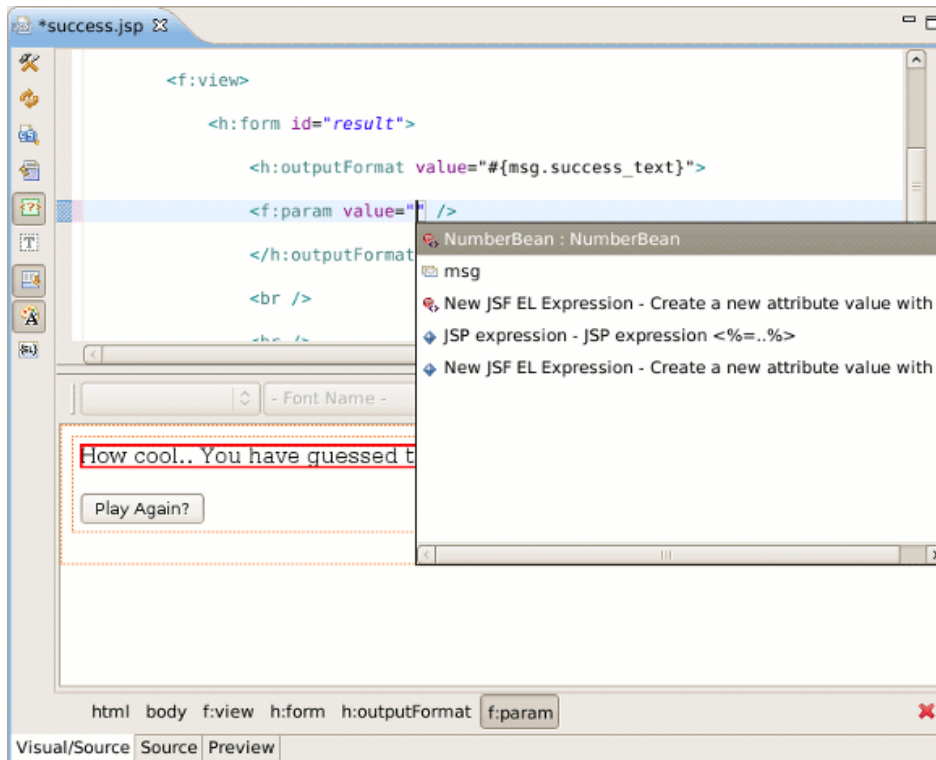


Figure 6.21. Code Assist for `<f:param>`

The `success.jsp` page is shown if you correctly guessed the number. The `<h:outputFormat>` tag will get the value of `success_text` from the properties file. The `{0}` in `success_text` will be substituted for by the value of the `value` attribute within the `<f:param>` tag during runtime.

In the final result you have a button which allows you to replay the game. The action value references a backing bean method. In this case, the method only terminates the current session so that when you are shown the first page, the input text box is clear and a new random number is generated.

- Switch to Preview mode to see how this page will look in a browser:

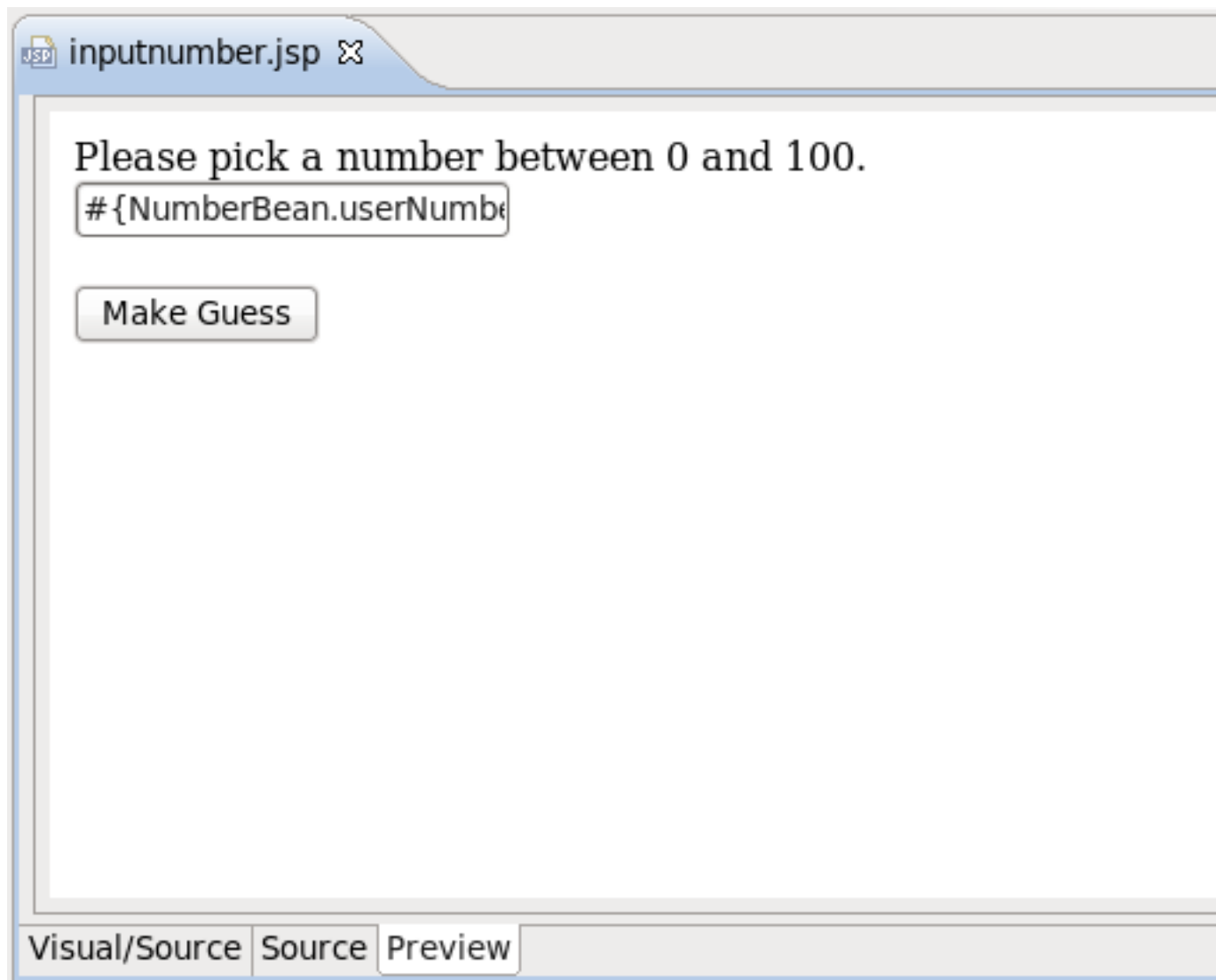


Figure 6.22. Success.jsp in Preview Mode

6.8. Creating index.jsp page

Now we need to create the `index.jsp` page.

The `index.jsp` page is the entry point of our application. It's just forwarding to the `inputnumber.jsp` page.

- Right click the `webContent` folder and select **New** → **JSP File**
- Enter `index` for name field and click the **Next** button.
- Untick the **Use JSP Template** check box and click the **Finish** button.
- Edit the source of the file so it looks like the following:

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
  <body>
    <jsp:forward page="/pages/inputnumber.jsf" />
  </body>
</html>
```

Note the *.jsf* extension of a page. It means that we trigger the JSF controller servlet to handle the page according the servlet mapping in the `faces-config.xml` file.

6.9. Running the Application

Finally, we have all the pieces needed to run the application.

- Start up JBoss server by clicking on the **Start** icon in the Servers view. (If the JBoss Server is already running, stop it by clicking on the red icon and then start it again. After the messages in the Console tabbed view stop scrolling, JBoss is available)
- Right-click on the project and select **Run As** → **Run on Server**
- Play with the application by entering correct as well as incorrect values

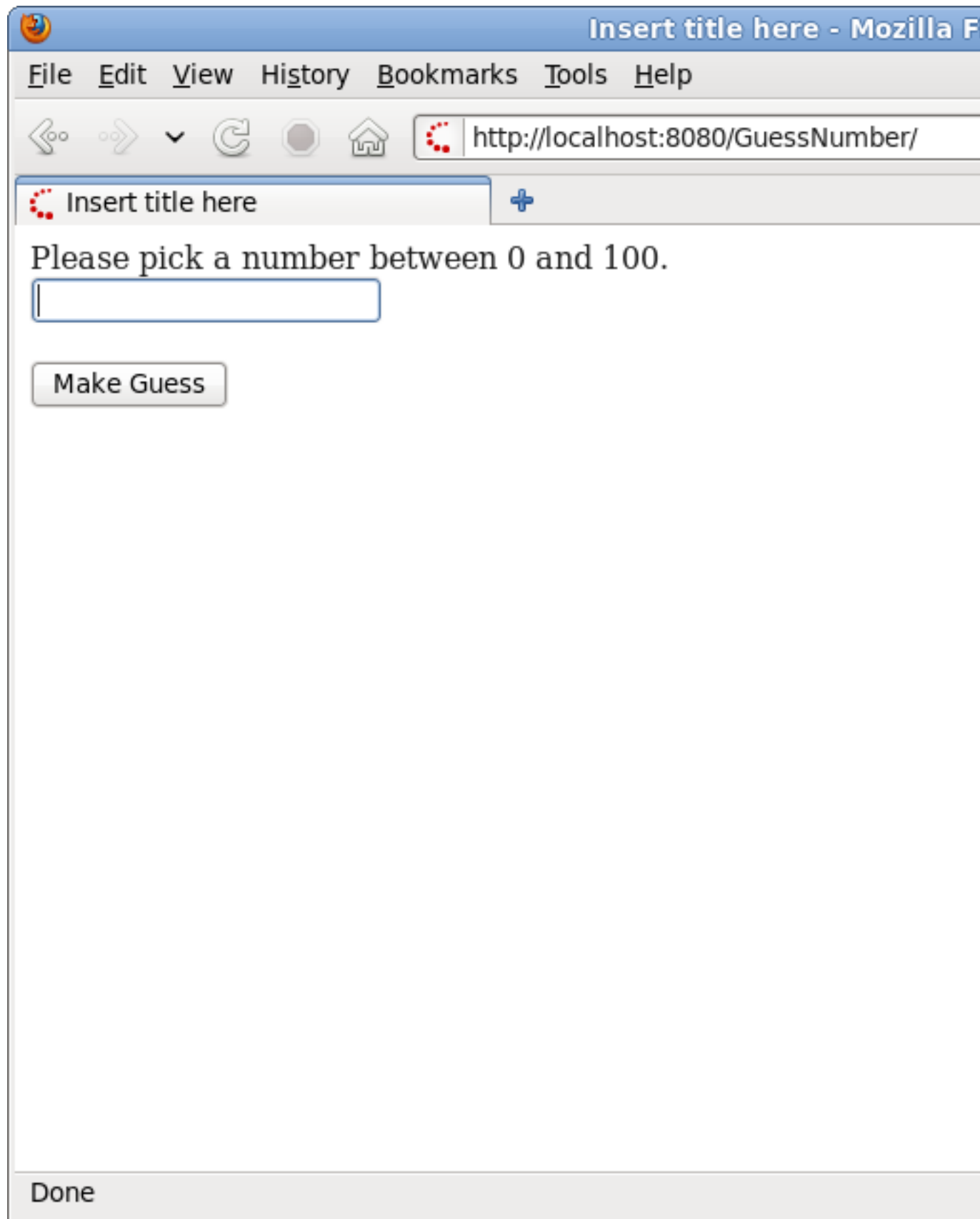


Figure 6.23. You are Asked to Enter a Number Between 0 and 100

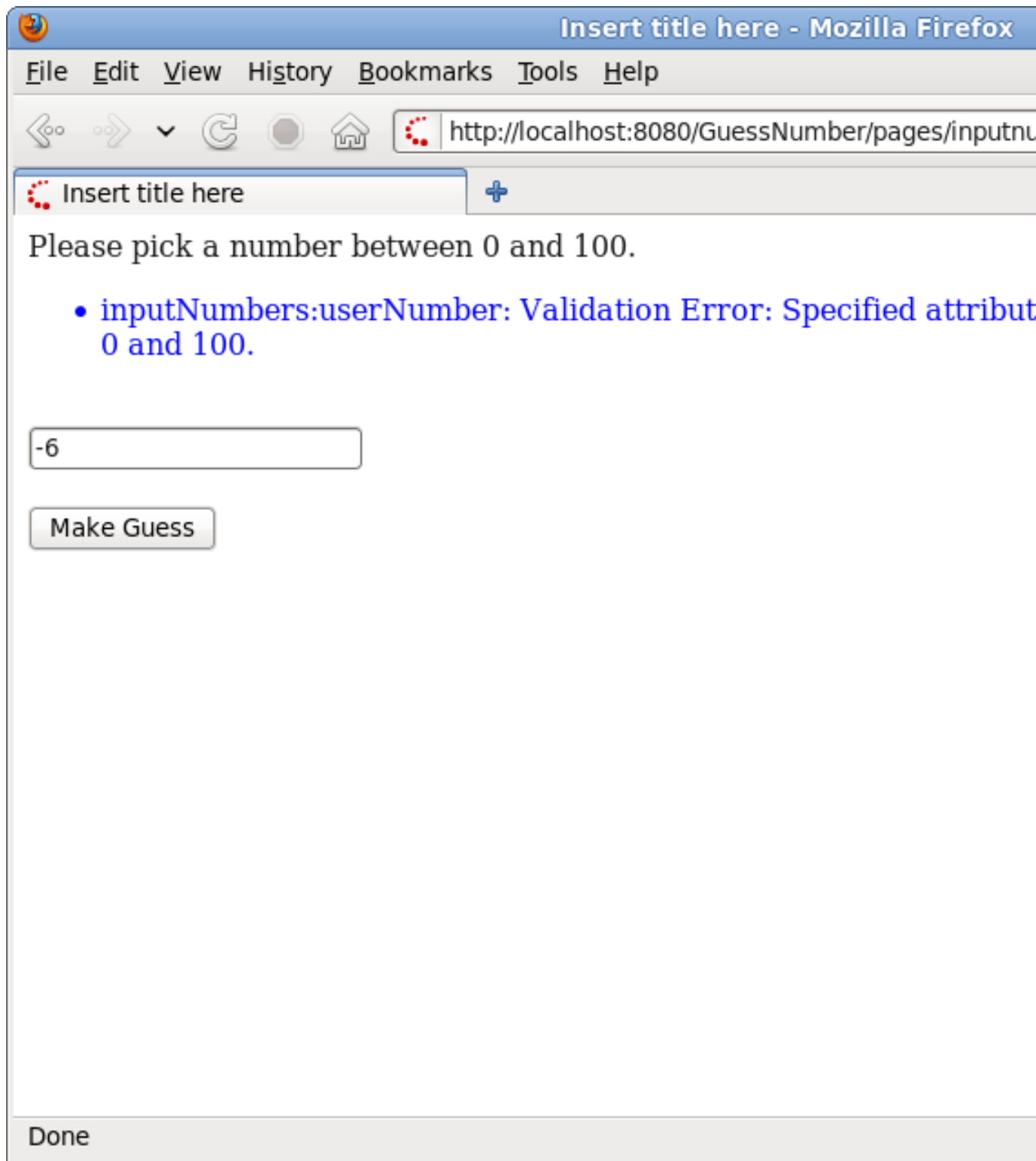


Figure 6.24. Your Input is Validated and an Error Message is Displayed if Invalid Input was Entered

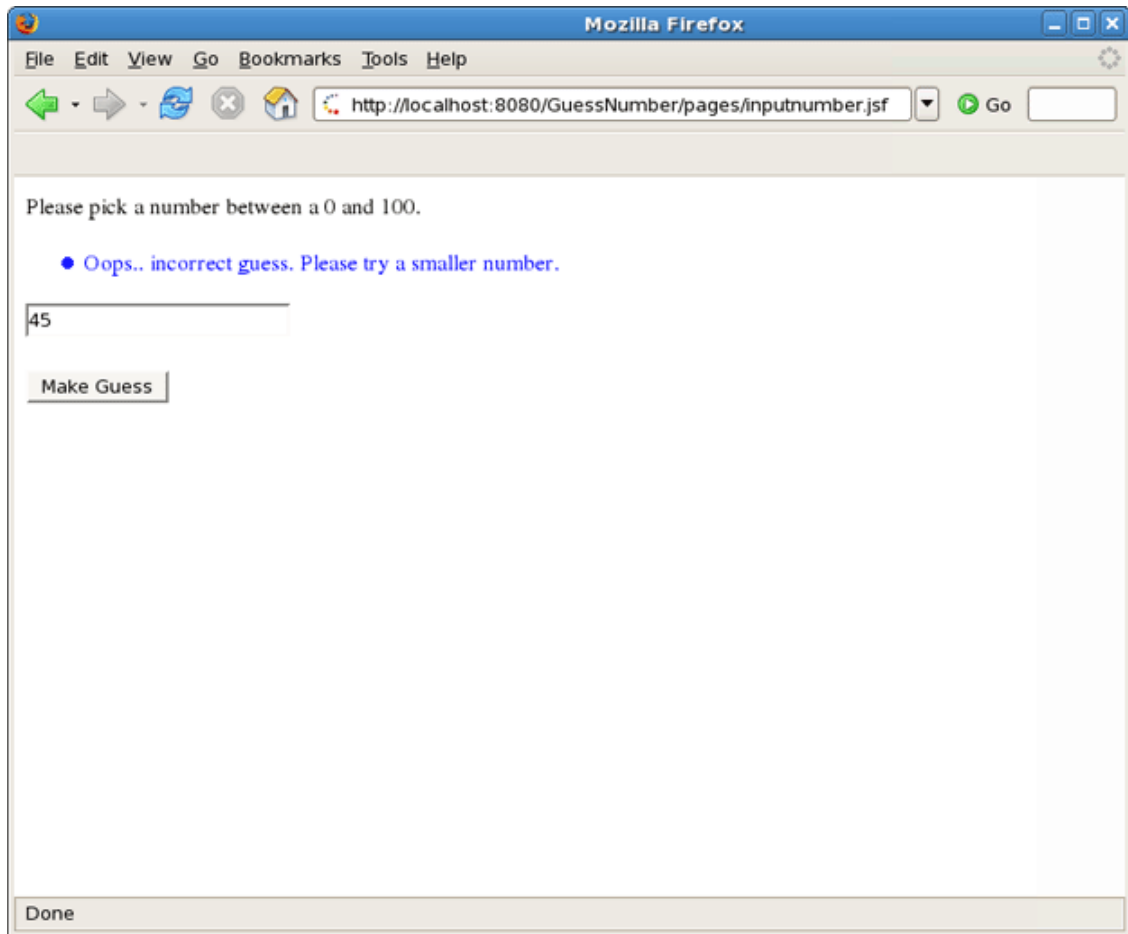


Figure 6.25. After You Enter a Guess, the Application Tells You Whether a Smaller or a Larger Number Should be Tried

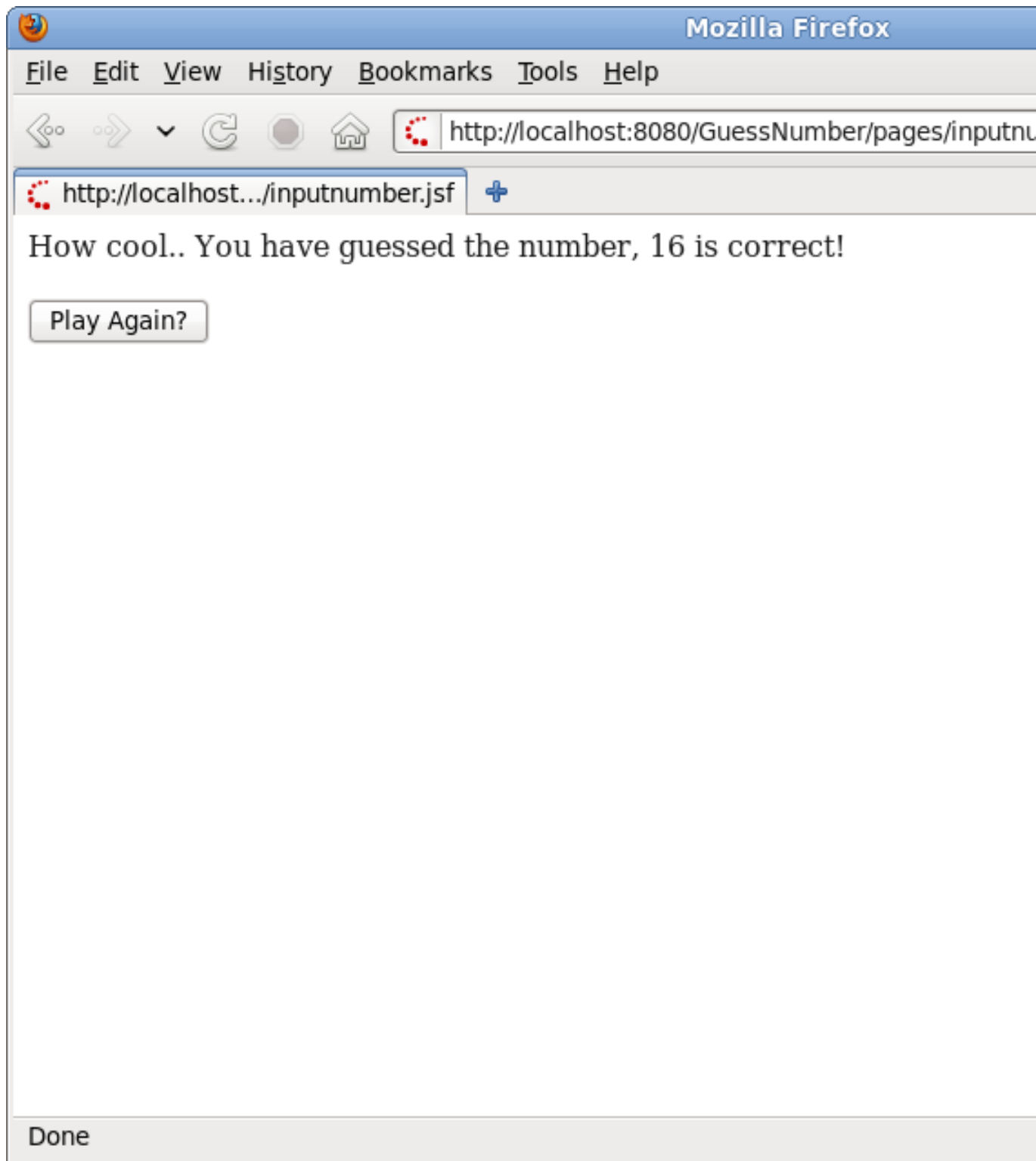


Figure 6.26. Your Guess is Correct

Uninstalling the JBoss Developer Studio

- Ensure JBoss Developer Studio is not running
- Run the Uninstaller

FAQ

Refer to the following FAQ to get the answers on the most "popular" questions concerning JBoss Developer Studio.

8.1. What should I do if the Visual Page Editor does not start under Linux

Linux users may need to do the following to get the Visual Page Editor to work correctly on their machines.

1. On Red Hat based Linux distributions install the libXp.i386 package

2. Type

```
ln -s libstdc++.so.5.0.7 libstdc++.so.5
```

3. and/or use

```
yum install libXp
```

4. Open the JBoss Developer Studio perspective. If you see the Help view open, close it and restart JBoss Developer Studio

5. If it doesn't help and you use Fedora with Eclipse Version: 3.4.1, the issue can be produced because the libswt-xulrunner-gtk-3449.so file doesn't present in eclipse-swt-3.4.1-5.fc10.x86_64.rpm/eclipse/plugins/org.eclipse.swt.gtk.linux.x86_64_3.4.1.v3449c.jar. To add this file to eclipse you should:

- Decompress eclipse/plugins/org.eclipse.swt.gtk.linux.x86_3.4.1.v3449c.jar from eclipse-SDK-3.4.1-linux-gtk-x86_64.tar.gz
- Copy libswt-xulrunner-gtk-3449.so file to your Fedora Eclipse location.
- Open the file jbdevstudio.ini, which can be found in your Fedora Eclipse location and add the following line:

```
-Dswt.library.path=/usr/lib/eclipse
```

,where /usr/lib/eclipse is the path to your eclipse folder.

6. If none of these work, do the following:

- Clear the JBoss Developer Studio log file, <workspace>\.metadata\.log
- Start JBoss Developer Studio with the -debug option:

```
jbdevstudio -debug
```

- Post the JBoss Developer Studio log file(<workspace>\.metadata\.log) on the forums.

8.2. Visual Editor starts OK, but the Missing Natures dialog appears

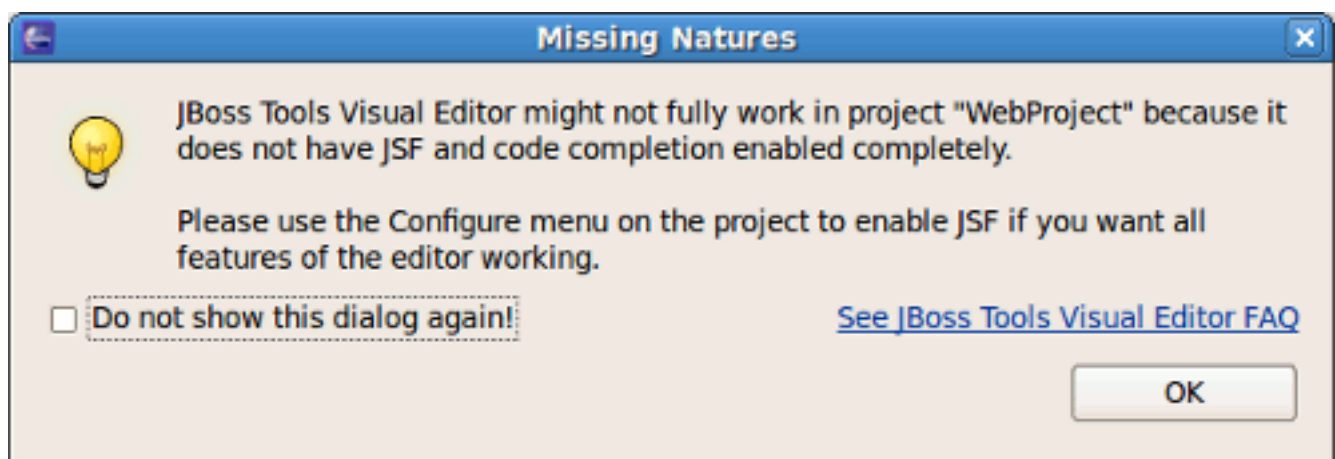


Figure 8.1. Missing Nature

Some functionality of Visual Editor may not work if a project doesn't have `org.jboss.tools.jsf.jsfnature` or `org.jboss.tools.jst.web.kb.kbnature` in .project configuration. To fix this problem and turn off the message box execute next steps:

1. Right mouse button click on a project in Package Explorer.
2. Select **Configure** → **Add JSF Capabilities** from the context menu.
3. Configure your project using Add JSF Capabilities wizard and press Finish.

If you are sure that your project does not need JSF capabilities, just disable this message box by checking `Do not show this dialog again!` checkbox.

8.3. I have an existing Seam 1.2.1 project. Can I migrate or import the project into a JBoss Developer Studio Seam project?

Use the following steps to manually transfer an existing Seam 1.2.1 project into a new JBoss Developer Studio Seam project:

- Create a Seam Web project to get the JBoss tools structure

Then from your Seam 1.2.1 seam-gen project start doing the following:

- Copy `src` to `src`
- Copy `view` to `Web content`
- Copy resources individual files to where they are in the seam web project etc.

8.4. I have an existing Struts or JSF project. Can I open the project in JBoss Developer Studio?

Yes. From main menu select **File** → **File** → **Import** → **Other** → **JSF Project (or Struts Project)** and follow wizards steps.

8.5. Can I import a WAR file?

Yes. Select **File** → **Import** → **Web** → **WAR file** then follow importing steps.

8.6. Is it possible to increase the performance of Eclipse after installing your product?

JBoss Developer Studio configures eclipse via the `jbdevstudio.ini` file to allocate extra memory, but if you for some reason need more memory then by default, you can manually make adjustments in this file. For example:

```
-vmargs -Xms128m -Xmx512m -XX:MaxPermSize=128m
```

8.7. How can I add my own tag library to the JBoss Tools Palette?

See the section on Adding Tag Libraries in the Visual Web Tools Guide.

8.8. How to get Code Assist for Seam specific resources in an externally generated project?

To get Code Assist for Seam specific resources in an externally generated project, you should enable Seam features in Project Preferences. Right click an imported project and navigate **Properties** → **Seam Settings**. Check *Seam support* box to enable all available Seam Settings.

8.9. How to import an example Seam project from jboss-eap directory?

To import an example Seam project from *jboss-eap* into your working directory, you should perform the following steps:

- Select **New** → **Other** → **Java Project from Existing Buildfile**
- Point to the `build.xml` file of any chosen project by clicking the **Browse** button
- Click the **Finish** button to open the project

As these seam examples are non WTP projects, next you should enable Seam support for them. To do that, right click the project and select **Properties** → **Seam Settings**.

8.10. Is a cross-platform project import possible for JBoss Developer Studio?

Yes. You can easily import created in Linux JSF, Struts or Seam project to Windows and vice versa.

To do the transferring JSF, Struts or Seam project, select **Menu** → **Import** → **General** → **Existing Projects into Workspace**.

Further Reading

- **Seam Dev Tools Reference Guide**

This guide helps you to understand what Seam is and how to install Seam plug-in into Eclipse. It tells you the necessary steps to start working with Seam Framework and assists in a simple Seam Project creation. Also you will learn how to create and run the CRUD Database Application with Seam as well as find out what Seam Editors Features and Seam Components are.

- **Visual Web Tools Reference Guide**

provides general orientation and an overview of visual web tools functionality. This guide discusses the following topics: editors, palette, web properties view, openOn, content assist, RichFaces support.

- **JBoss Server Manager Reference Guide**

This guide covers the basics of working with the JBoss server manager. You will read how to install runtimes and servers and quickly learn how to configure, start, stop the server and know how deployment and archiving process. You will find out how to manage installed JBoss Servers via JBoss AS Perspective. You will also read how to deploy modules onto the server.

- **jBPM Tools Reference Guide**

With jBPM Tools Reference Guide we'll help you to facilitate a cross-product learning and know how you can speed your development using special editors and visual designers. We'll also guide you through the steps on how to create a simple process and test it within jBPM jPDL perspective.

- **Hibernate Tools Reference Guide**

Throughout this guide you will learn how to install and use Hibernate Tools bath via Ant and through Eclipse. We'll supply you with the information on how to create mapping files, configuration file as well as a file for controlling reverse engineering by using specific wizards that Hibernate tooling provides. Also you will know about Code Generation and peculiarities of work within Hibernate Console Perspective.

- **ESB Editor Reference Guide**

This guide provides you with the information on ESB Editor and all necessary wizards for ESB files development.

- **JBoss Portal Tools Reference Guide**

The guide gives a detail look at how you can easily build a Portlet Web Application with JBoss Tools and deploy it onto JBoss Portal.

- **JBoss WS User Guide**

This guide gives you practical help on JBossWS usage. You will learn how to create a web service using JBossWS runtime, find out how to create a web service client from a WSDL document using JBoss WS and also see how to set your development environment.

- **Smooks Tools Reference Guide**

This guide is packed with useful and easy-to-understand information about graphical, configuration and source editor pages.

- **Drools Tools Reference Guide**

The guide help you to discover how to create a new Drools project, use debugging rules and work with different editors.

- **JMX Tools Reference Guide**

With the help of this guide you'll explore the best practices to follow when working with MBean Explorer, MBean Editor, Connections and etc.

- **Eclipse Guvnor Tools Reference Guide**

The purpose of this guide is to describe briefly the functionality present in the Eclipse Guvnor Tools (EGT) for Drools 5.

- **JSF Tools Tutorial**

This tutorial will describe how to deal with classic/old style of JSF development and how to create a simple JSF application.

- **JSF Tools Reference Guide**

From this guide you'll discover all peculiarities of work at a JSF project. You'll learn all shades that cover the process of project creation and take a closer look at the JSF configuration file. Also you'll get to know managed beans and how to work with them and find out, how to create and register a custom converter, custom validator and referenced beans in a JSF project.

- **Struts Tools Reference Guide**

In Struts Tools Reference Guide you will learn how to create and work with a new struts project. This guide also provides information about graphical editor for struts configuration files, tiles files, and struts validation files.

- **Struts Tools Tutorial**

This tutorial will describe the classical style of Struts development, and will step-by-step show you how to create a simple Struts application.