

# **JBoss Server Manager Reference Guide**

**Version: 3.3.0.M5**

---

---

---

<b>1. Quick Start with JBoss Server</b>	1
1.1. Key Features of JBoss Server	1
1.2. Starting JBoss Server	1
1.3. Stopping JBoss Server	2
1.4. Deploying an Application to a Server	3
1.5. Publishing to JBoss Server	7
<b>2. Runtimes and Servers in JBoss AS-Tools</b>	9
2.1. Runtimes	9
2.1.1. Installing a new runtime	9
2.1.2. Detecting an existing runtime	14
2.1.3. Duplicating an AS < 6.x runtime configuration	22
2.2. Servers	22
2.2.1. Creating a New Server	23
2.2.2. Creating Remote Servers	25
<b>3. JBoss Server Editor</b>	27
3.1. Server Editor - Overview Page	27
3.1.1. General Information	28
3.1.2. Management Login Credentials	28
3.1.3. Server Behavior	29
3.1.4. Publishing	30
3.1.5. Server Timeouts	30
3.1.6. Application Reload Behavior	30
3.1.7. Server State Detectors	30
3.1.8. Ports	31
3.1.9. Local Server Launch Configuration	32
3.1.10. Remote Server Launch Configuration	36
3.2. Server Editor - Deployment Page	36
<b>4. JBoss Perspective</b>	39
4.1. The Servers view	39
4.1.1. Servers view Toolbar	39
4.1.2. Servers view Structure	40
4.1.3. Drag-n-Drop to Servers view	51
4.2. Server Log View	52
4.3. Relevant Resources Links	53
<b>5. Projects</b>	55
5.1. Faceted Projects Overview	55
5.2. Adding Facets to a Project	55
5.3. Relevant Resources Links	61
<b>6. Deploying Modules</b>	63
6.1. Deploying on the Package Explorer	63
6.1.1. Deploying with Run On Server Wizard	63
6.2. Deploying with Servers View	64
6.3. Redeploying with Finger Touch	66
<b>7. Project Archives</b>	67

7.1. Project Archives View .....	67
7.1.1. Overview .....	67
7.1.2. Creating an Archive .....	67
7.1.3. Archive Actions .....	75
7.1.4. Publishing to Server .....	76
7.1.5. Relevant Resources Links .....	77
<b>8. TPTP Support .....</b>	<b>79</b>
8.1. TPTP Profiling .....	79
8.2. Relevant Resources Links .....	80

# Quick Start with JBoss Server

This chapter covers the basics of working with the JBoss Server.

## 1.1. Key Features of JBoss Server

The table below lists the main features included in JBoss Server:

**Table 1.1. Key Functionality for JBoss Server Adapter and Archive Tools**

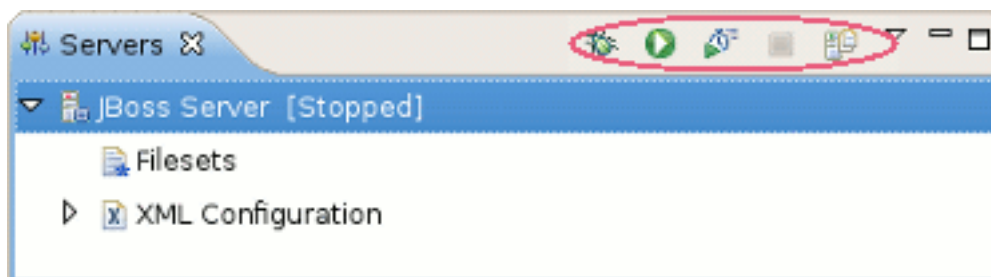
Feature	Benefit	Chapter
Runtimes and Servers	An in-depth look at the different ways to import and configure a JBoss installation to be used, along with the options available to you at creation time.	<a href="#">Runtimes and servers</a>
Views and Editors	This section will discuss the various views and editors that are accessible and related to JBoss AS Tools. This includes the primary view (Servers View), the server editor, and several integration views such as the MBean Explorer, the Console, and others.	<a href="#">Views and Editors</a>
Modules Deployment	A look into several ways to deploy a project, file, folder, or other type of module to a JBossTools server adapter.	<a href="#">Deploying modules</a>

If you already have imported or created a JBoss server and runtime, this chapter will show you the basics of how to start, stop, and publish to the server. Installing and customizing runtimes and servers will be covered covered in more detail in [Chapter 2, Runtimes and Servers in JBoss AS-Tools](#).

To start working with JBoss AS, you'll want to open the standard **Servers View** provided by WTP™. Start by selecting the menu **Window** → **Show View** → **Other** → **Server** → **Servers**.

## 1.2. Starting JBoss Server

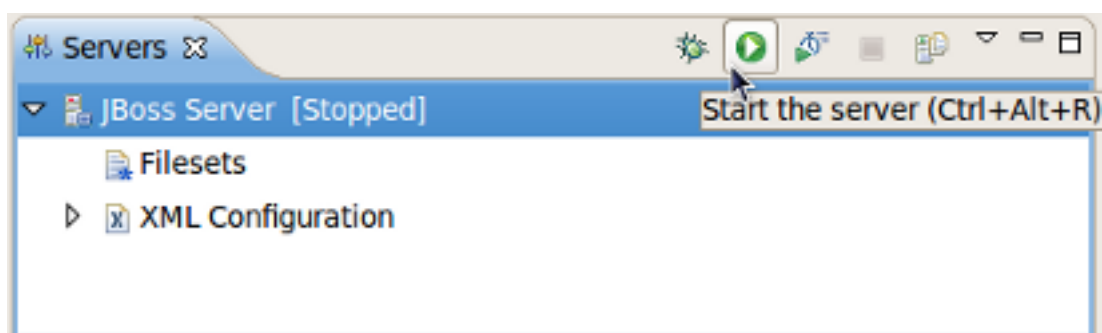
Starting JBoss Server™ is fairly straightforward. You can control the server state with the help of a special toolbar in the **Servers** view. This toolbar provides one-click access to controlling the server's state, and allows you to **start** the server in either regular or debug mode. You can also **stop** or **restart** the server, as well as **publish** to it.



**Figure 1.1. Servers Toolbar**

The first step to starting your server is to ensure your Servers View is opened. To open it, select **Window** → **Show View** → **Other** → **Server** → **Servers**.

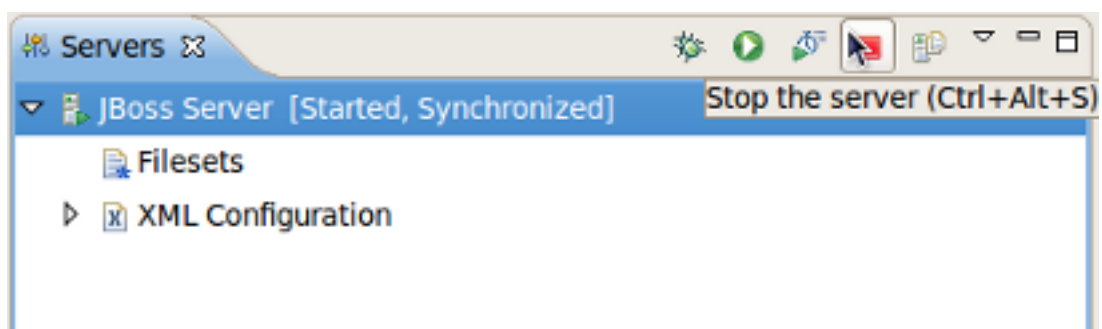
To launch the server click the start icon, consisting of a green circle with a white arrow inside, on the **Servers** view, or right click on the server name in the main section of the view and select **Start**. To start it in debug mode, you can select the **Debug** menu item, or click the debugging icon in the view's toolbar.



**Figure 1.2. Start JBoss Server**

## 1.3. Stopping JBoss Server

To stop the server, click the **Stop** icon in the **Servers** view, or right click the server name and select the **Stop** option.



**Figure 1.3. Stop JBoss Server**

When the server is stopped you will see **[Stopped]** next to the server's name in the Servers View. This decorator will inform you as to the server's running state, as well as its publish state, throughout your development session. On a workspace restart, however, the initial state is not shown. At this time, the state has not yet been initialized.

Learn more about the **Servers** view in [Section 4.1, “The Servers view”](#).

## 1.4. Deploying an Application to a Server

There are two times to deploy your application:

- While creating it
- After it already exists

When you create some types of JBoss Tools™ projects, such as Seam, JSF or Struts with the New Project or Import Project wizards, they will include the **Target Runtime** and **Target Server** sections. You can deploy the application through the appropriate configuration in these sections during project creation.

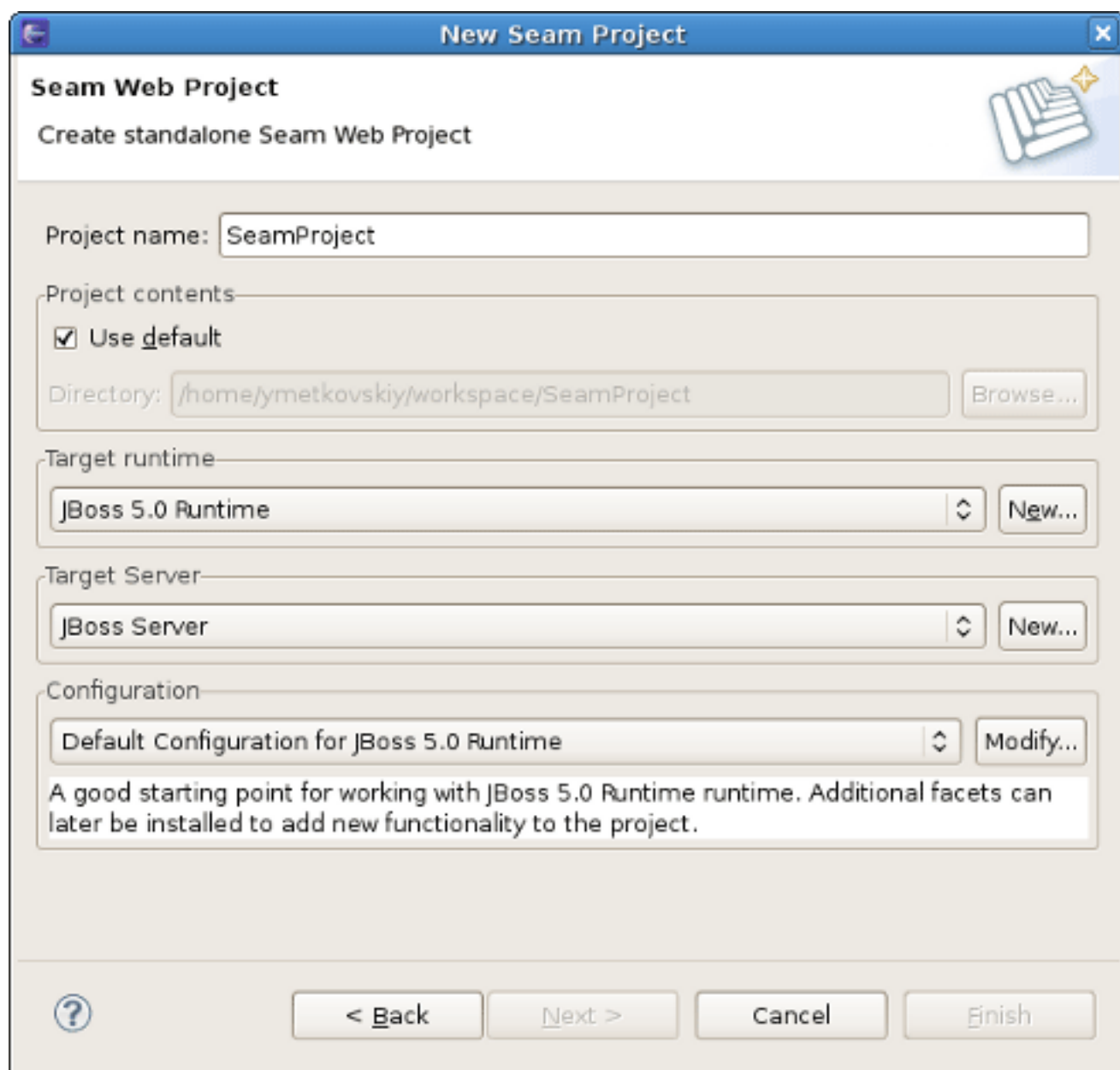
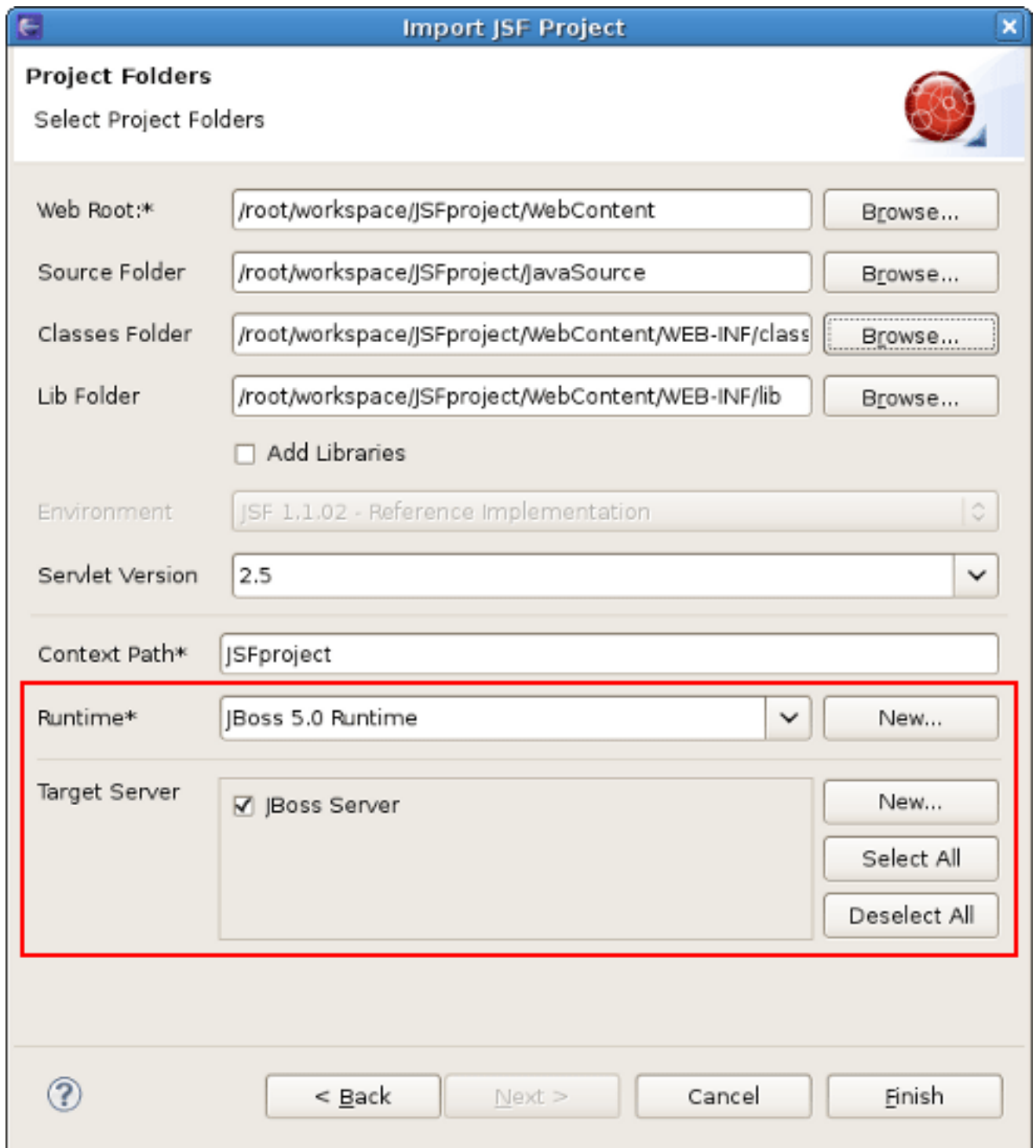


Figure 1.4. Runtime and Server Sections in the New Project Wizard

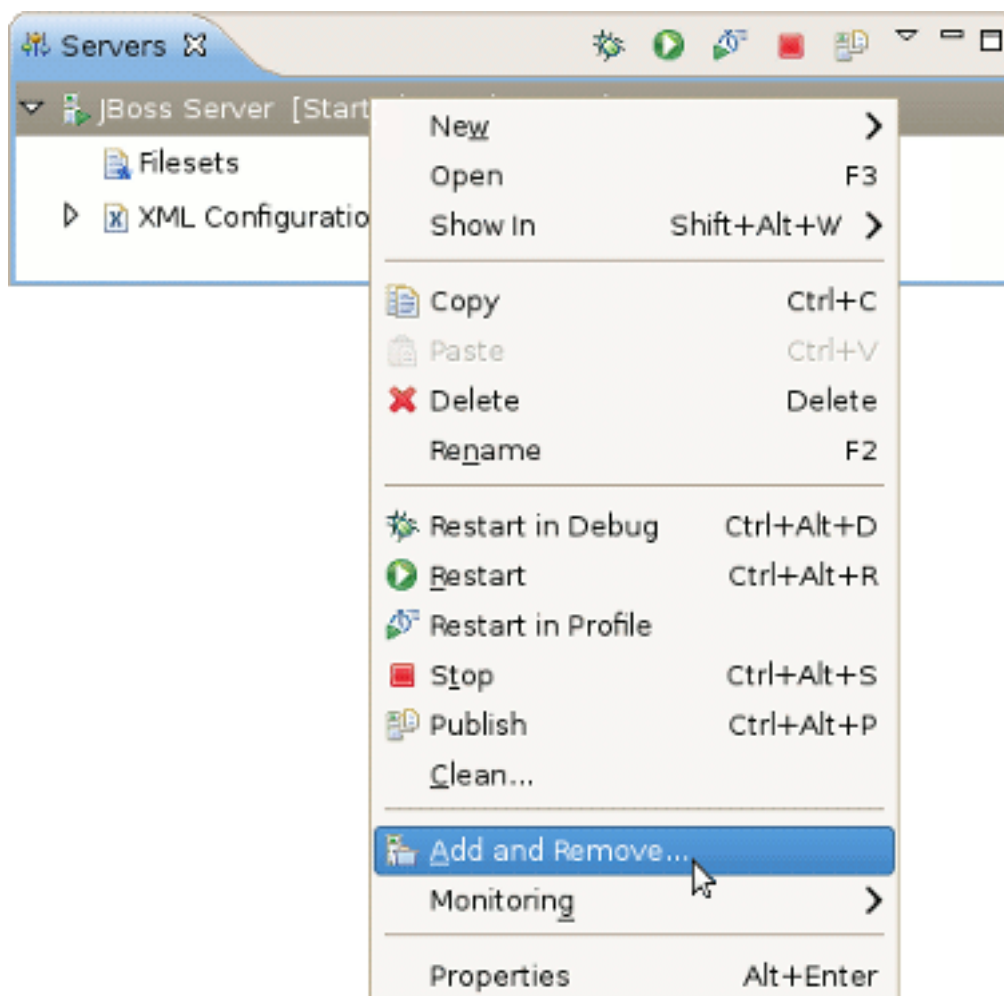




**Figure 1.5. Runtime and Server Sections in the Import Project Wizard**

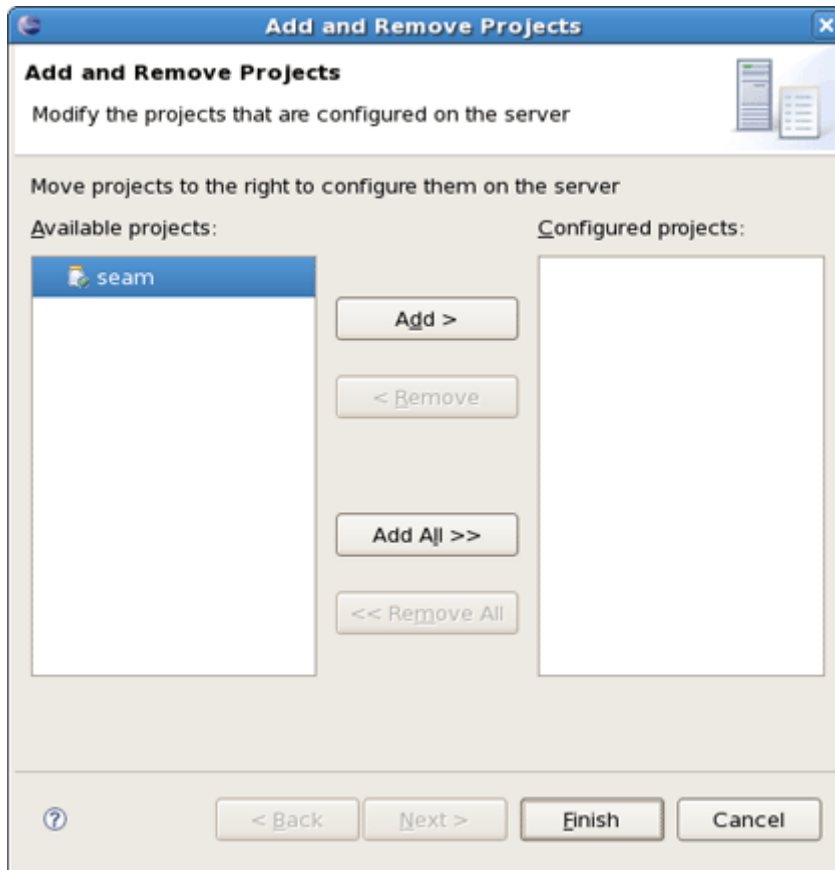
Other projects, such as those from WebTools™, allow you to target your project to a runtime for classpath evaluation purposes, but do not automatically deploy your application to the matching server adapter. (Remember, not all runtimes have an associated server. Some may be used solely for classpath evaluation.) Projects that behave this way include the Dynamic Web project, EJB Project, Utility Project, Enterprise Application Project, and others. You can, however, deploy these

existing applications to a server. To do so, first right-click the server you'd like to publish to in the **Servers** view. Then select **Add and Remove Projects** from the context menu.



**Figure 1.6. Add and Remove Projects From the Context Menu.**

If this application or module is not currently assigned to the selected server, it will be in the left-hand column, which lists modules available to be deployed. Clicking on your target module, and then on the **Add >** button will move the selected module to the right-hand module list. When you click finish, the module is now targeted to be deployed on the server. It may not publish immediately, however. The auto-publisher is disabled unless the server is already in started mode. You may need to force a publish event on the server, as described above, or simply wait until later. When you start your server, the modules will also be published.



**Figure 1.7. Modifying The Projects that are Configured on the Server**



### Note

It is now possible to deploy OSGI (Open Services Gateway initiative framework) projects to the JBoss Enterprise Application Platform 6 or JBoss Application Server 7.

## 1.5. Publishing to JBoss Server

The publishing of all the modules added to a Server is performed automatically when starting a Server.

Automatically publishing changes made to the workspace is enabled by default, allowing the workspace to remain in sync with the publish folder. However this autopublisher is only enabled when the server is in started mode. If you need to control when to publish the changes, just disable the automatic publish in the Server Editor (see [???](#)) and use the **Publish to Server**



button which will incrementally publish the workspace.

This section has provided some basic information that will allow you to use the common features provided by the JBoss server. However, JBoss server includes a great deal more functionality, which will be discussed in subsequent chapters.

# Runtimes and Servers in JBoss AS-Tools

In this chapter we will discuss how to install and configure JBoss runtimes and servers.

Runtimes in JBoss Tools provide key functionality for creating, running, and debugging J2EE applications. They provide classpath entries for projects, and are instrumental in starting, stopping, and publishing to the various server adapters. In their simplest form, though, a Runtime is nothing more than a representation of key information about a server configuration which can be used to provide classpath entries or other information important for the server lifecycles.

For AS7 / EAP6 related servers, the runtime consists of a server home, a JRE compatible with the server, and the configuration file used to describe the specific details about the server. For earlier versions of JBoss Application Server, the Runtime consists of the server home and JRE, as well as a configuration folder, and a configuration name.

In order to get started creating, running, and debugging J2EE applications, we should create our runtime and server instances.

## 2.1. Runtimes

In JBoss Tools, the main purpose of Server Runtimes is to point to a server installation somewhere on disk. In our case, this will be a JBoss installation. It can then be used for two primary purposes, as mentioned above:

- Providing classpath additions to WTP projects that require them.
- Acting as the backing data for a JBoss Server, which we'll look at later.

### 2.1.1. Installing a new runtime

You can install runtimes into Eclipse by selecting **Window** → **Preferences** menu and then selecting **Server** → **Runtime Environments** from the categories available on the left.

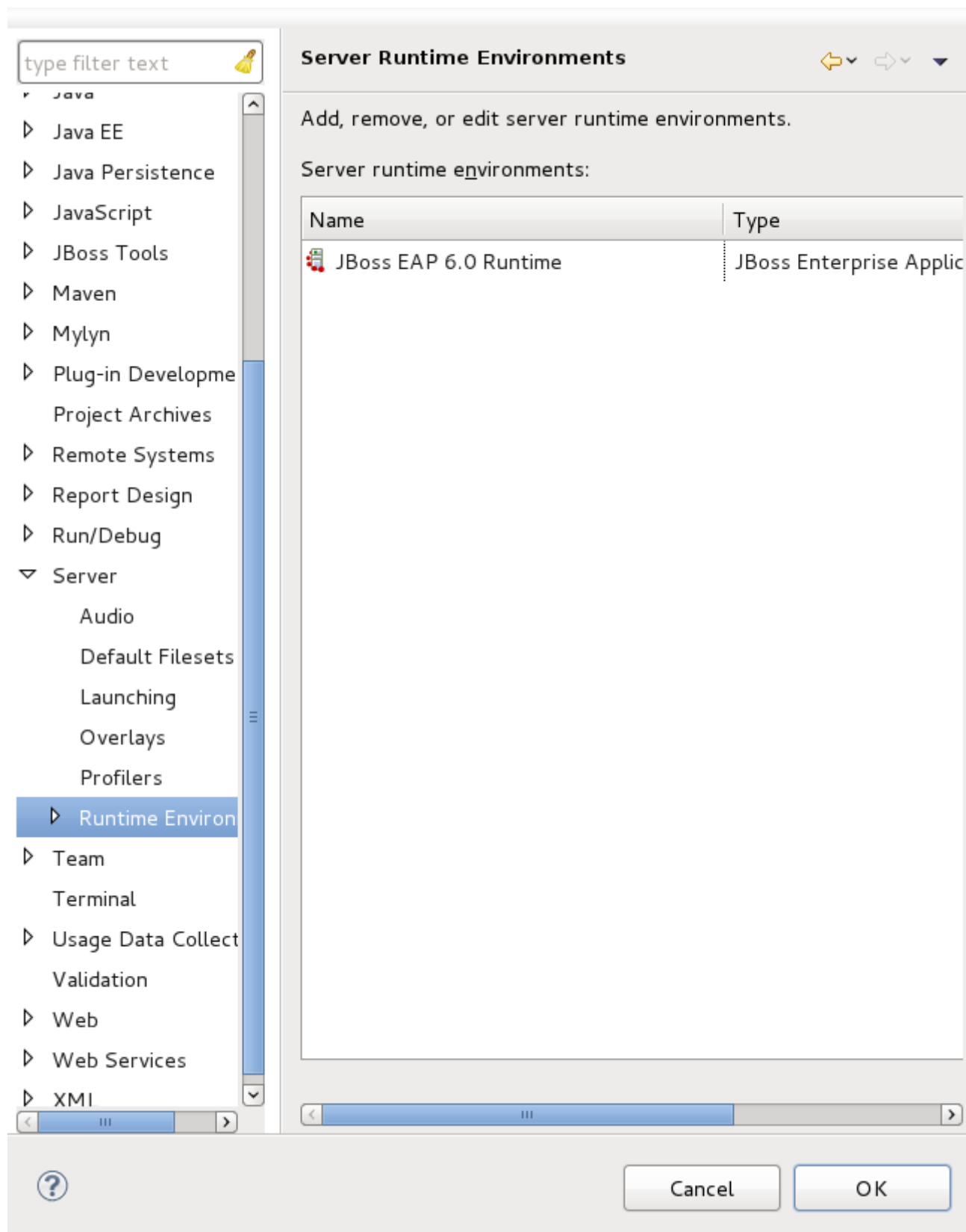
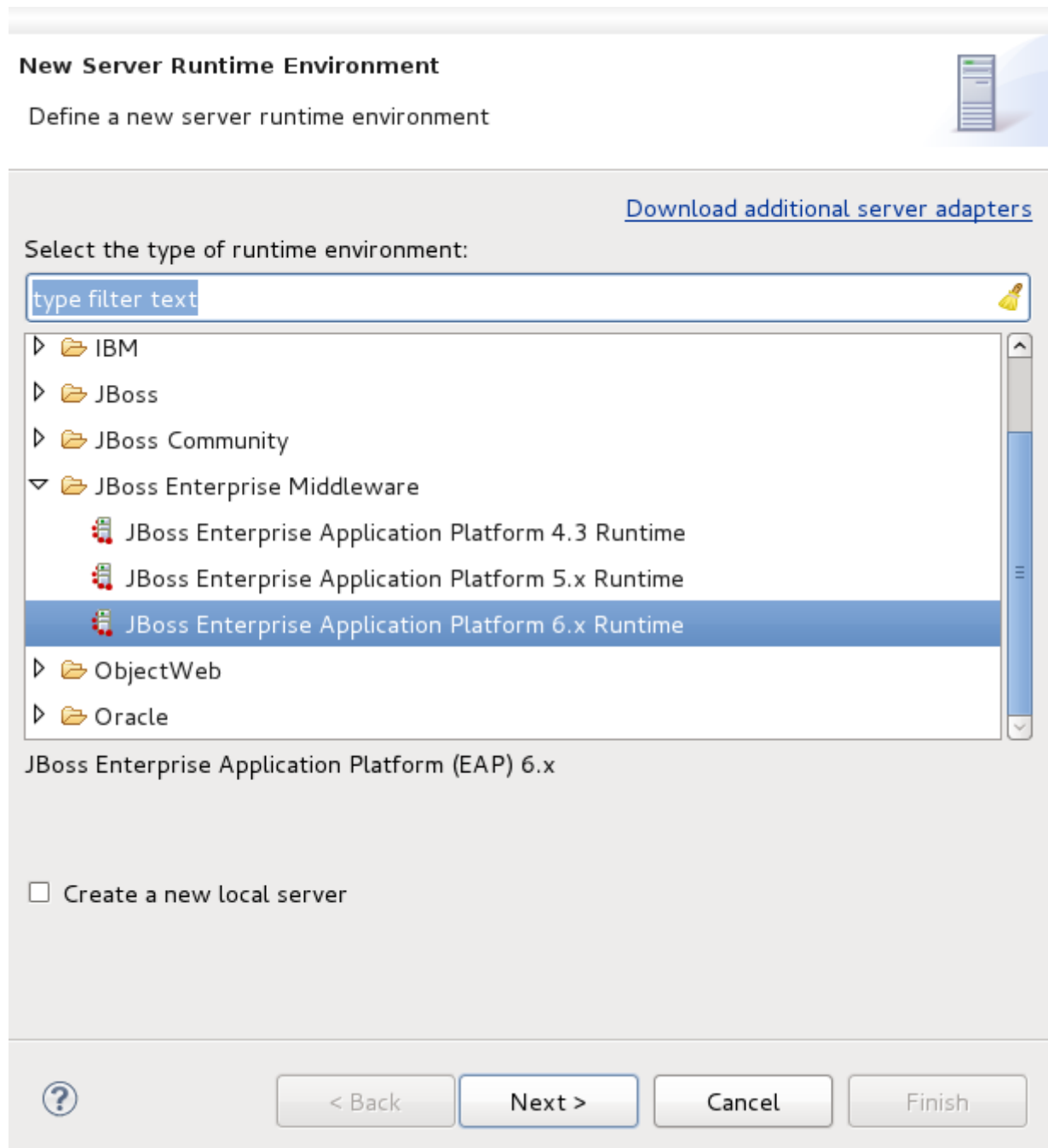


Figure 2.1. Installed Runtimes

From this preference page you can see all declared runtimes along with their types. Here, it is possible to edit or remove existing runtimes, as well as add a new one.

To create a JBoss runtime click the **Add** button and choose the appropriate type of runtime from the **JBoss Community** category.



**Figure 2.2. Adding a Runtime**



### Note:

Now there is a separation between .org servers (the **JBoss Community** category) and product server that comes with JBoss EAP in JBDS ( the **JBoss Enterprise Middleware** category).



### Note:


JBoss Tools provides server adapters for all versions of the JBoss Application Server and JBoss Enterprise Application Platform. We currently recommend you use a fully supported JBoss Enterprise 6.0 server adapter.

You will also note a Deploy-Only Runtime type. This type does not provide a classpath for WTP projects. It is used solely by it's server type for the purpose of setting up a simple deploy directory for users who do not wish to make use of starting, stopping, or debugging their projects inside Eclipse.



JBoss Runtime

JBoss Enterprise Application Platform 6.0



A JBoss Server runtime references a JBoss installation directory. It can be used to set up classpaths for projects which depend on this runtime, as well as by a "server" which will be able to start and stop instances of JBoss.

Name

Home Directory

JRE

Configuration file:

**Figure 2.3. Adding a JBoss EAP 6.0 Runtime**

The following tables describe all the available options for JBoss runtimes.

**Table 2.1. Runtime Options for AS 3.2 to AS 6.0, EAP 4.3 to EAP 5.2**

Name	Description
Name	The name of a new Runtime for a chosen server. We suggest that you do not leave the default value for this field. It is better to give descriptive names that will help to distinguish one runtime from another.
Home directory	The path to a directory where the runtime is installed.
JRE	A compatible Java Runtime Environment which can be used for launching or classpath resolution.

Name	Description
Directory	The path to a directory where the configurations are installed.
Configuration	The list of configurations (all, default, minimal), which is updated as soon as you browse to a valid runtime installation folder.

**Table 2.2. Runtime Options for JBoss AS 7.x / EAP 6.x**

Name	Description
Name	The name of a new Runtime for a chosen server.
Home directory	The path to a directory where the runtime is installed.
JRE	A compatible Java Runtime Environment.
Configuration File	A path to the configuration file you are targeting. This path must be relative to the standalone folder.

As a result of having each runtime represent a specific configuration rather than the server installation as a whole, it is very likely you will create several different runtimes to test each of your configurations. With this in mind, it becomes important to ensure your runtimes, and later your servers, are given descriptive names that help you to remember which is which.

Click the **Finish** button to see your new runtime in the list.

**Note:**

For the most part, changes to runtimes will be persisted, and all dependent servers will be updated according to those changes. However, you should be aware that if multiple servers depend on the same runtime, modifications to that runtime will change several servers, not just one.

As a word of warning, renaming your runtime will cause all servers targeting that runtime to lose their reference. This may manifest itself by the server being unable to start, or publish. The most visible way to verify that your server is in a consistent state is to double left-click on the server to open the Server Editor, and look in the **Overview** → **Runtime Environment**. If the combo has no selected item, your server is in an inconsistent state. To fix this, simply select the newly renamed runtime from the combo box, and save the editor.

### 2.1.2. Detecting an existing runtime

JBoss Tools features the ability to search, detect and add existing JBoss server runtimes installed on your system. If you don't have an existing runtime you can download one through the **Download** option or [Section 2.1.1, “Installing a new runtime”](#) will guide you through the creation process. To begin searching for your existing JBoss runtime select **Window** → **Preferences** → **JBoss Tools** → **JBoss Runtimes**.

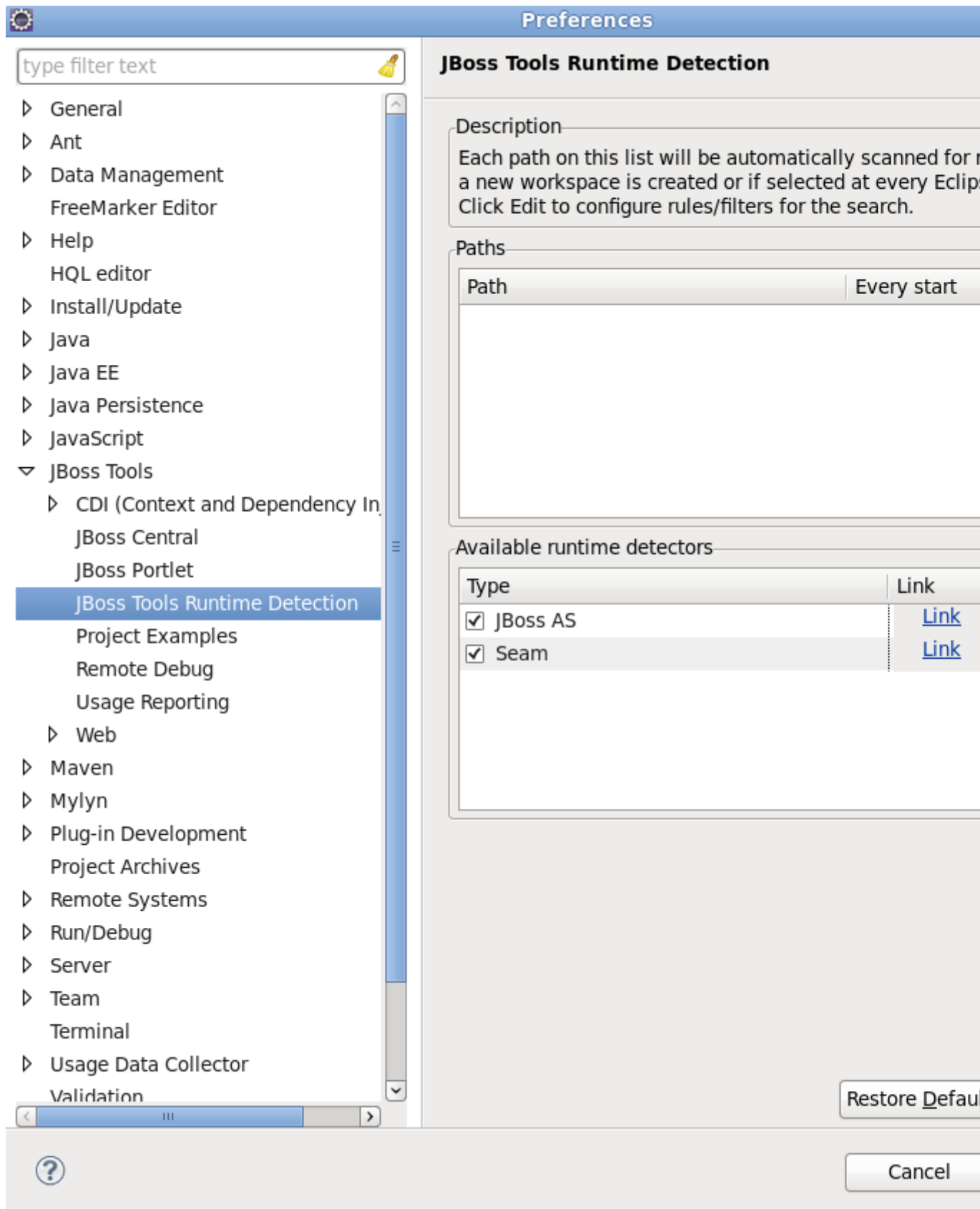


Figure 2.4. Preference page for JBoss Runtimes

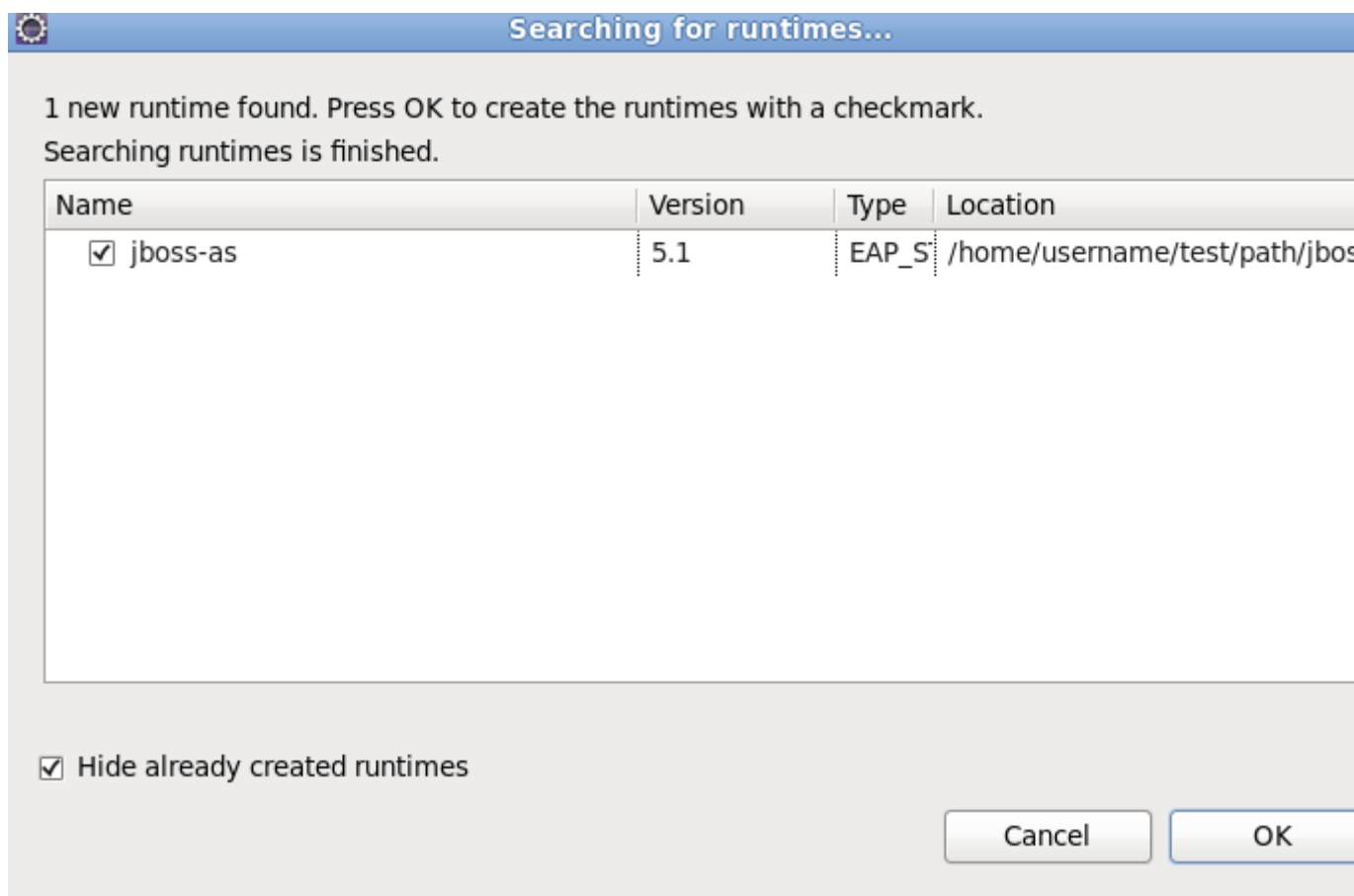
The JBoss Tools Runtimes preference page is split into two different sections. One section defines **Paths** to be searched for installed server runtimes, the other section defines the runtime detectors available when the paths in the previous section are checked.

The **Add** button in the **Paths** section opens a file system browser window. Select the directory you wish to have recursively searched for JBoss runtimes. The directory will be searched and all found servers will be displayed as a list in the **Searching for runtimes** dialog. From the returned list, choose the runtimes you wish to make available by clicking the box beside each runtime and clicking the **OK** button.



### Note

If you are using a full JBoss Developer Studio installation, runtime detection now recognizes the ESB runtime distributed as part of the JBoss Service-Oriented Architecture Platform runtimes during a scan. If you are using the a-la-carte installation options provided by JBoss Tools, you may or may not benefit from this enhancement depending on what plugins you have installed.



**Figure 2.5. JBoss Runtime search results**

The path you searched is now added to a list in the **JBoss Tools Runtime Detection** dialog **Paths** section. All the paths in this section will be automatically searched when a new workspace is created. This is convenient in that it allows you to quickly create a new workspace with your runtime settings much easier to replicate. If you wish for a path to be searched on each and every startup, then check the checkbox in the **Every start** column associated with it.

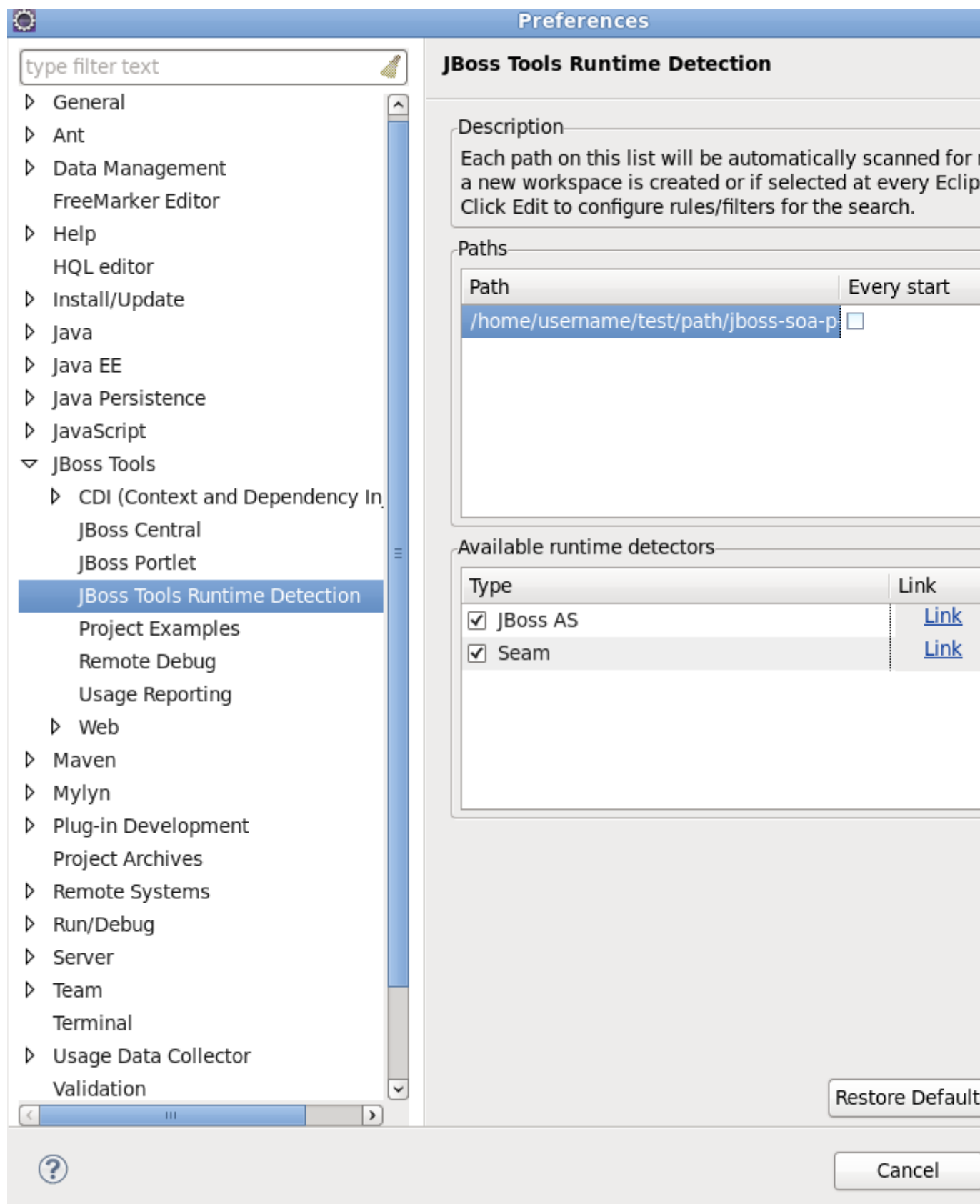


Figure 2.6. JBoss Runtime search results

If you don't have a runtime already downloaded, you can download a free community application server through the **Download** button.



### Important

No official support is available for community application servers (this includes enterprise customers using JBoss Developer Studio).

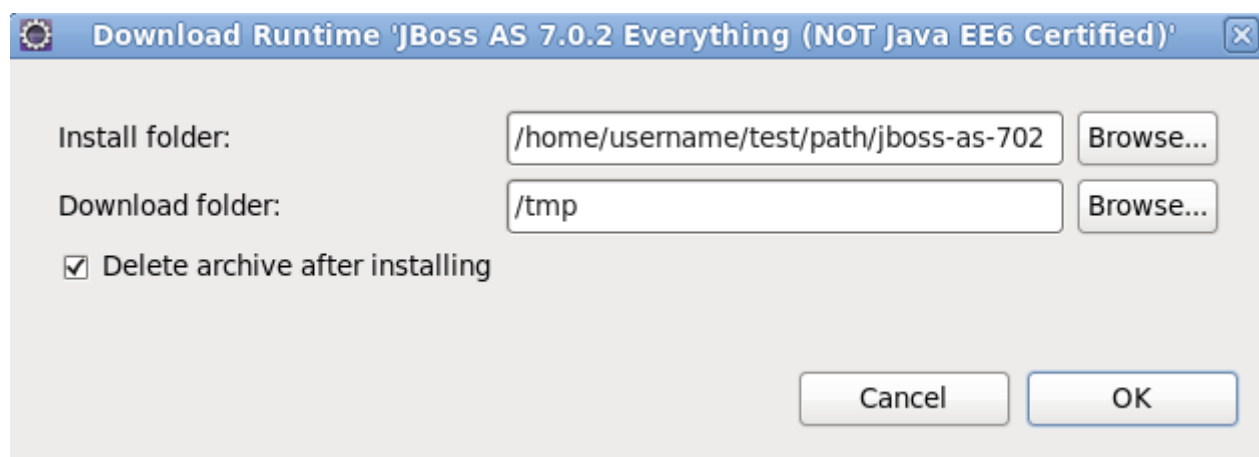
Clicking on the **Download** button will display a new screen of available runtimes that can be downloaded. Highlight the server you wish to download and install, and click the **OK** button.

Runtimes		
Name	ID	Version
JBoss AS 4.2.3	org.jboss.tools.runtime.core.as.423	4.2.3
JBoss AS 6.1.0	org.jboss.tools.runtime.core.as.610	6.1.0.Final
JBoss AS 7.0.1 Everything (NOT Java EE6 Certified)	org.jboss.tools.runtime.core.as.701	7.0.1.Final
JBoss AS 7.0.2 Everything (NOT Java EE6 Certified)	org.jboss.tools.runtime.core.as.702	7.0.2.Final
JBoss AS 7.1.0 Certified Java EE 6 Full Profile	org.jboss.tools.runtime.core.as.710	7.1.0.Final
JBoss AS 7.1.1 Certified Java EE 6 Full Profile	org.jboss.tools.runtime.core.as.711	7.1.1.Final
JBoss Seam 2.0.2.SP1	org.jboss.tools.runtime.core.seam.2	2.0.2.SP1
JBoss Seam 2.2.2.Final	org.jboss.tools.runtime.core.seam.2	2.2.2.Final

Cancel
OK

**Figure 2.7. JBoss Runtime search results**

A new dialog will appear asking you to specify an **Install folder** and **Download folder**; the option to **Delete archive after installing** is checked by default. Once you have specified the two paths above, click the **OK** button and the server will begin downloading.



**Figure 2.8. JBoss Runtime search results**

Once the server has been downloaded and installed, you will notice that the path to the new server now appears in the **Paths** section of the **JBoss Tools Runtime Detection** dialog.



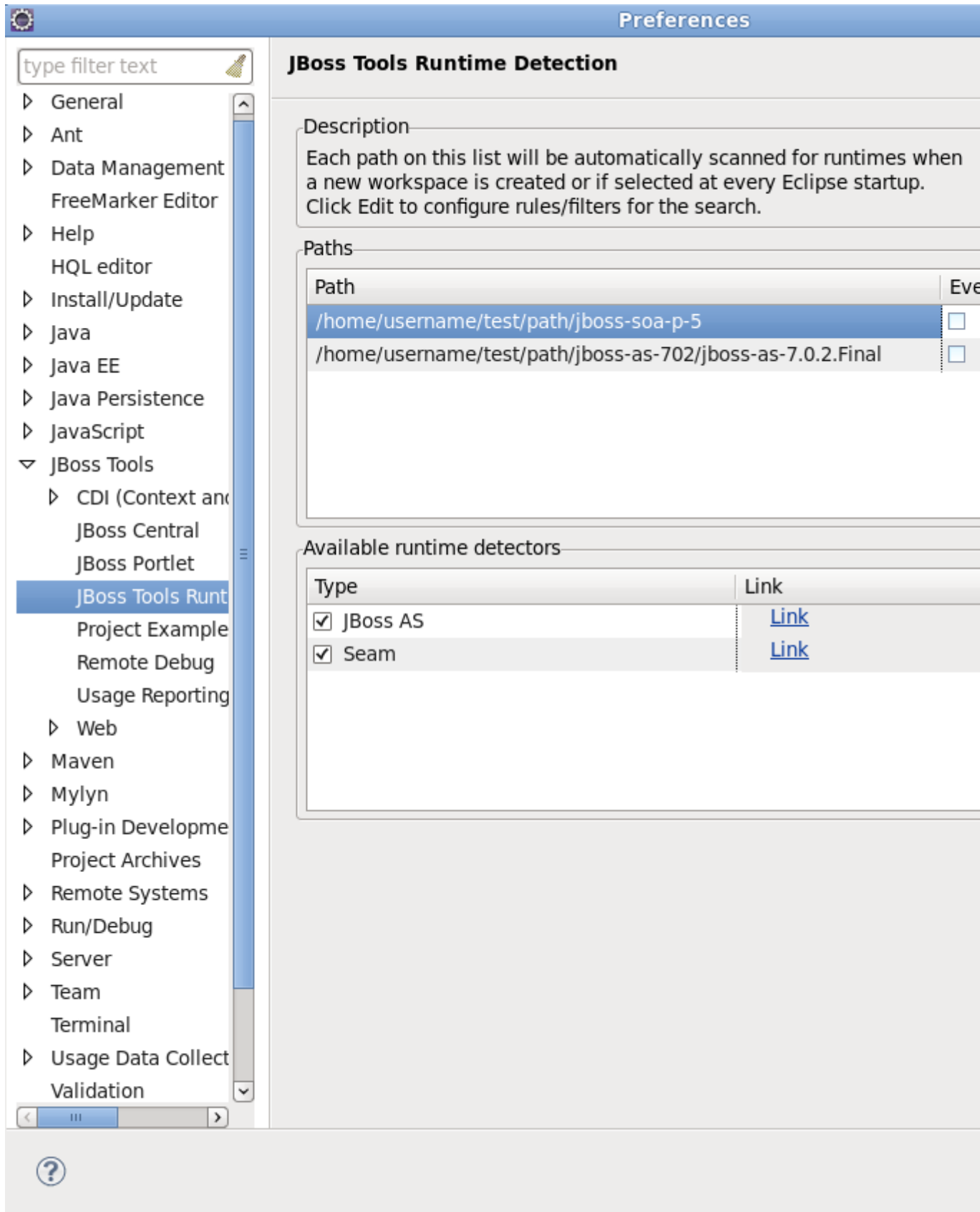
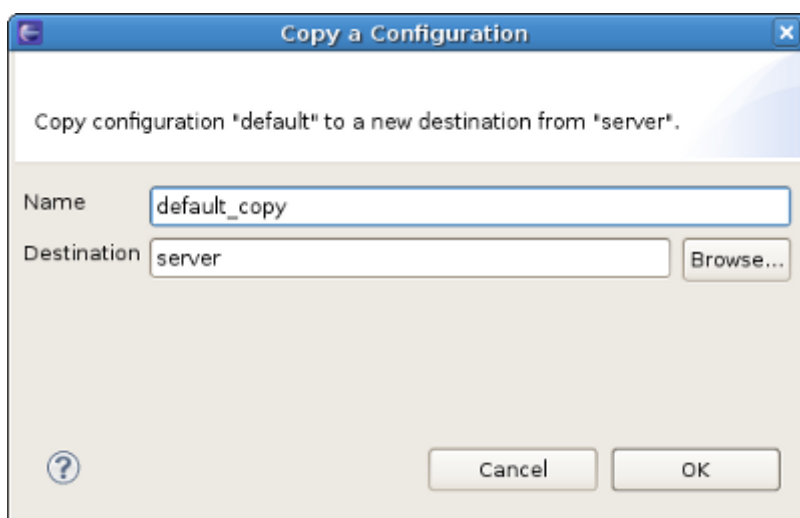


Figure 2.9. JBoss Runtime search results

### 2.1.3. Duplicating an AS < 6.x runtime configuration

While installing a new runtime you can create a clone configuration from an existing one. To do this you should perform all the steps in [Section 2.1.1, "Installing a new runtime"](#), but do not click the **Finish** button in the New Server Runtime Environment dialog.

Make sure that you browse to a valid runtime folder and can see the list of configurations (all, default, minimal) in the Configuration section. Then choose an appropriate Configuration from the list and click the **Copy** button. You will then see the following dialog.



**Figure 2.10. Copy the existing configuration**

First, give the new clone configuration a name. Then, click the **Browse** button and select the new location for your configuration to live, or leave as it is if you want it to be located together with other runtime configurations.

Click the **OK** button and you should see the next wizard with the newly copied configuration.

**Figure 2.11. Runtime with copied configuration**

Click the **Finish** button and you will see your new runtime in the list.

You can also change the configuration of existing runtime to a copied one in the same way by selecting **Window** → **Preferences** → **Server** → **Runtime Environments** and clicking the **Edit** button.

## 2.2. Servers

WTP servers are Eclipse-representations of a back end server installation. They are used to start or stop servers, deploy to servers, or debug code that will run on the server. They also keep track

of the modules (JARs, WARs, etc) you deploy to the server, and allow you to undeploy those modules (see [Section 6.1.1, “Deploying with Run On Server Wizard”](#)).

Servers can be started or stopped via the **Servers** view in your workbench. They are usually backed by a runtime object representing that server's configuration details.

### 2.2.1. Creating a New Server

There are many ways to get to the new server wizard. One way is to select **File** → **New** → **Other...** → **Server**. This should show the wizard like below.

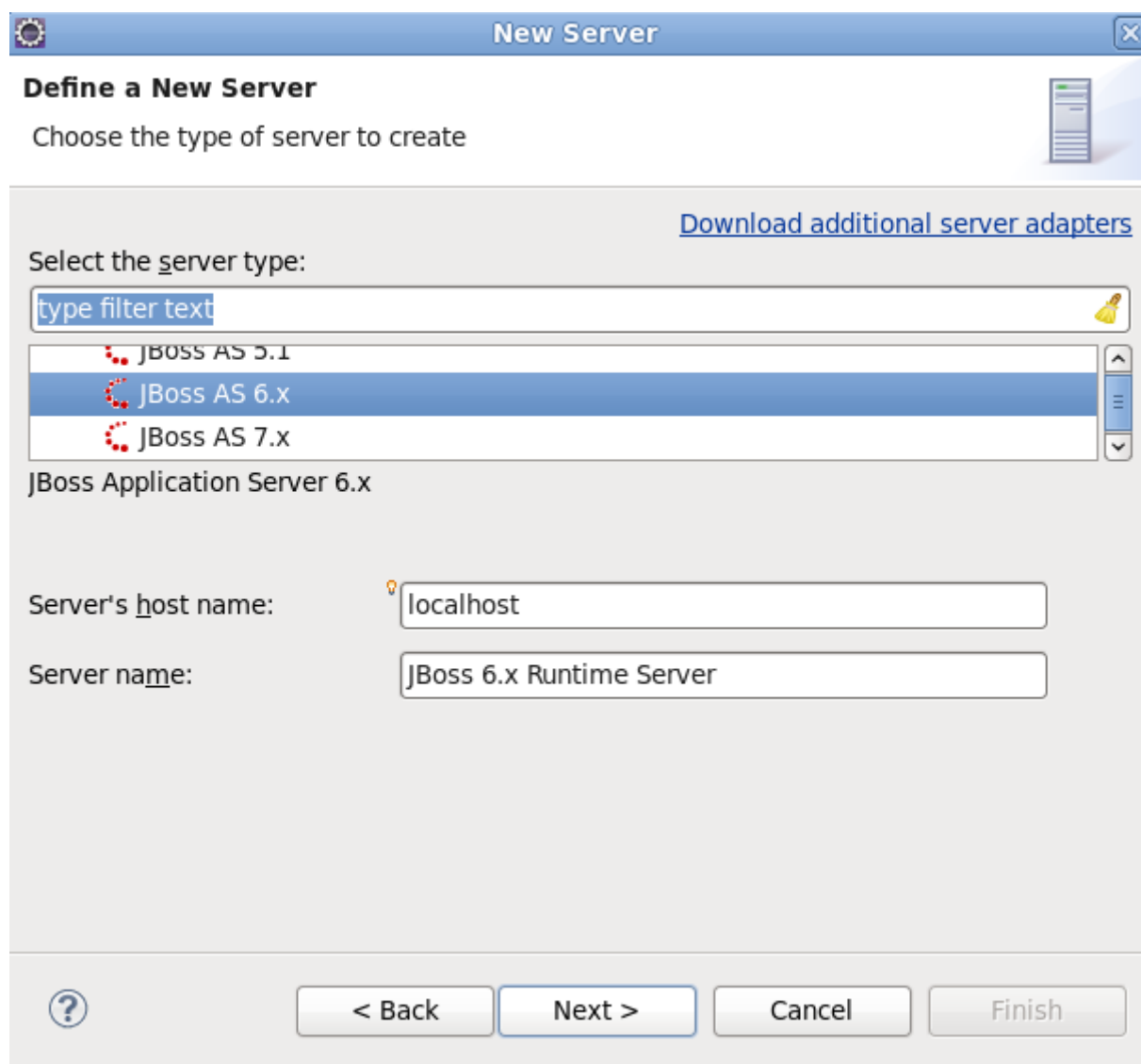
#### Figure 2.12. Adding a JBoss Server

A server object keeps track of the command line arguments for starting or stopping a server process. The runtime keeps track of the location of the installation and is used to help generate these command line arguments.

The **New server wizard** allows you to name the server via the **Server name** field, or you can use a generated default name.

You can select one of the already-created runtimes from the **Server runtime environment** combo box. If there is no runtime that matches your needs, press the **Add...** link nearby to bring up the wizard for creating a new runtime (see [Figure 2.3, “Adding a JBoss EAP 6.0 Runtime”](#)). To make changes to an existing runtime, go to server preferences by pressing the **Configure runtime environments...** link. Be aware that any changes you make here may change other servers dependent on that runtime. We recommend creating new runtimes for each different scenario.

If the server you want to create does not have any installed runtime yet, the combobox and the links are absent.



**Figure 2.13. Installed Server Runtime Environments**

If there is no runtime when creating your server, the next page of the wizard will be identical to the runtime page mentioned in [the previous section](#). This page guides you through creating a runtime for your server.

After either targeting your server to an existing runtime, or creating a new runtime, the final page of the wizard presents a summary of the selected options, giving you a chance to verify that you have selected the appropriate runtime.

**Figure 2.14. Installed Server Runtime Environments**

You will also see several other options here. The first option reads **Server is externally managed**. **Assume server is started**. This option indicates that starting the server should actually take no

action, launch no java command, and assume that the user is managing the server lifecycle on his own.

The second option reads **Listen on all interfaces to allow remote connections**. This option is most often required when using the tools to control a remote server. Effectively, this adds the `-b 0.0.0.0` flags to the server's launch command, which allows your server to respond to requests on all hostnames.

The third option reads **Expose your management port as the server's hostname**. This option helps ensure that attempts to connect to your server over the management port actually succeed. If your server adapter does not have this option selected, requests to run management commands may be rejected. This should not be a problem for any server adapters representing a local server instance. Locally, JBoss passes around a filesystem token for management authorizations. However, if your server adapter is representing a remote server, failure to expose the management ports may lead to an inability to communicate with the remote instance.

Not all of these options will show for all servers. JBoss 7.0.0 and 7.0.1 for example do not support the `-b` binding flag, while versions `< 7.0` do not have the same remote management options. Be aware that this list changes based on the context of what server types you are creating.

Click the **Finish** button to complete the process of the server creation.

Now that we have created our runtimes and servers, we can explore the services and tools provided by the JBoss Server Manager.



### Important

It is not recommended to run two servers on the same host, at the same time as you may experience a conflict in ports. If a server is already running on the same host a warning will appear indicating this and will ask if you wish to **Set the server adapter to 'started', but do not launch** or **Launch a new instance anyway**

**Figure 2.15. Conflicting Ports Warning**

## 2.2.2. Creating Remote Servers

You may also create a server adapter to control a remote server instance. These types of server adapters are slightly different to set up, but generally follow the same process. the primary difference here is that on the last page of the New Server Wizard, you will also need to select **Remote System Deployment** option in the combo box at the bottom of the page as shown below..

**Figure 2.16. Remote System Server Creation for AS < 7.0**

### Figure 2.17. Remote System Server Creation for AS 7.x / EAP 6.x

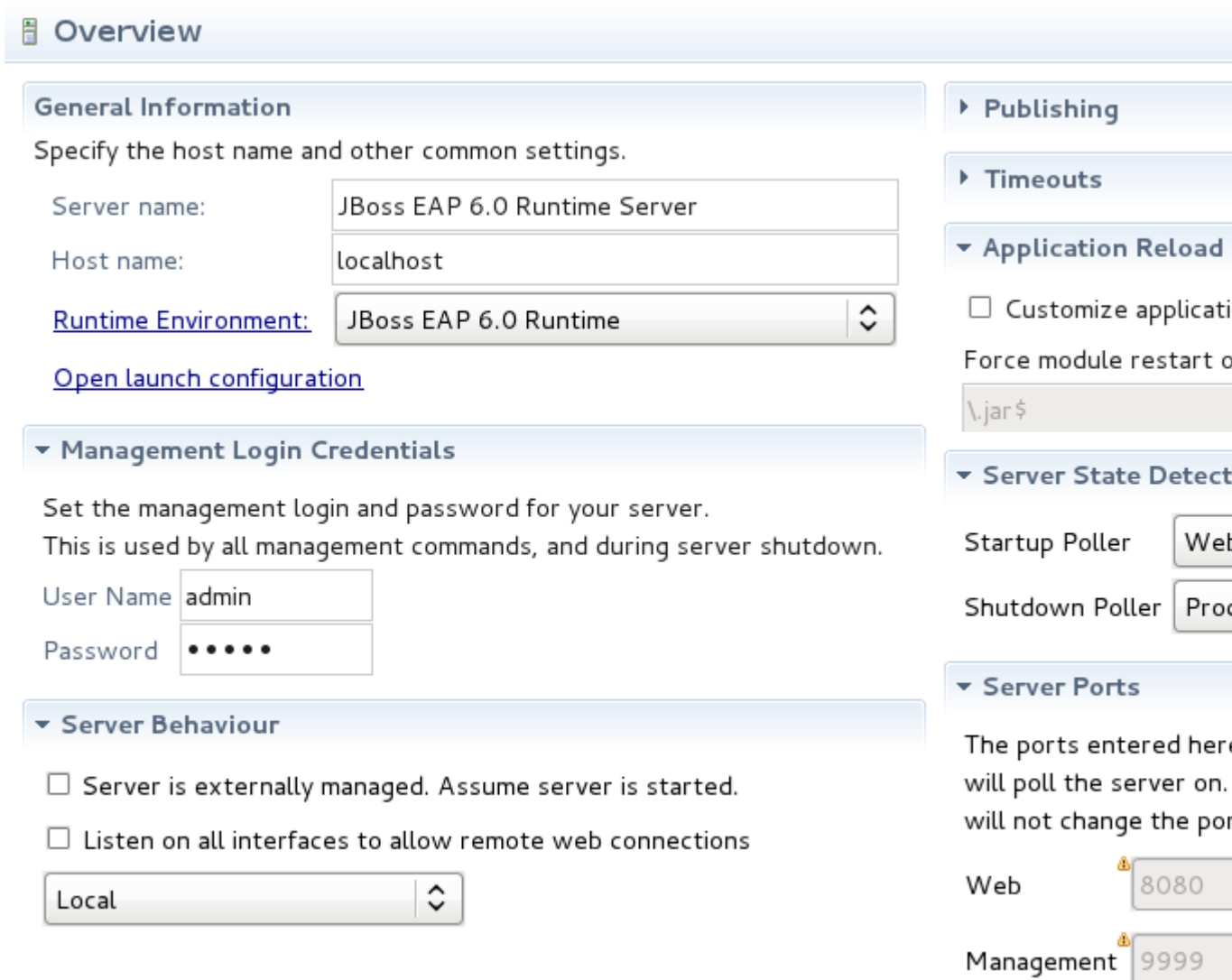
Once you've chosen the Remote Server Deployment behavior type, your next step is to choose a remote host. If you do not have a remote host defined, you'll want to click **New Host...**, which will walk you through the process of setting up a remote host. After that, you must fill in the remote details for your server configuration. This includes filling in your remote server's home directory and the appropriate configuration details depending on your chosen server version. As with local runtimes, earlier versions of JBoss require a configuration folder and a configuration name, while recent versions require only a configuration file.

# JBoss Server Editor

This chapter describes how to manage and change the settings and behaviour of an existing JBoss AS-Tools Server Adapter. The server editor is the primary vehicle for users to modify the behavior and settings for their server, including what launch arguments to use, how to discover if the server is started, and many others.

## 3.1. Server Editor - Overview Page

By double-clicking on any server, an window will appear allowing you to edit the servers settings.



**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

Local

**Publishing**

**Timeouts**

**Application Reload**

☐ Customize application

Force module restart o

\\jar\$

**Server State Detect**

Startup Poller Web

Shutdown Poller Proc

**Server Ports**

The ports entered here will poll the server on. will not change the por

Web 8080

Management 9999

**Figure 3.1. Server Editor - Overview Page**

The Overview page is an important entry-point to modify settings for your server. This page has the following sections.

- General Information
- Management Login Credentials
- Server Behavior
- Publishing
- Timeouts
- Application Reload Behavior
- Server State Detector
- Server Ports

### 3.1.1. General Information

#### Figure 3.2. General Information

The **General Information** section provides you with your server name, host name, a link to your runtime, and the ability to open and customize your launch configurations. In general, changing any of these values should be safe, but it is not recommended to rename your server or modify your runtime's name.

You can modify the settings of your runtime by clicking on the **Runtime Environment** hyperlink. This will open the Edit Runtime wizard. This wizard looks identical to the New Runtime Wizard, and allows you to modify any settings on your runtime. Be aware, however, that changes made here may modify behavior of any other servers that target the same runtime. Caution is advised.

By clicking the **Open Launch Configuration** hyperlink, you can see and modify the launch behavior of your server adapter. In this way, you can customize the start arguments. More details will be provided later on.

### 3.1.2. Management Login Credentials

#### Figure 3.3. Management Login Credentials

To connect to your server's management services, you'll need to provide your credentials in this text field. For your protection, the password field is obscured. The password field is also stored in Eclipse Secure Storage, so you'll never have to worry about the security of your credentials.

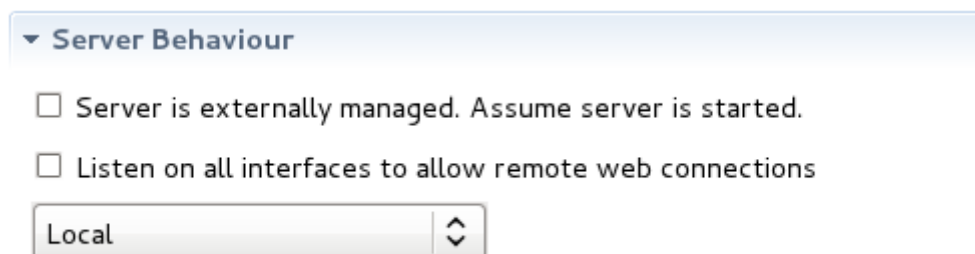
These credentials are used to access your server's management services. For earlier versions of JBoss AS or JBoss EAP, this would be a connection to JMX. For recent versions, it's credentials for the management service provided by that runtime. If you're noticing strange behavior, or if the tools are not recognizing that a server is started, it may be caused by incorrect management credentials. In general, though, such errors will be displayed to the user to allow a chance to fix the problem.



### 3.1.3. Server Behavior

The **Server Behaviour** settings tab allows you to set how tool interaction with the server should be undertaken.

When you created the server, if you selected that it was a local server then you will notice that the option **Server is externally managed. Assume server is started** is unchecked and the combo-box displays **Local**.



▼ **Server Behaviour**

☐ Server is externally managed. Assume server is started.

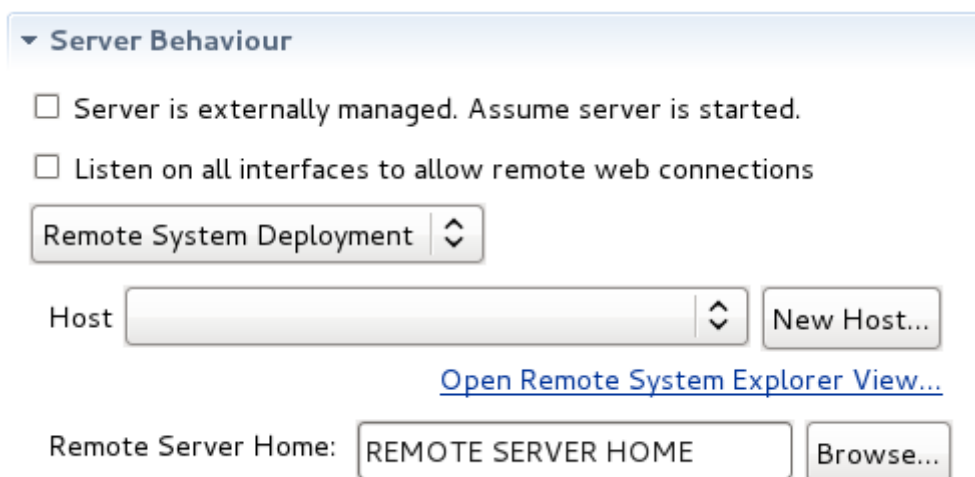
☐ Listen on all interfaces to allow remote web connections

Local

**Figure 3.4. Server Behaviour - Local**

If you created a remote server then you will see that the combo-box displays **Remote System Deployment**. Also populated will be the **Host** and **Remote Server Home** settings.

You are also able to change a servers behaviour from **Local** to **Remote System Deployment** through this settings tab. In doing so you will see that the **Host** is not set by default, but the other fields contain default values.



▼ **Server Behaviour**

☐ Server is externally managed. Assume server is started.

☐ Listen on all interfaces to allow remote web connections

Remote System Deployment

Host

New Host...

[Open Remote System Explorer View...](#)

Remote Server Home: REMOTE SERVER HOME Browse...

**Figure 3.5. Server Behaviour - Remote**

You can select the **Listen on all interfaces to allow remote web connections** when using JBoss Application Server 3 to 7 or JBoss Enterprise Application Platform 4 to 6. This option will force the server to launch with the option `-b 0.0.0.0`. This option will change the host address

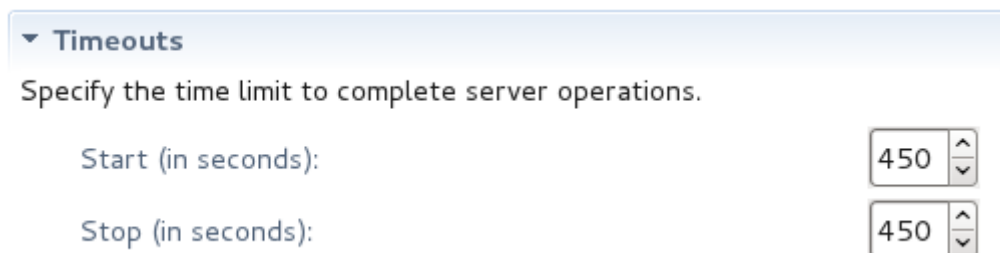
to 0.0.0.0, useful for testing web applications on your local machine. JMX commands and web browser activities will still use the host set in the **General Information** section.

### 3.1.4. Publishing

#### Figure 3.6. Publishing

The **Publishing** section allows you to specify the autopublish behavior for your server. There are three options available. The first option is to disable autopublishing entirely. The second is to fire the autopublisher only when a workspace resource has changed. The final option would be to only fire the autopublisher if a build event has occurred. Depending on your personal development style, any of these options may fit your wishes.

### 3.1.5. Server Timeouts



▼ Timeouts

Specify the time limit to complete server operations.

Start (in seconds): 450

Stop (in seconds): 450

#### Figure 3.7. Timeouts

The **Timeouts** section allows you to specify a time limit for the server to complete operations within. If an operation does not start or fails to finish before the times you specify, that operation will be cancelled to avoid server failure.

### 3.1.6. Application Reload Behavior

#### Figure 3.8. Application Reload Behavior

This section of the Server settings allows you to customize the reload behavior of your application, depending on server and module changes. Typically, a module will be forcefully restarted if an inner library such as a .jar file is modified. However, some users may prefer to reload the module even if any .class file changes. This section takes a regex pattern which matches against modified files during a publish. When finding a match, the module will be forcefully restarted after the publish is complete.

### 3.1.7. Server State Detectors

#### Figure 3.9. Server State Detectors

Upon trying to start or stop a server, the tooling attempts to recognize the state the server is in. During these operations, the tools will periodically poll the server to verify it's state. This section allows you to select which method the tools should use to verify the state of the server. The most common and safest type is the Web Port Poller, which simply pings the web port on your server's host to see if a server responds. (More details on "Ports" can be seen in the next section.)

Startup and Shutdown pollers can be selected independently. Possible polling mechanisms include polling JMX, the JBoss Management service, or a simple ping on the Web Port. Another option called the Timeout Poller waits for a specified duration and then simply declares the start or stop operation a success, performing no actual verification at all. Finally, the shutdown poller may also be set to the "Process Terminated" poller. This poller checks to see if the server process is still alive, and, when terminated, will set the server state to Stopped.



### Important

The "Process Terminated" polling mechanism should not be used for servers operating with a Remote System Explorer behavior.

## 3.1.8. Ports

**Figure 3.10. Ports**

The server adapter tries to automatically detect the ports it needs for integrating with a JBoss Server by default. It does this typically by parsing through configuration files, searching for some XPath, and using the values found as the chosen port. The **Server Ports** section in the **Server editor** provides fields to customize the ports that the tooling will use to communicate with the server. Above, you see a name, a text box with an integer value representing the port, a checkbox toggling automatic detection, and a hyperlink to dive deeper into the discovery details. Sometimes, it is necessary to override this automatic detection if you are using a custom configuration. As shown above, the tools need to know how to accurately discover the web port and the management port. Recent servers allow for more accurate detection by also discovering the port offset setting.



### A Note on Port Offsets

The JBoss Application Servers allow for a port offset to be declared. Essentially, a port offset is an integer to be added to all ports on startup. So if your server is set to have a web port 8080, but you provide a port offset of 200, the web port (and all other ports) will be 200 higher. This allows you to start multiple servers on the same machine without port conflicts.

The simplest way to ensure that the tooling is pinging or communicating on the correct port is to uncheck **Detect Automatically**. This sets the text field as editable, and you can simply type the correct port. To look deeper at the actual settings of how the port is discovered, or to modify this behavior, click the **Configure...** link to bring up the wizard for adjusting the settings for the ports.

### Figure 3.11. Server Ports Preferences

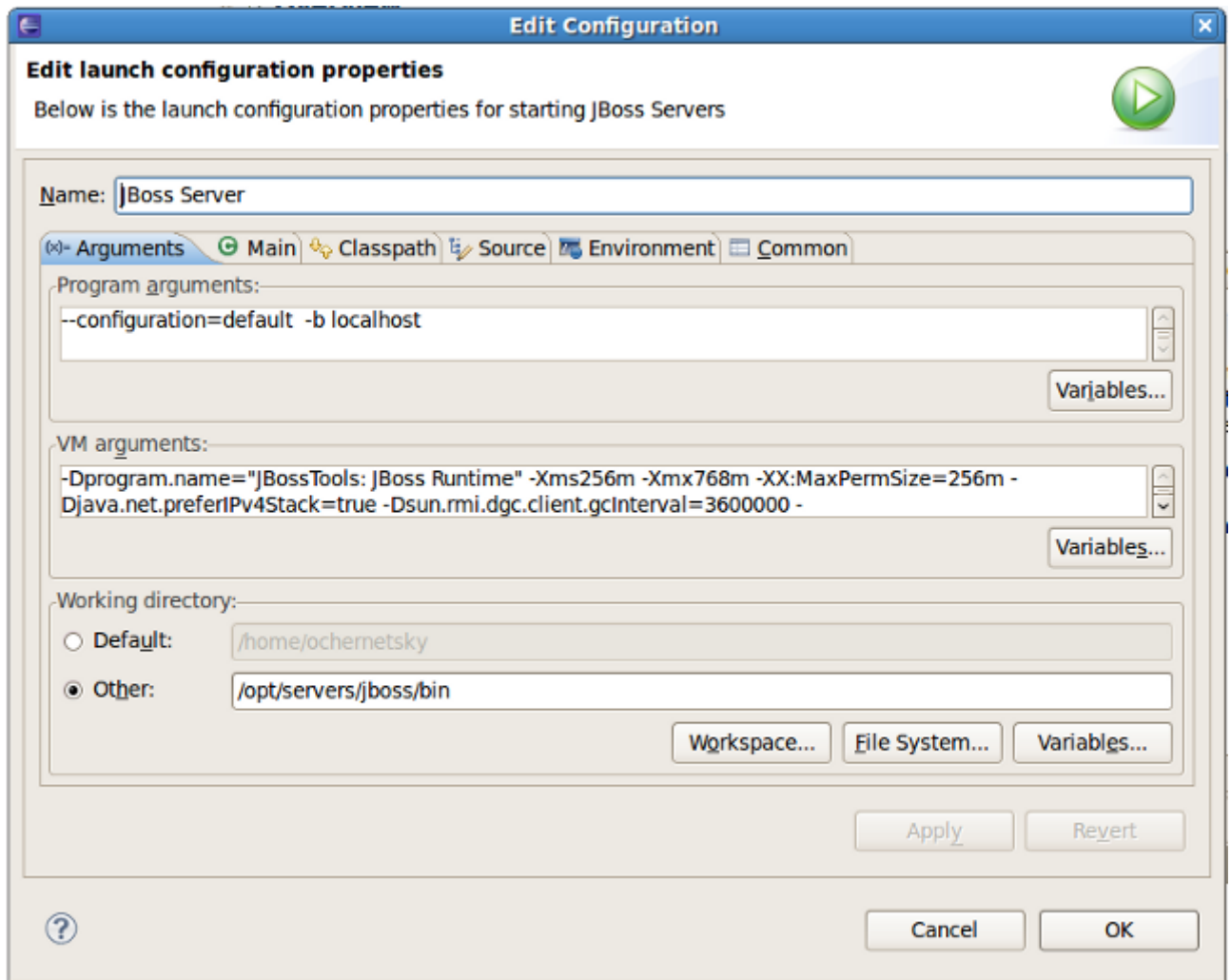
Click the **Edit XPath** button for the chosen port to configure its XPath's values.

### Figure 3.12. XPath Pattern for a Server Port

In this dialog, you can customize the XPath and an optional attribute name to discover the proper port string. The dialog provides a **preview** button to see the results that match the XPath. Ideally, you'll want to craft an XPath that matches only one result. You can limit the files to be searched by use of a fileset pattern. This will help limit other files from matching the result set, and ensuring an accurate detection of the proper port.

## 3.1.9. Local Server Launch Configuration

The **Server editor** window also allows you to modify the server's launch configuration. The settings are available by clicking the the **Open launch configuration** link in the **General Information** section of the editor. The resulting window provides tabs for setting command line arguments, main, classpaths and other things that are relevant to launching the server.



**Figure 3.13. Launch Configuration Properties**

The first tab shows the JBoss server arguments



#### Note:

Please note that some of the values in the Launch Configurations for JBoss Servers are strictly enforced in order to avoid inconsistencies between server's and their configured runtime.

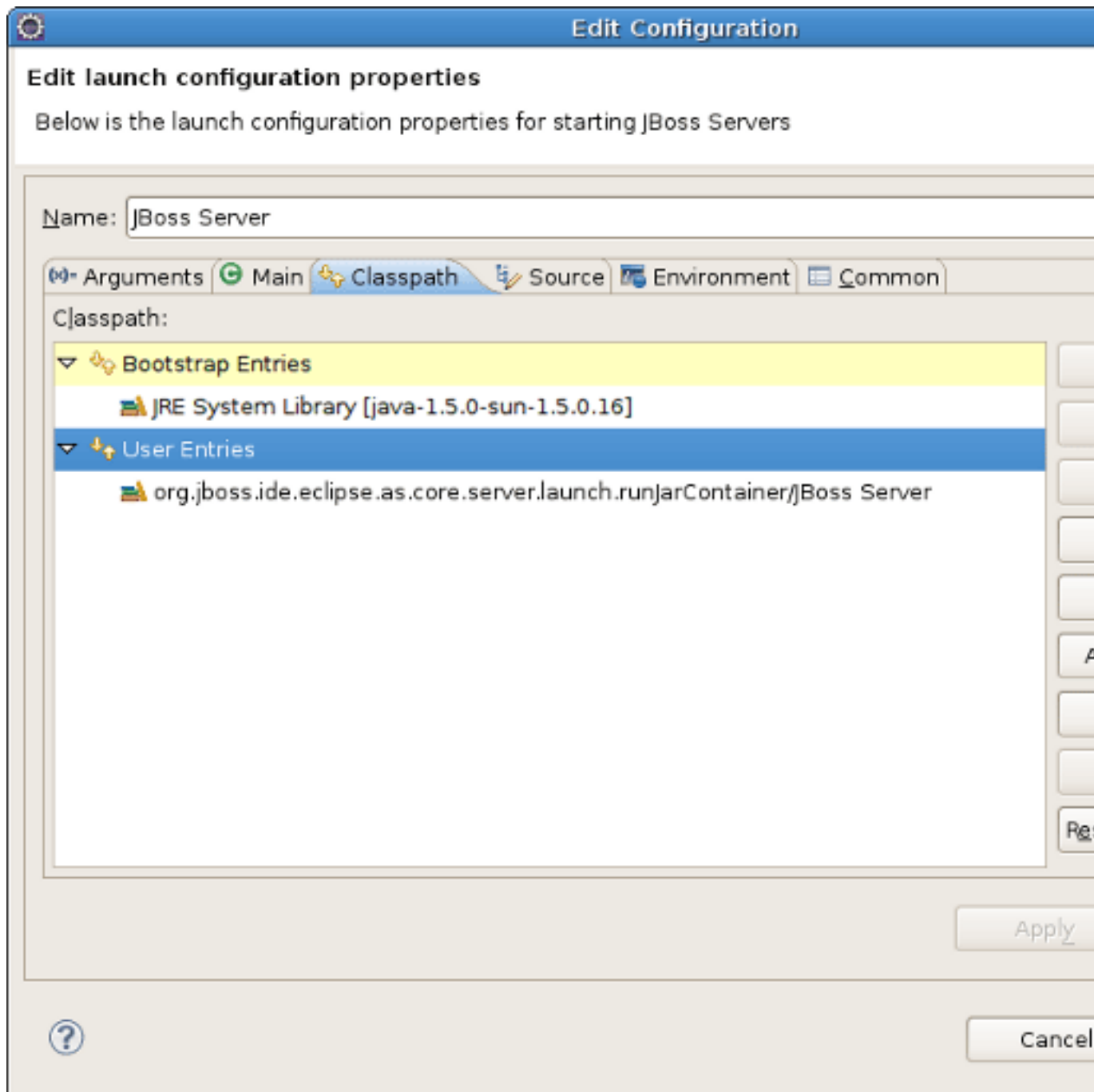
For example, if you change the launch configuration program arguments to "-c myConfig" but do not change the targeted runtime configuration, then your program arguments will be ignored. The configuration of the server runtime "wins" so to speak. This ensures consistency. Therefore, if you change the location of the runtime, your launch configurations will automatically pick that up.

Non-critical or custom arguments can be passed in or overridden with no problem at all. It is only arguments that otherwise conflict with the data in the runtime that will not be respected.

On the second tab you find the main class used for launching JBoss AS (the default is `org.jboss.Main`). This value can be changed if necessary.

Until JBoss Tools 3.0.0.GA the servers classpath was read only, but that caused problems for users wanting to add their own JARs in the startup classpath. That is relevant if you need to patch the server, add a custom charset or other tweaks that require early access to the classpath.

Now all servers have a custom 'server runtime classpath container', which is there by default and point to the default JARs in JBoss. You can now adjust the classpath. Then just make sure this container is there if you want the classpath to be picked up.



**Figure 3.14. Server Classpaths**

If for some reason you have a launch configuration without this container, the **Restore Default Entries** button should add it properly. Also, the **Restore Default Entries** button will remove any extra entries you added yourself.

### 3.1.10. Remote Server Launch Configuration

#### Figure 3.15. Remote Launch Configuration Properties

For servers representing a remote server, the launch configuration window allows for a complete customization of the remote launch startup and shutdown commands if the user chooses. The top half of the window represents the commands to be issued during server startup; the bottom is used for shutting down the server. These commands are executed as a remote command over SSH. Both sections include a checkbox marked **Automatically Calculate**. If this box is checked, the commands are not editable, but all critical arguments are generated using information not from the runtime, but rather from the **Server Behavior** section of the **Server Editor**, where your remote server home and configurations are declared.



#### Important

If you opt to uncheck the **Automatically Calculate** checkbox, there will be no overwriting of any critical paths at all. The user will have 100% control over this, and the tools will no longer attempt to helpfully overwrite important paths as they do for the local servers.

## 3.2. Server Editor - Deployment Page

Using **Deployment tab** you configure local deployment settings.



**Deployment**

Local Deployment

**▼ Default Settings**

This section sets where any non-customized module will be deployed to.  
 The temporary deploy folder should be on the same file-system as the deploy folder.  
 This will ensure safe and complete file copies.  
 Customizations can be made in the table below on a per-module basis.  
 Blank columns will use the default values.  
 Changes should *\*not\** be made to this page while the server is running.

☒ Use workspace metadata (does not modify JBoss deploy folder)  
☐ Use the JBoss deploy folder  
☐ Use a custom deploy folder

Deploy Directory    
 Temporary Deploy Directory

☐ Deploy projects as compressed archives

Module	Deploy Directory	Temporary Deploy Directory
--------	------------------	----------------------------

Overview | **Deployment**

**Figure 3.16. Deployment tab**

Using the group of radio buttons in the **Default Settings** section a user can set where the application will be deployed to. By default it is deployed to a folder inside the user's workspace metadata, located inside `[workspaceDirectory]\.metadata\.plugins`. If you would like the application to be deployed to your JBoss server's deploy folder, select the **Use the JBoss deploy folder** option. The option to specify a custom deploy folder is also available.

You should also see a checkbox available labeled **Deploy projects as compressed archives**. This option forces all of your publish operations to result in a zipped archive output file. Rather than an exploded directory war, for example, you would end up with a zipped .war file.

**Figure 3.17. Deployment tab - Per-Module customizations**

As shown above, the bottom section of this editor page allows customizations on a per-module basis. For example, if you want one of your modules to be published to a custom directory, while the rest live happily inside the server's deploy folder, you can now override this setting here. If you'd like to change the output file name from the assumed `MyProject.war` to the `OtherName.war`, this is also possible.

To begin editing these values, simply click on the cell you wish to edit. All paths should be absolute, or, relative to the default deploy directory as chosen in the top section of the editor page. Be aware that when publishing, we require the temporary folder to be on the same filesystem as the final deploy location. If this constraint is not satisfied, publish will often fail. Do not forget to save the editor before these results can take effect.



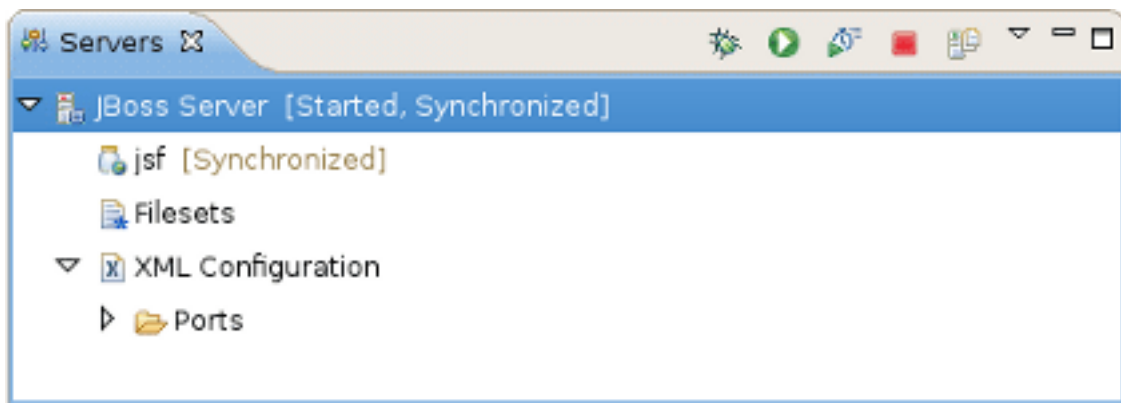
# JBoss Perspective

This chapter describes how to manage installed JBoss Servers™ via the **JBoss Perspective**. The Servers view will primarily be discussed.

## 4.1. The Servers view

The **Servers** view is built on the Common Navigator Framework allowing extensions and is using label decorators that make the UI compact enough without losing the vital information.

Let's have a detailed look at the **Servers** view and its constituent components.



**Figure 4.1. The Servers view**

### 4.1.1. Servers view Toolbar

In the right top corner of the **Servers** view there is a special toolbar which provides a quick access for starting a server (in the debug mode, run mode, or profile mode), restarting a server, stopping a server and a publishing to a server.



**Figure 4.2. The Servers view Toolbar**

In order to debug your applications or EJB's that are deployed to the server, the server must be started in debug mode. By starting the server in debug mode, Eclipse will allow you to set breakpoints on code in your workspace and step through the code.

The **Publish to the server** button will republish any modules where it has determined that the workspace is out of synchronization with the server. It will attempt to do an incremental publish if the module in question is capable of doing one.

4.1.2. Servers view Structure

The **Servers** view displays all defined servers as well as their current status (that is whether they are started or stopped) in square brackets next to the server name.

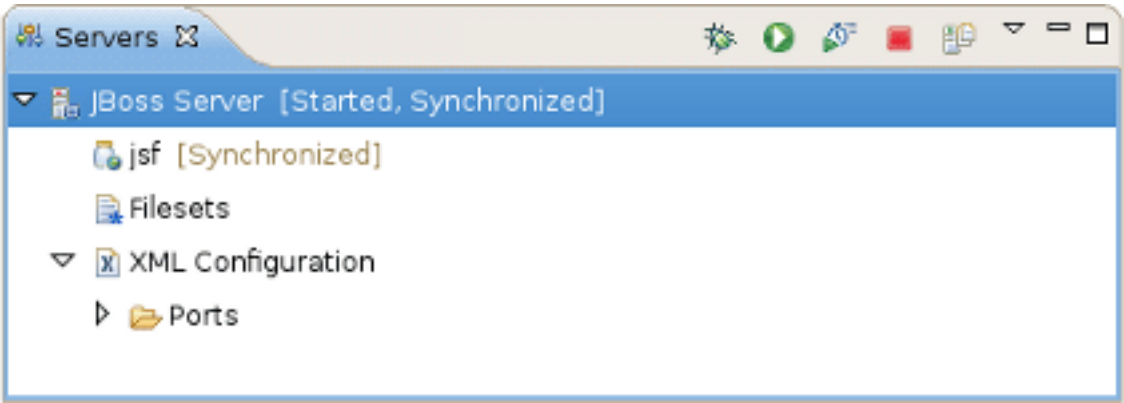


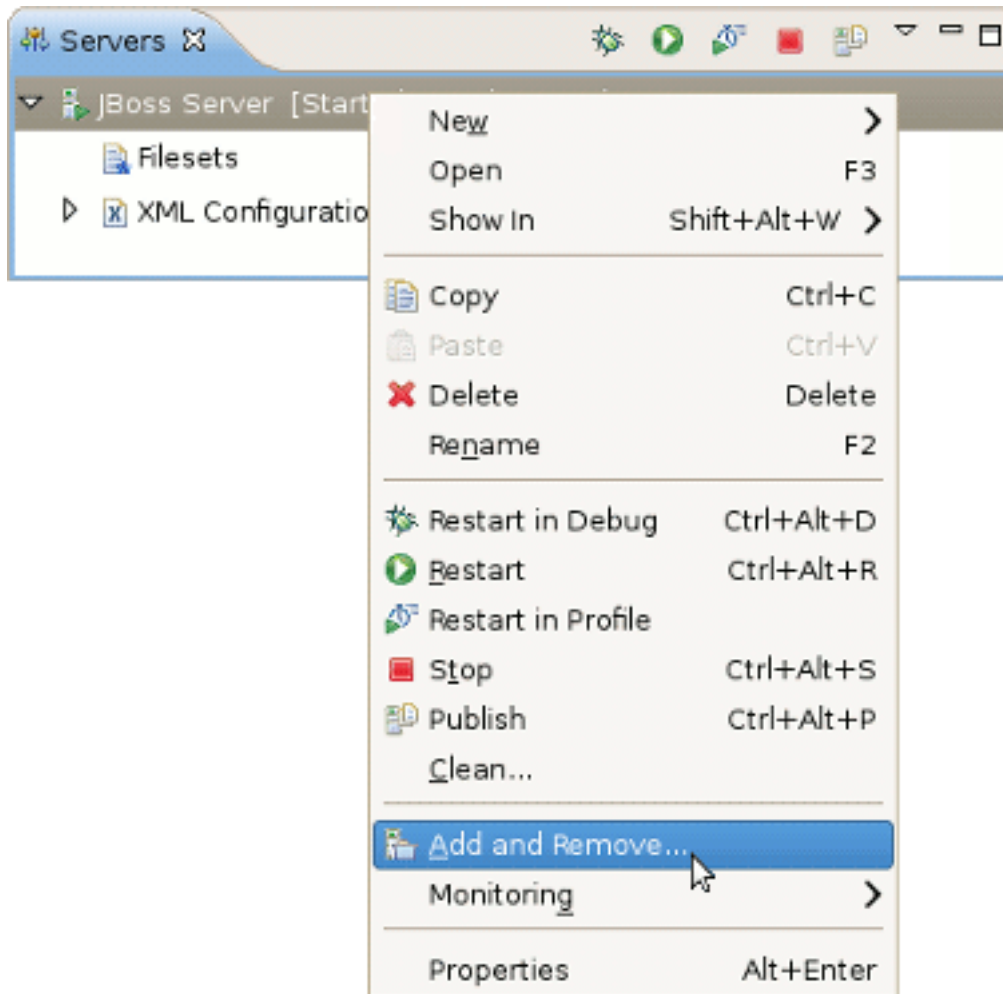
Figure 4.3. The Servers view

The following table lists possible server statuses.

Table 4.1. Server Publish Status

Status	Description
Republish	The status which allows you to see if changes are awaiting
Publishing...	The status which shows if changes are being updated
Synchronized	The status which allows you to see if changes are synchronized

You can control a server behavior as well as adjust a number of server preferences through the context menu.



**Figure 4.4. Context Menu Commands**

All available context menu commands are described in the following table.

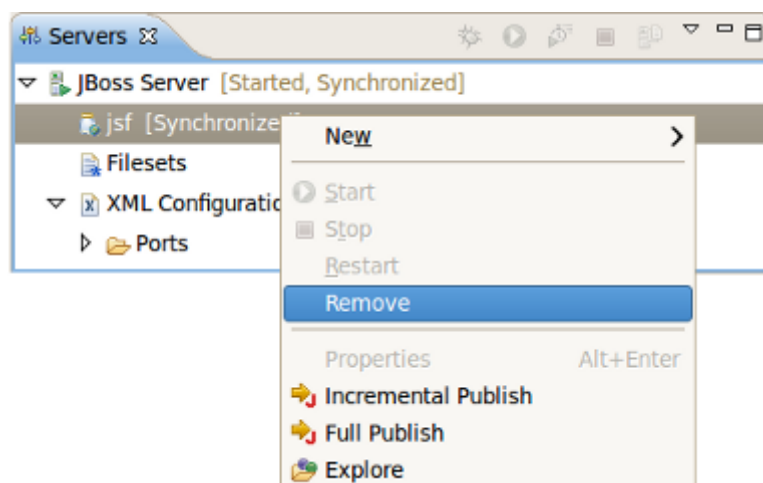
**Table 4.2. Server Properties through the Context Menu**

Name	Description
New Server	The option allows you to define a new server
Open	The option opens the Server editor
Show In	This option provides easy access to the Console, Debug, Server Log or MBean Explorer views
Show In -> File Browser	This action uses the native OS file explorer to browse the deploy destination of a local server.
Delete	Standard option that allows you to delete the chosen server
Start	This will start the server in a run mode
Debug	This will start the server in a debug mode
Stop	This will stop the server

Name	Description
Publish	This will synchronize the publish information between the server and workspace
Add and Remove Projects	This option will publish a new project to the server (if it's type is supported)
Monitoring	Allows you to add ports to be monitored on the current server
Properties	Opens a window that allows you to adjust the current server preferences

Under the server element in the **Servers** view, you can see modules that are currently deployed to the server and some server extensions that provide additional information on the server.

The context menu for any module allows you to remove it from the server and force a full or incremental republish upon it.



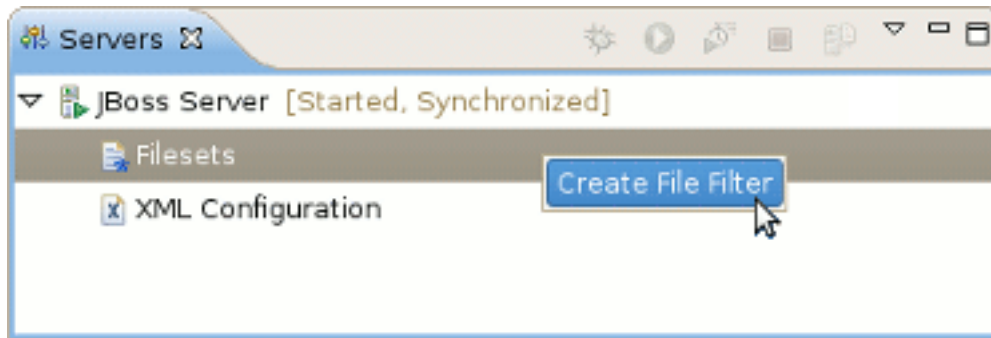
**Figure 4.5. Modules Action**

### 4.1.2.1. Filesets

The **Filesets** category in the **Servers** view provides a way to filter files.

To add a new file filter, right-click the **Filesets** category and select the **Create File Filter** option.

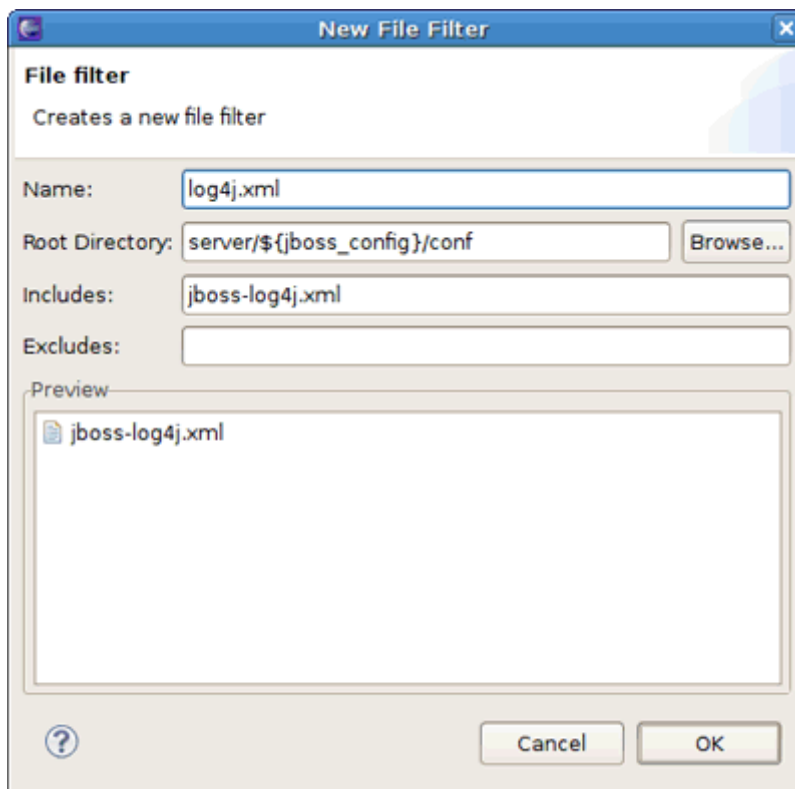
The **New File Filter wizard** should appear.



**Figure 4.6. Creating a New File Filter**

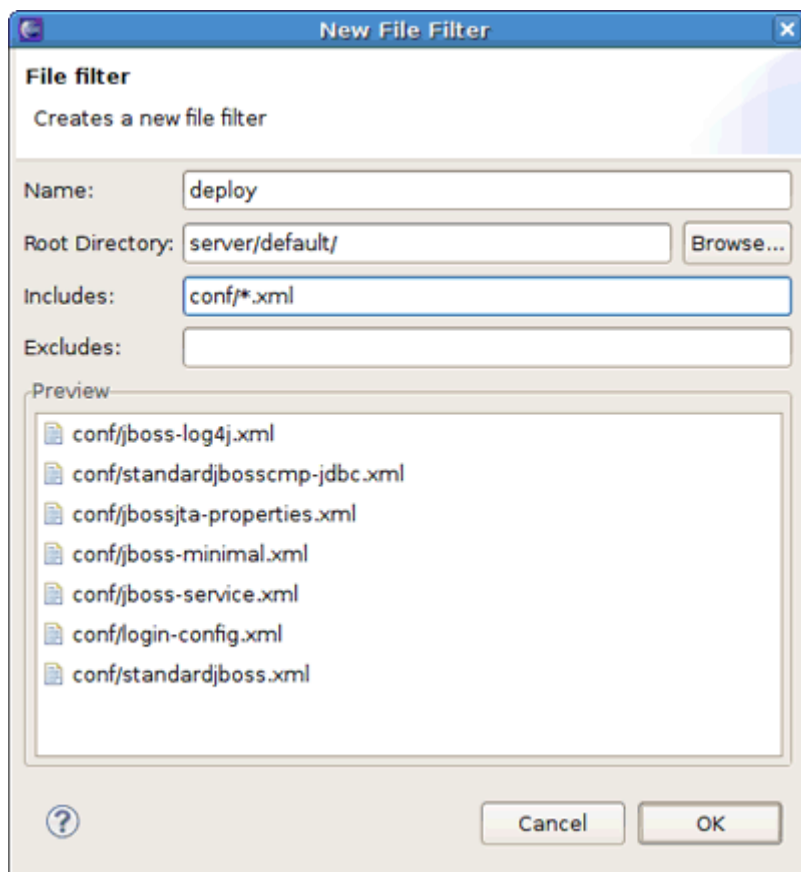
The wizard asks you to enter the filter name and add includes and excludes patterns. The preview box underneath provides a list of files matched to the defined patterns (see the figures below).

In order to set up a default fileset relative to the fixed configuration of the server runtime, use the following variable: `${jboss_config}`, i. e. you should enter `server/${jboss_config}/` in the **Root Directory** option. This allows you to modify the runtime's configuration and not have to manually update paths.



**Figure 4.7. New File Filter Wizard**

Notice, that the *Browse* button still returns an absolute path:



**Figure 4.8. New File Filter Wizard**

After the filter is created, you can explore it by expanding the **Filesets** category in the **Servers** view.

It is now possible to edit files directly from the **Filesets** category. Double clicking on a file from **Filesets** opens up the editor automatically, or you can use the **Edit File** context menu command.



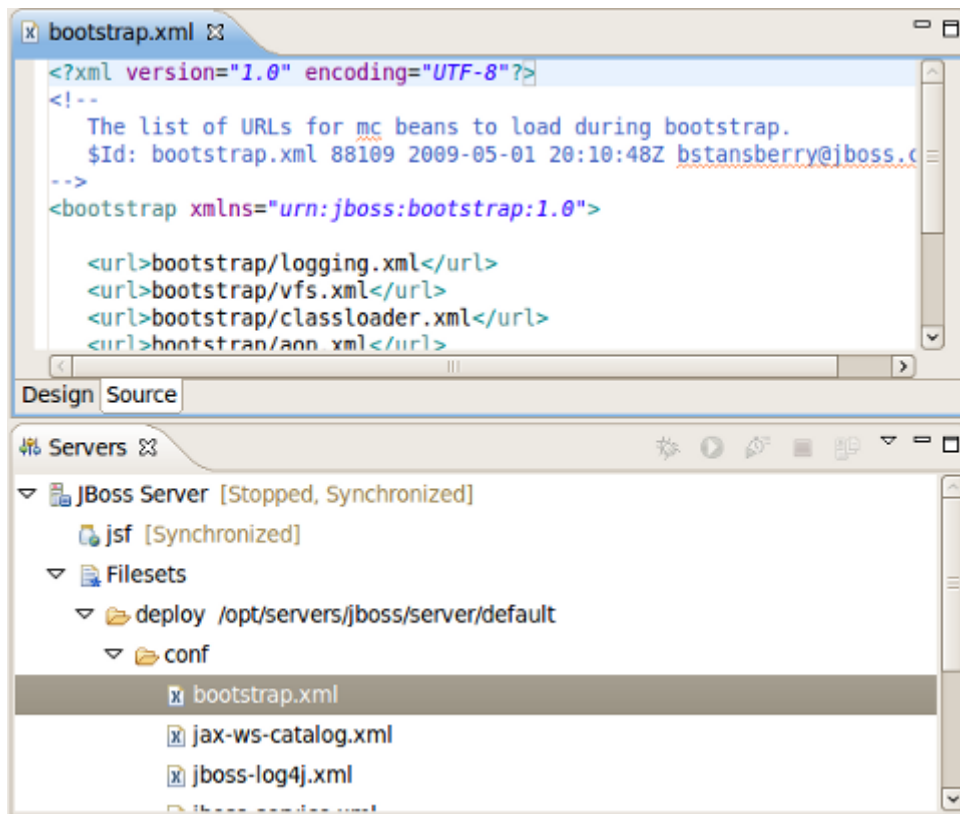


Figure 4.9. Direct Editing from the Filesets

To delete a file filter (or just a file) from the **Filesets**, right-click a file filter or file and select the **Delete File Filter** or **Delete File** command.

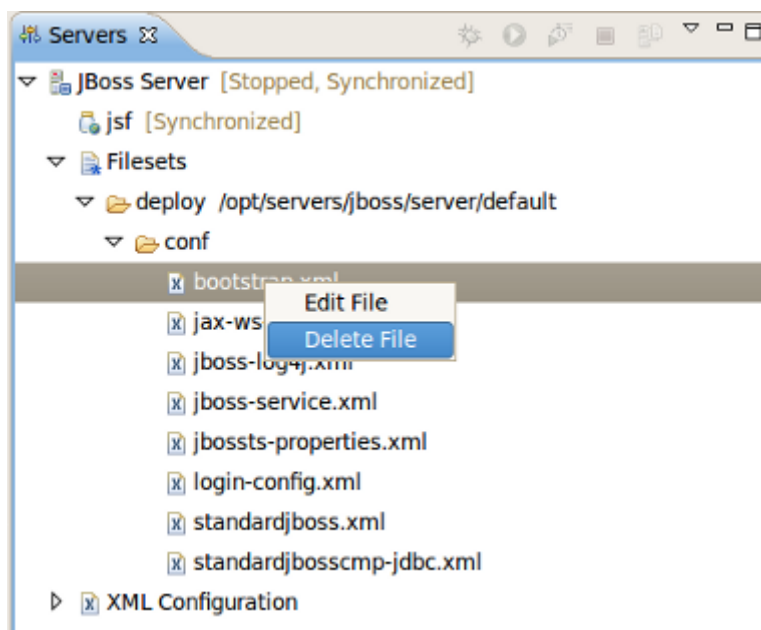
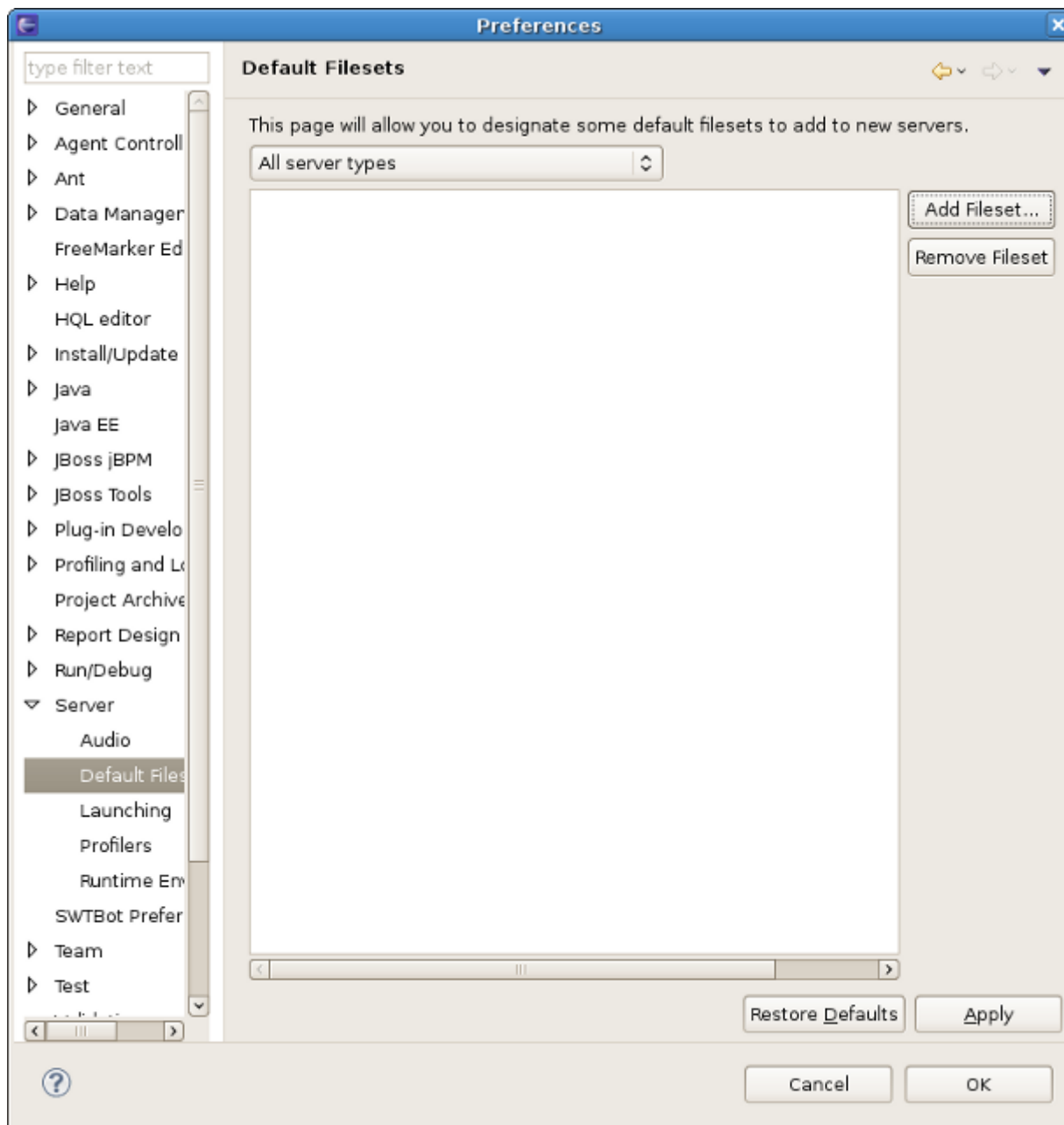


Figure 4.10. Deleting the File from the Filesets

If you want to set filesets for some server types, select **Window** → **Preferences** and then select **Server** → **Default** from the categories available on the left.



**Figure 4.11. Deleting the File from the Filesets**

On this preference page you can add a fileset to any server type or to all servers at once. To do this you should select the server type in the combo box and click the **Add fileset...** button. In the opened **New File Filter wizard** follow the steps described in [Section 4.1.2.1, “Filesets” \[42\]](#) and finally click the **Apply** button on the preference page.

The defined file filter will be automatically added to new servers during their creation.

### 4.1.2.2. XML Configuration

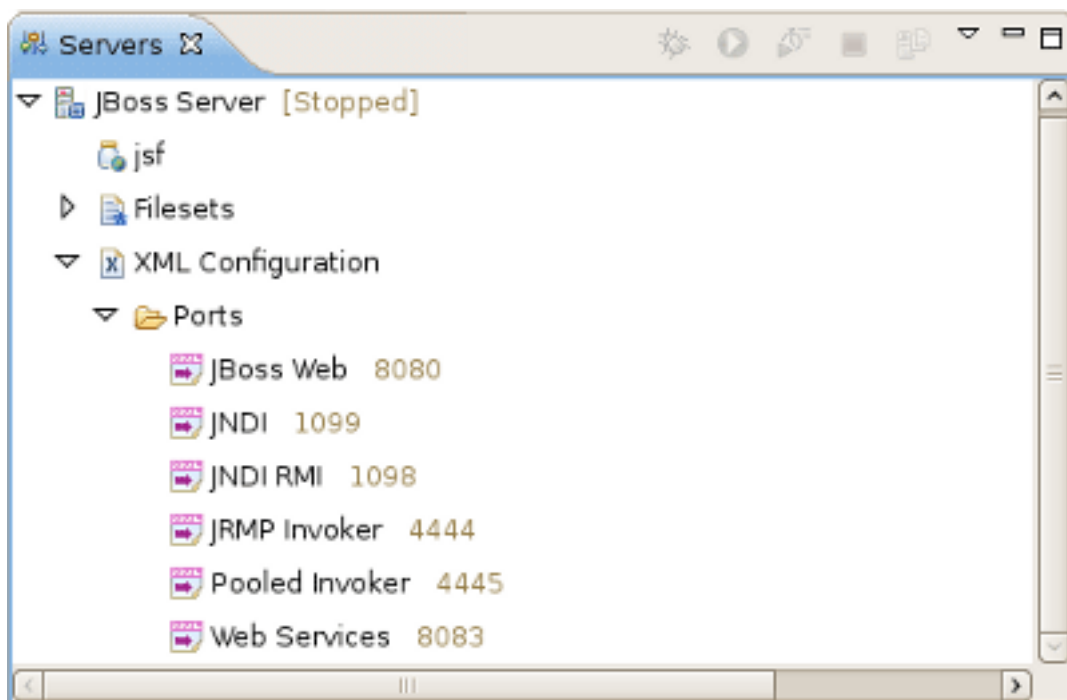
The **XML Configuration** category allows you to quickly browse to descriptor files in your server's deploy directory and check or change the values. Basically, **XML Configuration** includes XML XPaths, where an XPath is a path used to access some specific part of an XML document.



#### Note:

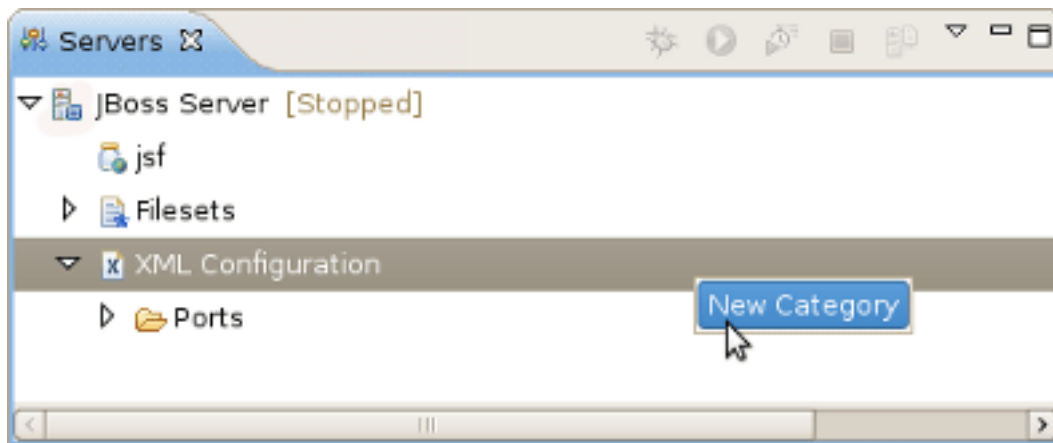
This document assumes that you are familiar with XPath. If not, we highly suggested that you look through an appropriate manual or tutorial on the topic.

The **XML Configuration** category itself contains only a list of categories. **Ports** are provided by default and display many of the most commonly used ports in the JBoss Server™.



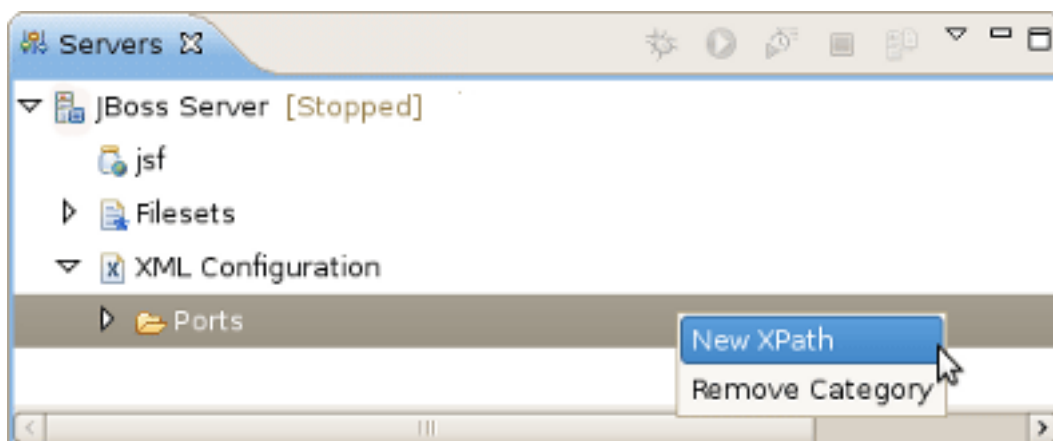
**Figure 4.12. XML Configuration**

By right-clicking on the **XML Configuration** node you can create a new category. Besides, context menu for **XML Configuration** category makes possible to disable it. You can disable any category in the bottom part of the **Servers** view. Look for them in the **Inactive Categories** afterwards to re-enable.



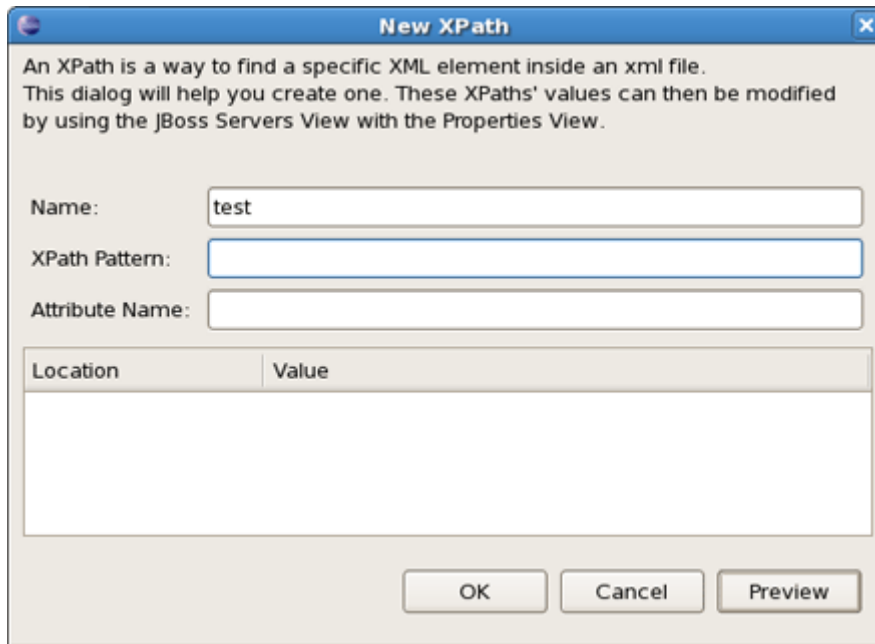
**Figure 4.13. Adding New Category**

By right-clicking on the **Ports** category, or any other category in **XML Configuration**, you can create a new XPath.



**Figure 4.14. Adding New XPath**

After that, the dialog shown below will appear.

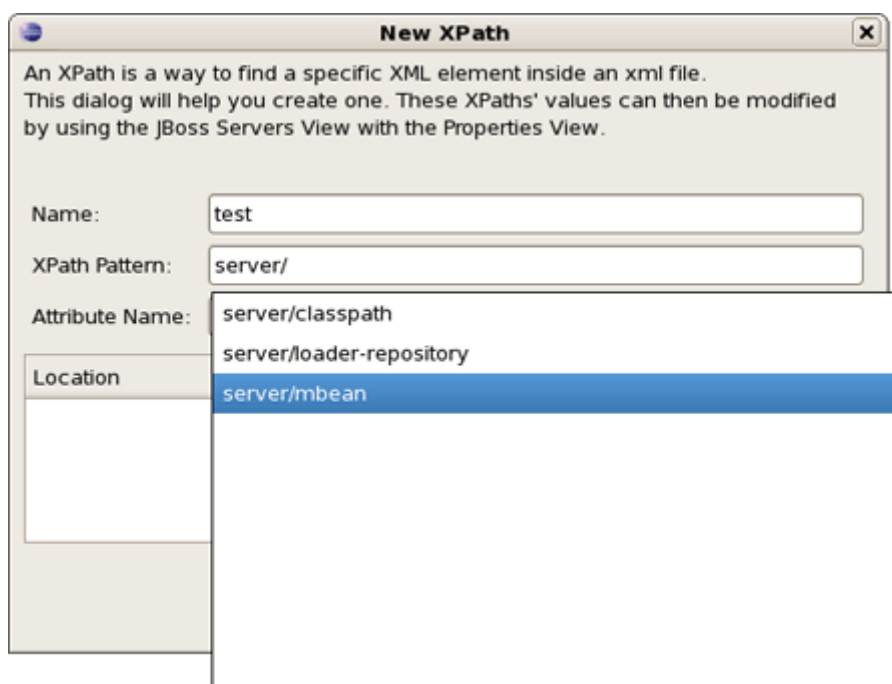


**Figure 4.15. Adding New XPath**

The goal here is to get an end result where the XPath matches up with a necessary property. With that in mind, let's look how it works. If the property you want to reach is the value of the `name` attribute in the element `<mbean>`, then your **XPath Pattern** should end with `mbean` and your **Attribute Name** should be `name`, as demonstrated in the next figure.

```
...
<server>
...
  <mbean code="org.jboss.ejb.EJBDeployer"
        name="jboss.ejb:service=EJBDeployer" xmbean-dd="">

    <!-- Inline XMBEAN Descriptor BEGIN -->
    <xmbean>
      <description>
        The EJBDeployer responsible for ejb jar deployment</description>
      ...
    </xmbean>
  </mbean>
</server>
```



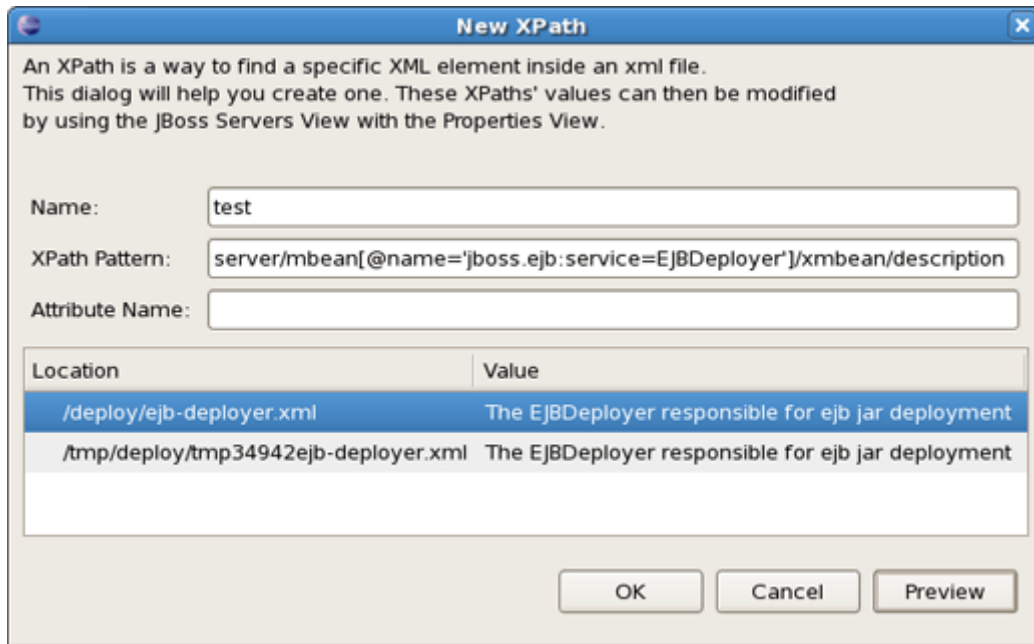
**Figure 4.16. XPath Preview**



**Tip:**

Notice when you type the fields autocomplete to help you locate exactly what XPath you're looking for.

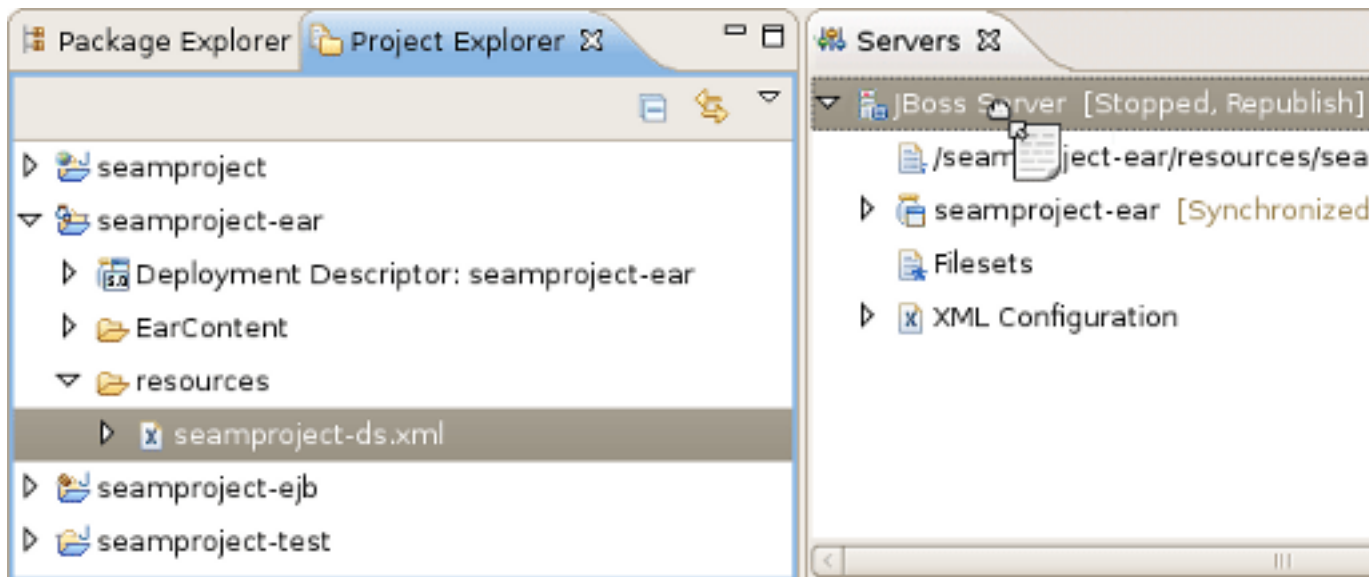
If your desired field is the text of an element `<description>`, your **XPath Pattern** should end with `description` and **Attribute Name** field should be left blank. When finished, click the **Preview** button to see how many matches are found for that particular XPath.



**Figure 4.17. XPath Preview**

### 4.1.3. Drag-n-Drop to Servers view

The **Servers** view supports drag-n-drop of deployable and runnable projects and resources.



**Figure 4.18. Dragging to the Servers view**

With drag-n-drop the following actions can be performed:

- Dragging a project to a server will deploy it to the server and run it by showing the main page in a browser.

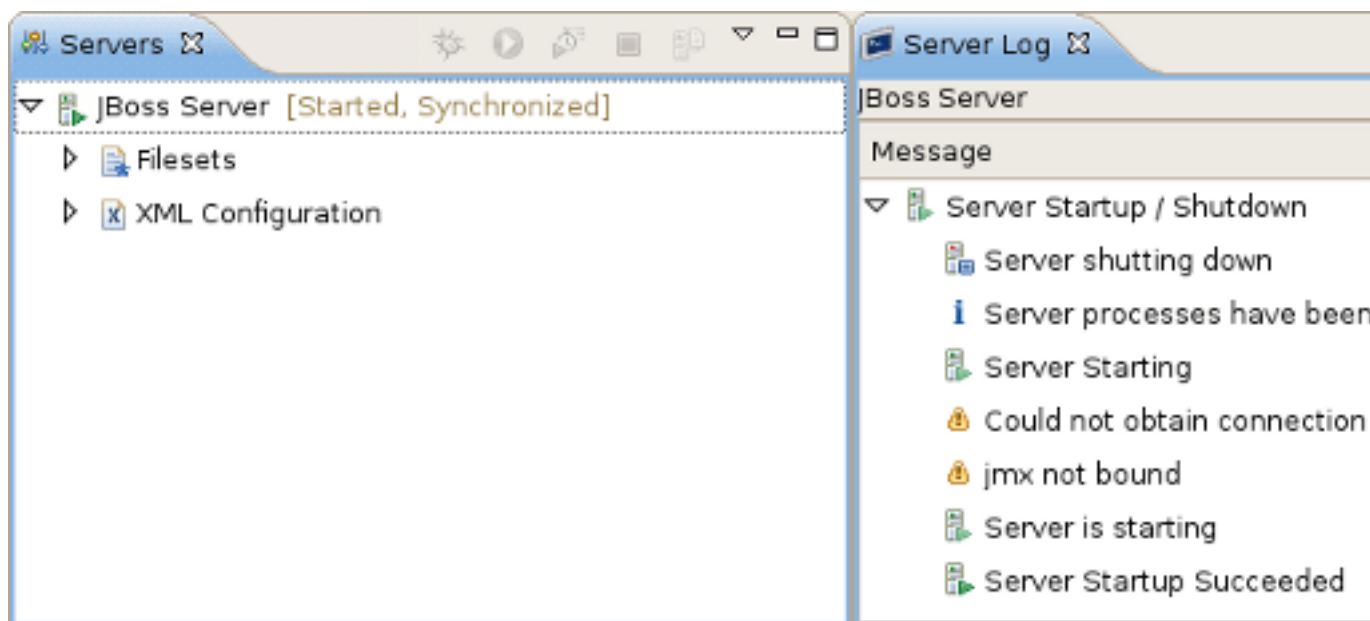
- Dragging an `.xhtml` file from the `WebContent` folder will do the same and show the corresponding page in a browser.
- Dragging a deployable resource (i.e. a datasource `-ds.xml` file that has been made deployable) will simply deploy that resource directly to the server.

In short, the feature does the same thing as if you used the **Run On Server** or **Add and Remove Projects** option in the context menu of the server.

## 4.2. Server Log View

You can monitor the current server behavior with the help of the Server Log. To open a server in the Server Log view you should right-click on the server and follow to **Open in** → **Server Log**.

The **Server Log** view shows relevant information to your server's startup, shutdown and publish processes. This allows you to keep an eye on what's going on (such as automatic incremental deployment if you have it enabled).



**Figure 4.19. Event Log Actions**

The **Server Log** view toolbar contains several icons that perform the following actions:

**Table 4.3. Server Log Toolbar Icons**

Name	Description
Export Log	Allows you to export the log into a text file
Clear Log Viewer	This option clears the current server log
Delete Log	Click to delete the server log
Open Log	Click to open the server log text file



Name	Description
Restore Log	Click to restore the server log

## 4.3. Relevant Resources Links

Find more about XPath in the [XPath Documentation](http://www.w3.org/TR/xpath20/) [http://www.w3.org/TR/xpath20/].



# Projects

The most popular of the projects we deal with are the J2EE ones, such as Dynamic Web Project, EJB Project, or EAR project. JBoss Tools™ web projects include Struts, JSF and Seam projects. These are referred to as faceted projects. This chapter will cover facets, which are used to provide a consistent structure and packaging features to any type of project.

## 5.1. Faceted Projects Overview

The idea behind faceted projects is that each project can accept units of functionality, or facets, which can be added or removed by the user. These facets either add to the project's classpath, enable a builder, or watch the project in some other fashion. Typically every project concerned has at least one facet when it is created. As an example, a Web project has a WebDoclet facet, or an EJB Project has an EJB Module facet as prerequisites.

WTP projects have been criticized for being over-engineered or too restrictive in their design. WTP projects are set up in a tree-relationship to each other, where one project can be a child of another. For example, an EAR project may have a Web Project child, an EJB project child, or other types.

However, the benefit of this is that the structure of your projects is then known and packaging it up *should* be trivial. If your project is non-standard, or you feel too confined by such rigid structural requirements, you can still choose to package your project using the Archives plugin (see [Section 7.1, “Project Archives View”](#)).

## 5.2. Adding Facets to a Project

This section will cover the facets added by JBoss Tools and show how you can configure them in a project by adding new ones or modifying existing facet configurations.

One way to configure the facets is doing it while organizing a new project. To demonstrate this create a new **Dynamic Web Project** by clicking on the **Dynamic Web Project** option in the **Create Projects** section of **JBoss Central**.

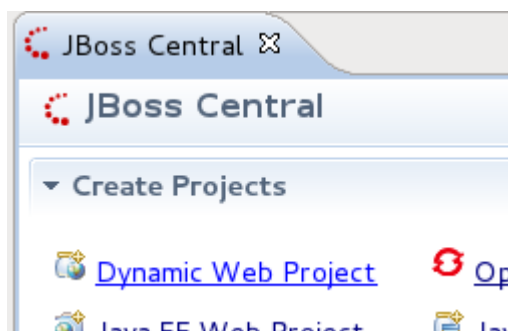
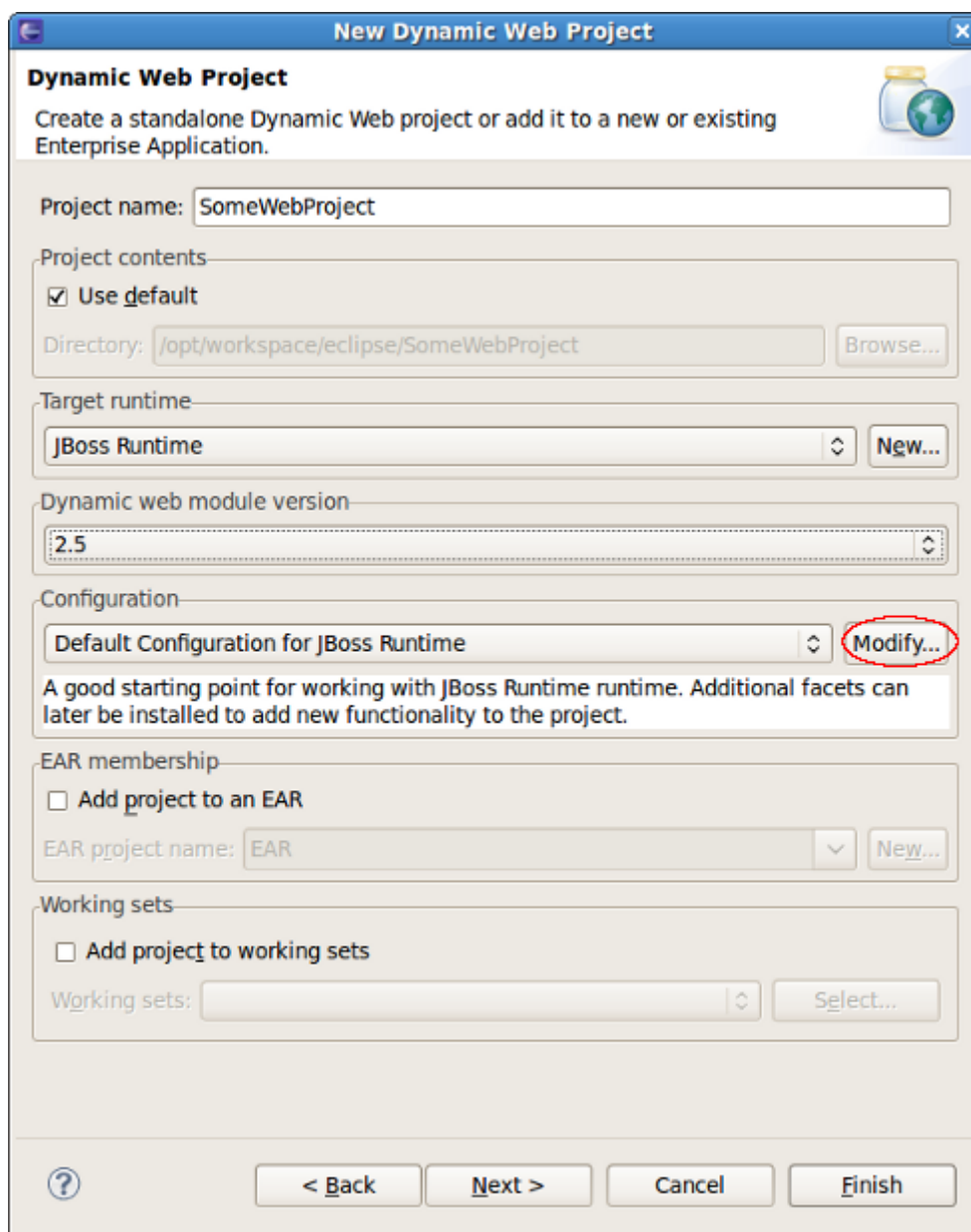


Figure 5.1. New Dynamic Web Project

Click the **Next** button and you will see a Dynamic Web Project page like on the figure below.

The first page of most WTP projects allows you to target a specific runtime, which represents a server's library location. It will also provide you the ability to add this project to an EAR project and select a preselected default set of facets, called a configuration, rather than manually select each required facet.

Selecting the runtime allows the project to install the proper classpaths to the project so it knows what code to compile against.

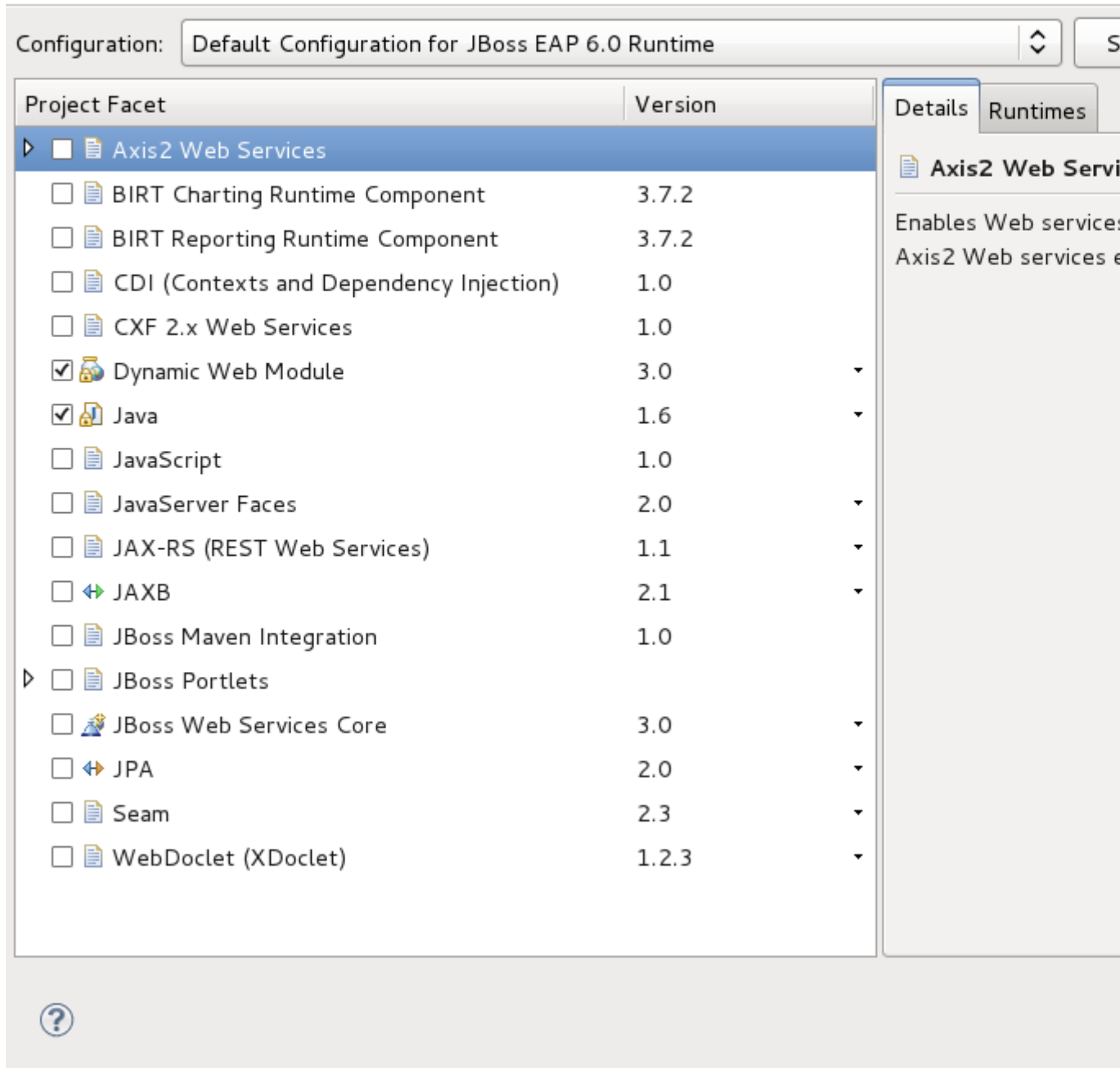


**Figure 5.2. New Dynamic Web Project**

Click the **Modify** button next to the **Configuration** section to open a wizard which allows you to modify the chosen configuration. The wizard is shown in the image below.

## Project Facets

Select the facets that should be enabled for this project.



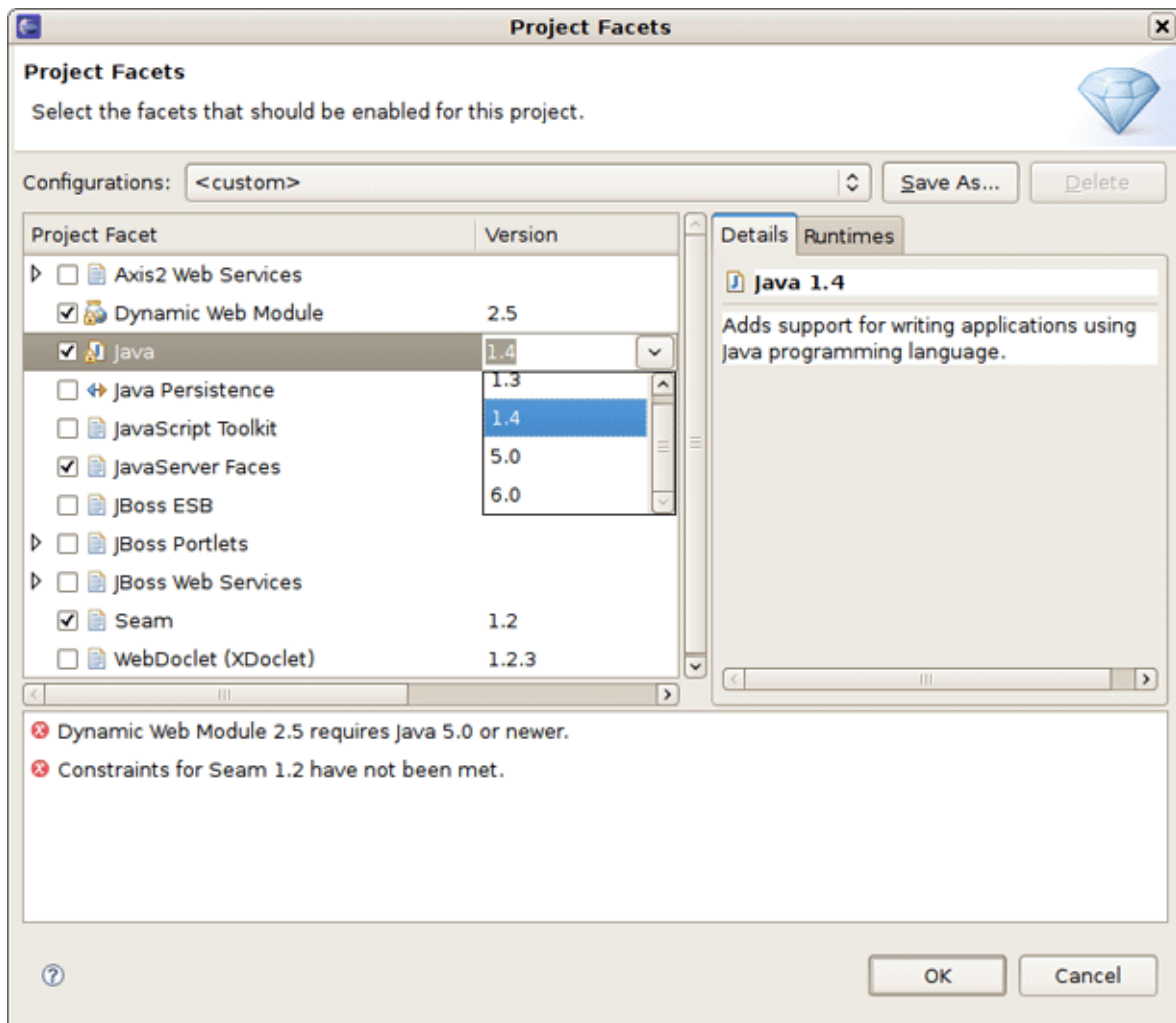
**Figure 5.3. Project Facets Wizard**

Here part of the listed facets are those which are provided by WTP. Some of them are added by JBoss Tools. They are:

- BIRT Charting Runtime Component

- BIRT Reporting Runtime Component
- CDI (Contexts and Dependency Injection)
- CXF 2.x Web Services
- JAX-RS (REST Web Services)
- JAXB
- JBoss Portlets
- JBoss Web Services Core
- JPA
- Seam 2

On this wizard page you can enable or disable any facet as well as change its version. What you should note here is that some facets or facets versions may conflict with each other. In case of incompatibility you will be notified in the combobox underneath.

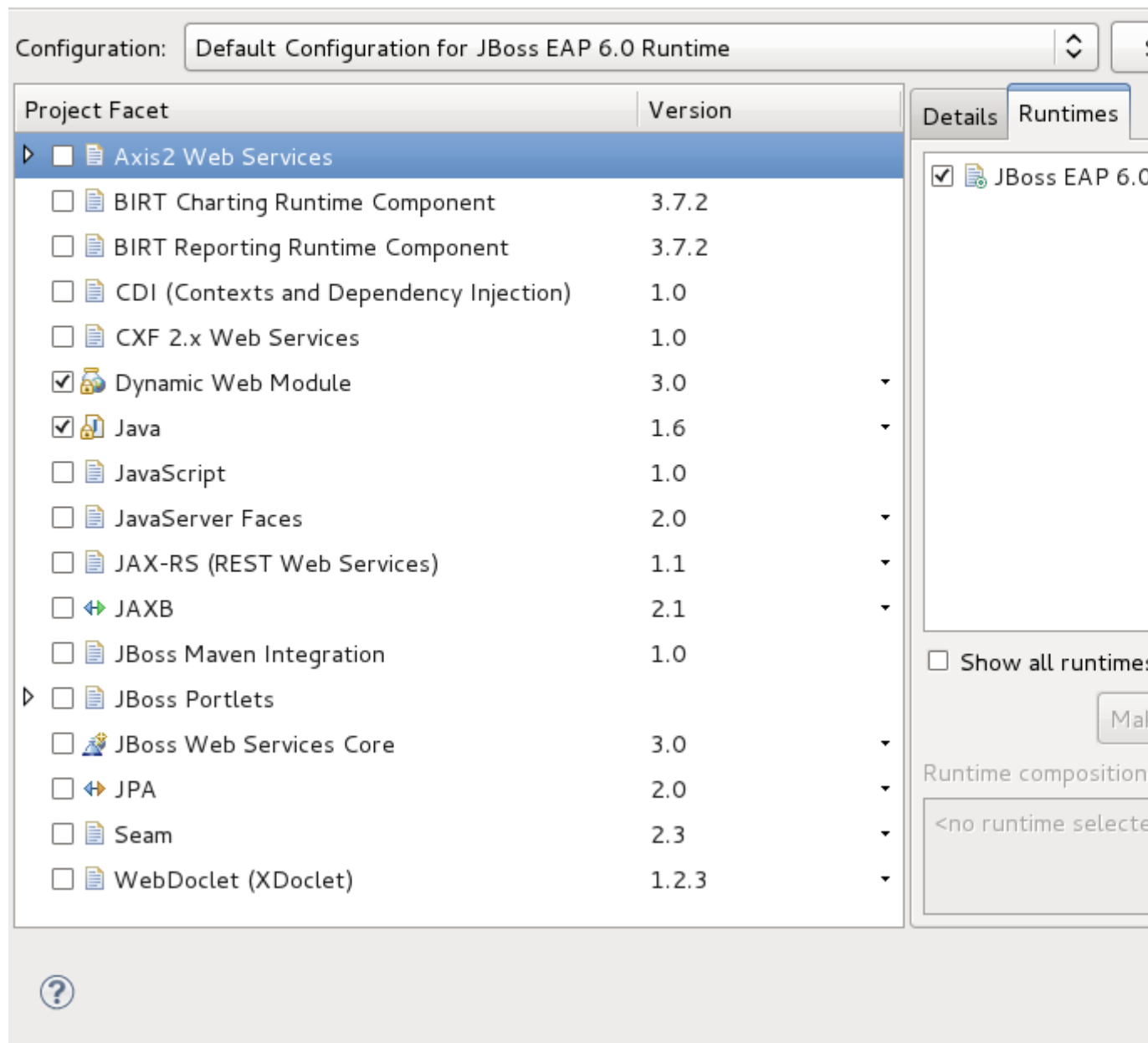


**Figure 5.4. Facet Constraints**

When switching on the **Runtimes** tab on the right you will see the current server Runtime.

## Project Facets

Select the facets that should be enabled for this project.



**Figure 5.5. Runtimes on the Project Facets Wizard**

On this tab you can also create a new Server Runtime and make it primary by enabling it and then clicking the **Make Primary** button.

Clicking on the **OK** button will save the chosen configuration of the facets and return you to the Dynamic Web Project wizard (see [Figure 5.2, “New Dynamic Web Project”](#)). Additional pages in the wizard are specific to either the project type or the facets selected.



If you need to configure the facets for an existing project, right click on the project, select **Properties** and then select **Project Facets**. This will bring up the Project Facets wizard (see [Figure 5.3, “Project Facets Wizard”](#)), where you can create your own custom facets configuration.

## 5.3. Relevant Resources Links

More information on the WTP facets can be found in the [Eclipse help](http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.jst.j2ee.doc.user/topics/cfacets.html) [<http://help.eclipse.org/ganymede/index.jsp?topic=/org.eclipse.jst.j2ee.doc.user/topics/cfacets.html>].



# Deploying Modules

In this chapter it will be described how to deploy modules onto the server.

There are several ways to deploy to a server, provided by the Web Tools Platform (WTP) and some additional methods provided by JBoss Tools. These methods are described further in this chapter.

## 6.1. Deploying on the Package Explorer

On the package explorer it is possible to publish either a project to a server or just a single file. Let's look at how to do this.

### 6.1.1. Deploying with Run On Server Wizard

The first WTP method is to right-click on a project, such as a Dynamic Web project, EJB project, or EAR project and then select **Run As** → **Run on Server**. The resulting dialog allows you to select which supporting server the project can be published to.

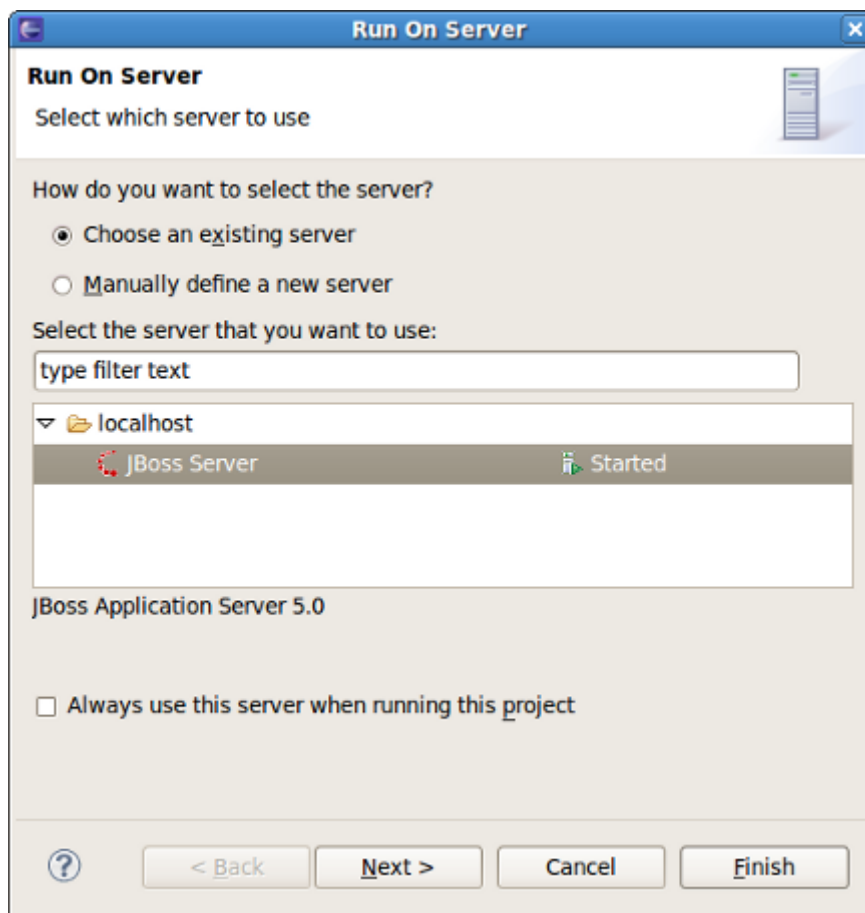
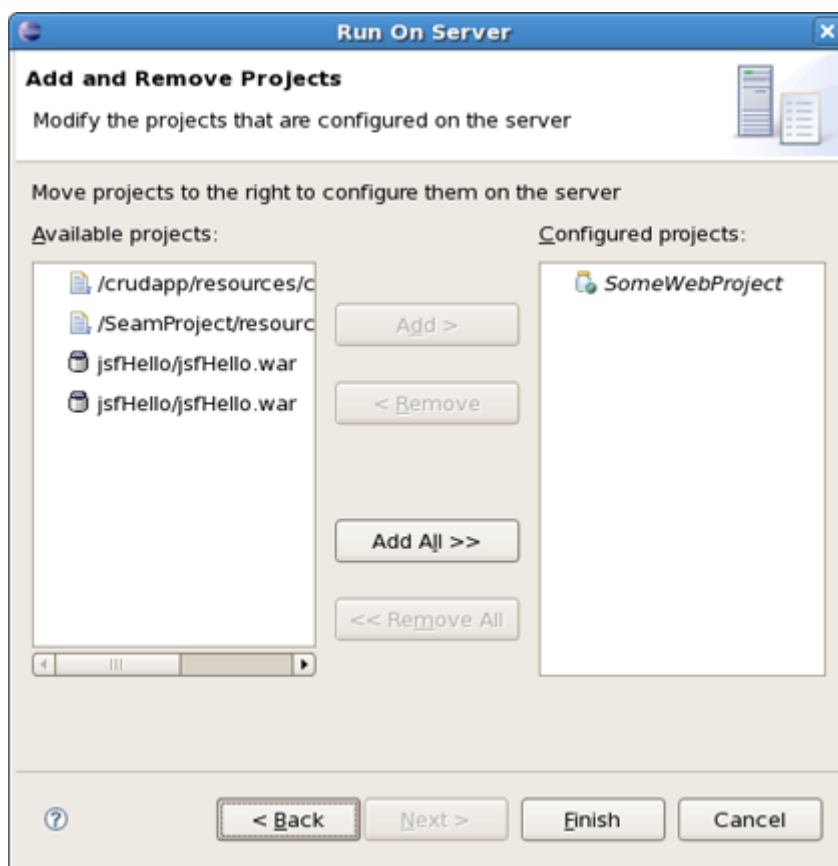


Figure 6.1. Define a New Server

Click the **Next** button to see add or remove projects page where you can choose projects to configure them on server.



**Figure 6.2. Add or Remove Projects**

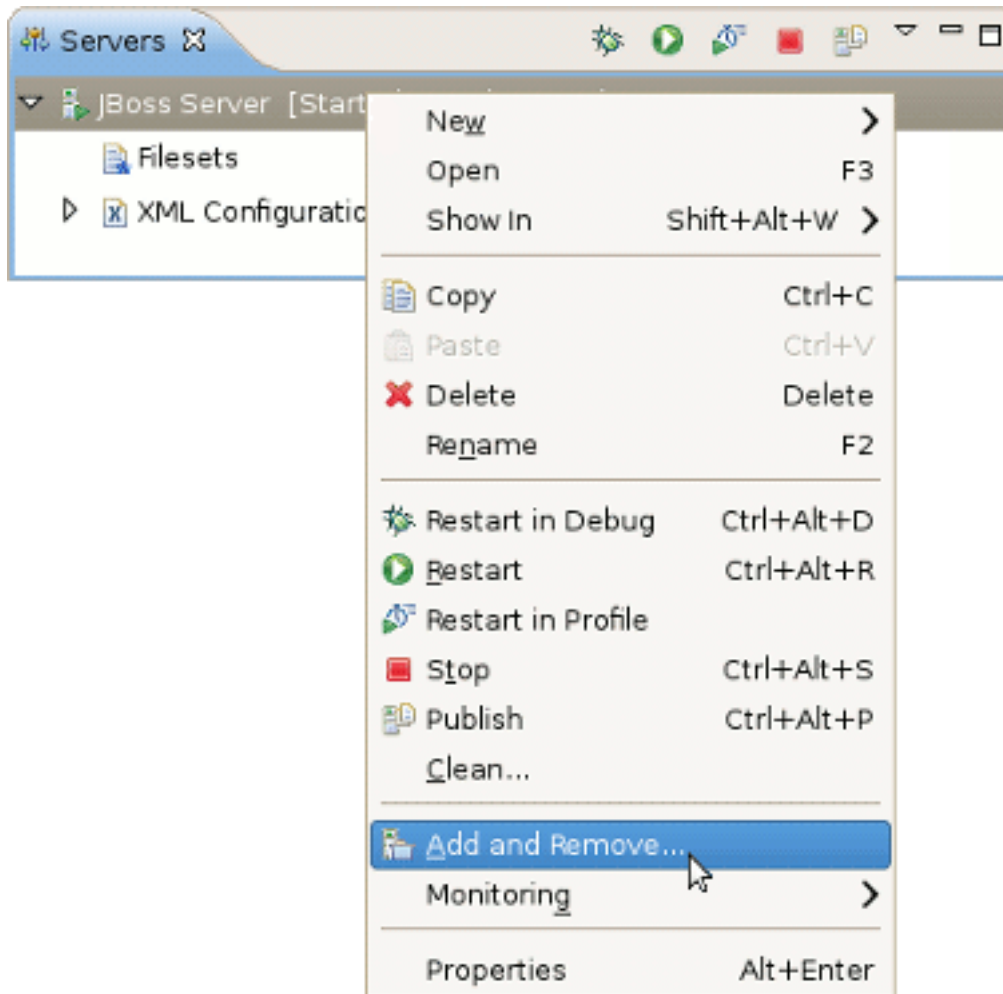
This page of the wizard also allows to undeploy modules from the server. For that choose proper module(s) from the right and click the **< Remove**. The modules will be completely undeployed after restarting your server or republishing.

Generally, for the JBoss AS Server Adapters, publishing using this method will force a default, best-guess, packaging configuration for your project. This best-guess does not publish incrementally, but instead repackages your entire project into a `.war`, `.jar`, or `.ear` as appropriate and then copies that file into the proper deploy directory. For quicker smarter deployment, you will need to create archives using the Project Archives view (see [Section 7.1, “Project Archives View”](#)) and customize packaging yourself.

## 6.2. Deploying with Servers View

The root elements of the Servers View are the server objects. Expanding these, you will see the modules listed. With this in mind, we suggest two more ways to deploy resources onto the server.

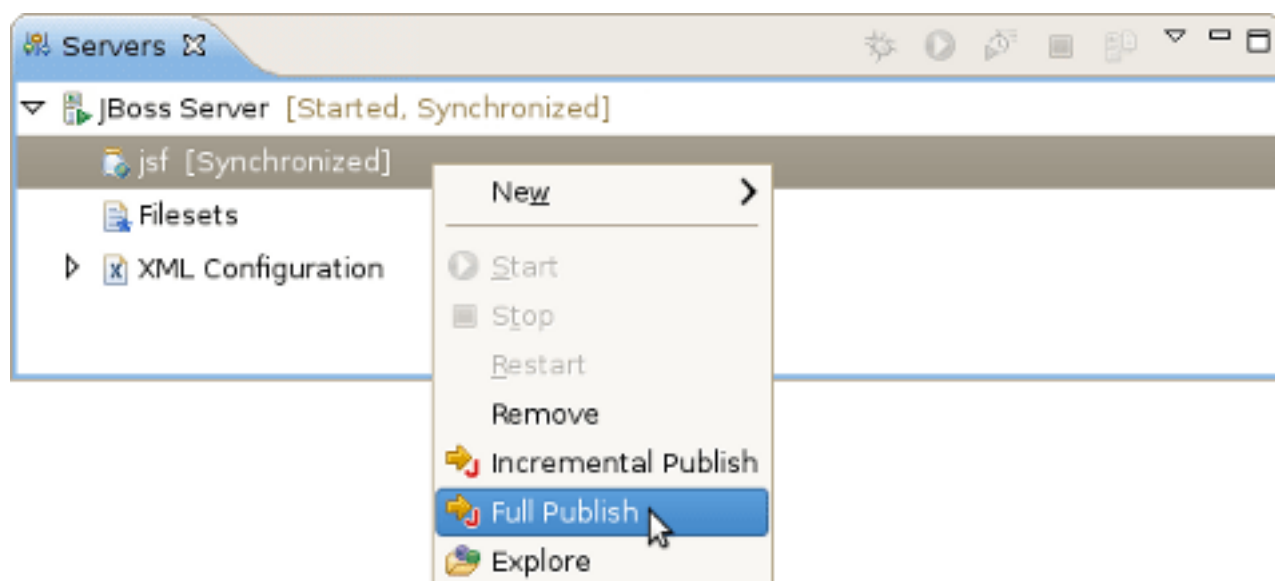
For the first method, you should right click on a server and select the **Add and Remove** menu item.



**Figure 6.3. Add and Remove Projects**

This will bring up a dialog (see [Figure 6.2, “Add or Remove Projects”](#)) that allows you to either publish projects or modules to a server, or remove them from the server. If the selected module is a project like a Dynamic Web project, EJB project, or EAR project, it will be published as through **Run on Server** wizard, with a best-guess full package. If, however, the selected element is an archive from the Project Archives view (see [Section 7.1, “Project Archives View”](#)), it will be published according to the rules of that module type.

Nested beneath each server, there is a list of modules. Each module displays the module name, as well as decorated information on the module's various states. Right-clicking on the desired module and selecting **Full Publish** will force a full rebuild of the entire module.



**Figure 6.4. Full Publish**

Here, **Incremental Publish** is meant to enable publishing of only those parts where changes have been made.

### 6.3. Redeploying with Finger Touch

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



**Figure 6.5. 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).

# Project Archives

## 7.1. Project Archives View

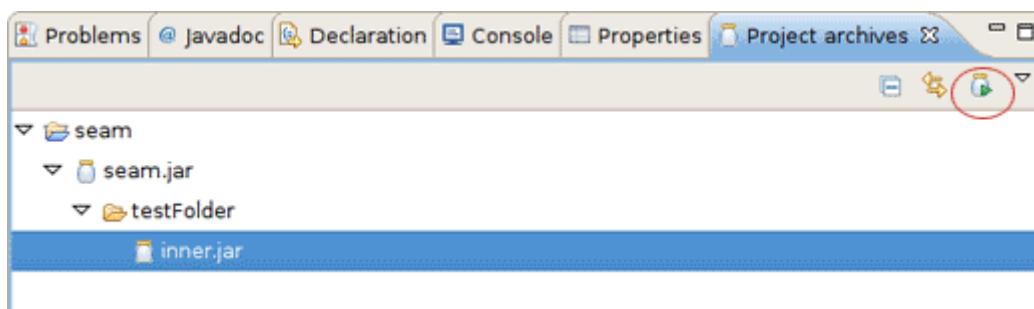
Every application, whether Plain Old Java, J2EE, or some other language altogether, needs to be packaged in some way. In Java-related projects, many people use ANT.

But JBoss Tools™ comes with our own Archives tool with simpler and less-verbose XML and a handy user interface. The Project Archives plugin consists primarily of the **Project Archives** view to set up each packaging configuration.

Let's look through all functionality that the **Project Archives** view provides.

### 7.1.1. Overview

The packaging configuration for each project is stored in the project's root folder in a file named `.packages`, which has a fairly simple XML structure. Modifying the file by hand is neither required nor recommended, as the UI is the only supported way to modify your packaging structure.



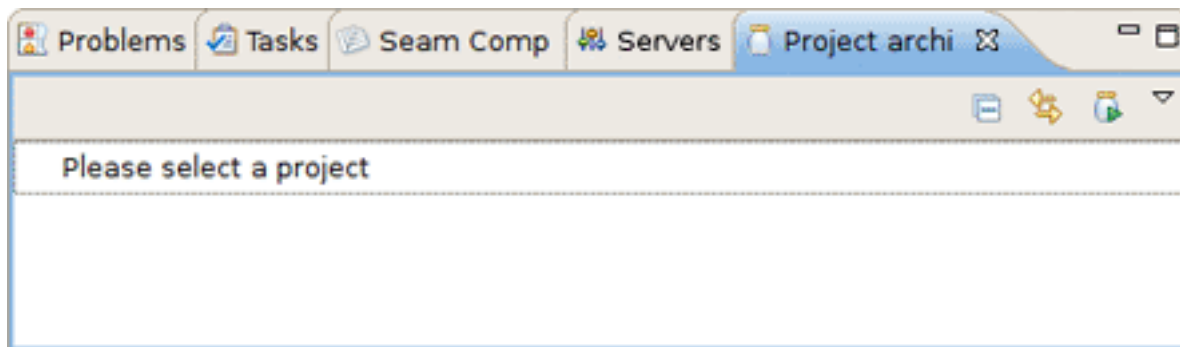
**Figure 7.1. Archives View**

A project's configuration contains archives. As you can see on the image above a project can contain more than one archive. Internal archives and filesets can be directly inside of an archive, or in a sub-folder of that archive.

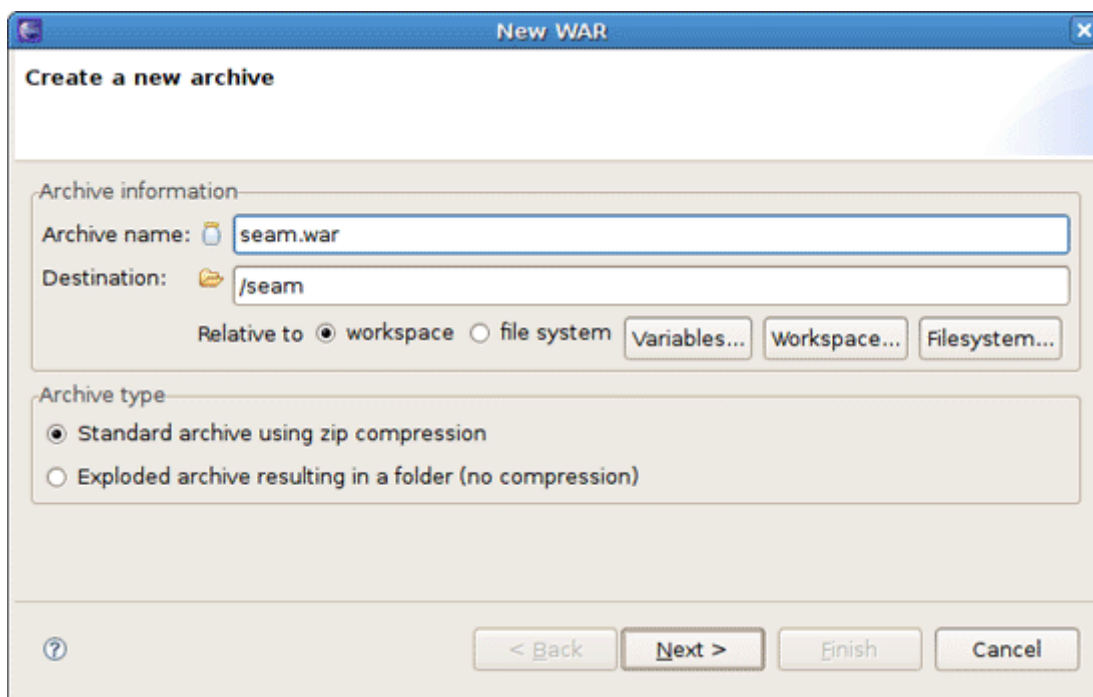
In the upper right corner of the view you can see an icon which, when clicked, will build the selected top-level archive. Additionally, you can select **Project** → **Build Packages** when a project is selected in the **Packages View** to build all declared packages in that project's `.packages` file. This will execute a full build on all declared archives.

### 7.1.2. Creating an Archive

When you open the **Project archives** view for the first time, it asks you to select the project for which you want to create an archive.

**Figure 7.2. Archives View**

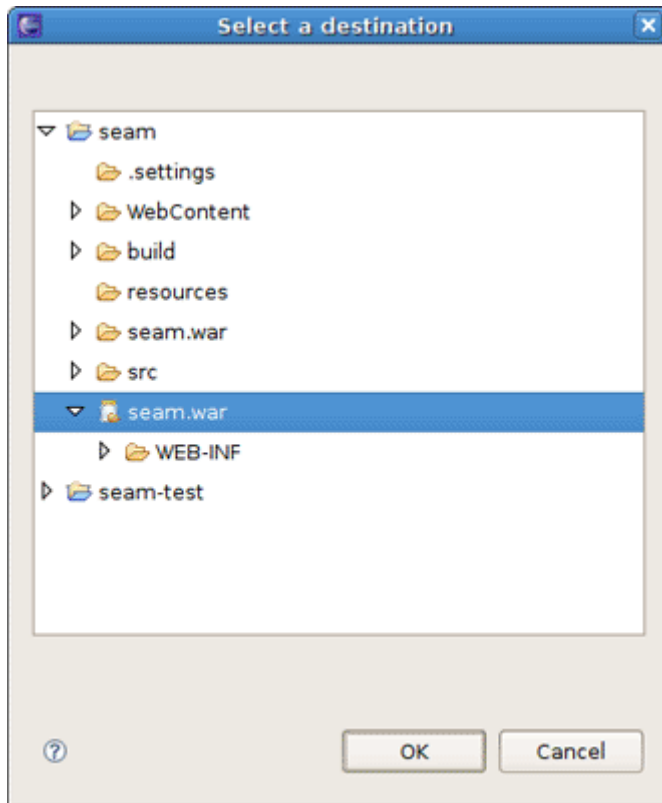
To get started creating a jar, you need right-click inside the view and select **New Archive**. When creating your new JAR output, you can customize the name, add folders, filesets and inner JARs to it. Shown below is the first page of the New archive wizard.

**Figure 7.3. New JAR Wizard**

The page is pretty simple. First it prompts you to set the name of your new archive and a destination.

The destination of an archive can be anywhere on the file system, anywhere in the workspace, inside another archive, or inside a folder declared inside an archive. Select the appropriate checkbox (either **workspace** or **file system**) to specify that the destination is related to either the workspace or filesystem. You can browse to workspace or filesystem destinations by clicking on their respective buttons. To select a destination inside some other archive, you'll need to click the **Workspace** button.

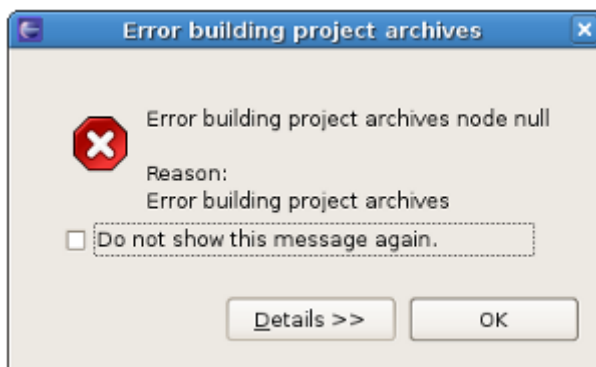




**Figure 7.4. Selecting the destination in the workspace**

Also in the wizard for creating a new archive you can choose whether an archive to be compressed or exploded into a folder (without compression). You need just select proper checkbox in the **Archive type** section.

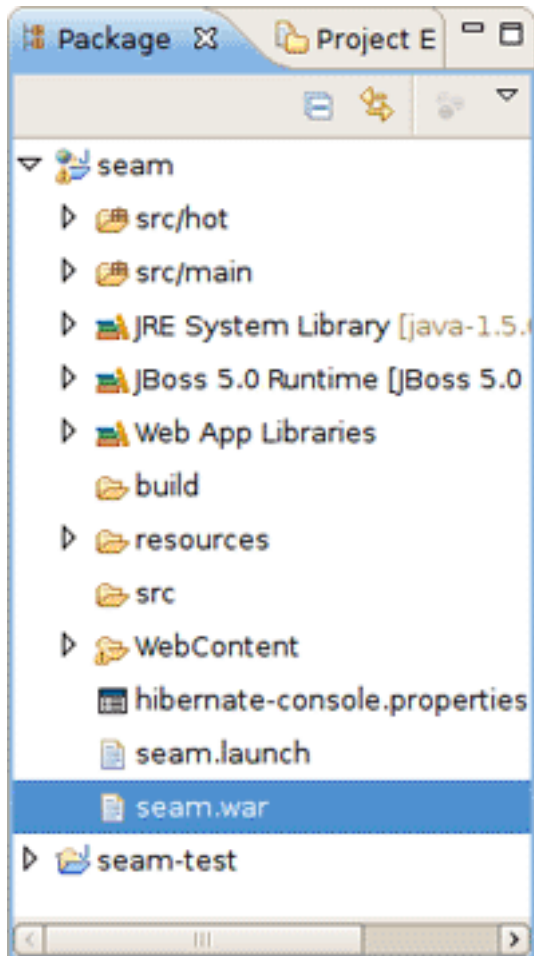
If a build or incremental update fails Project Archives will show an error dialog:



**Figure 7.5. Selecting the destination in the workspace**

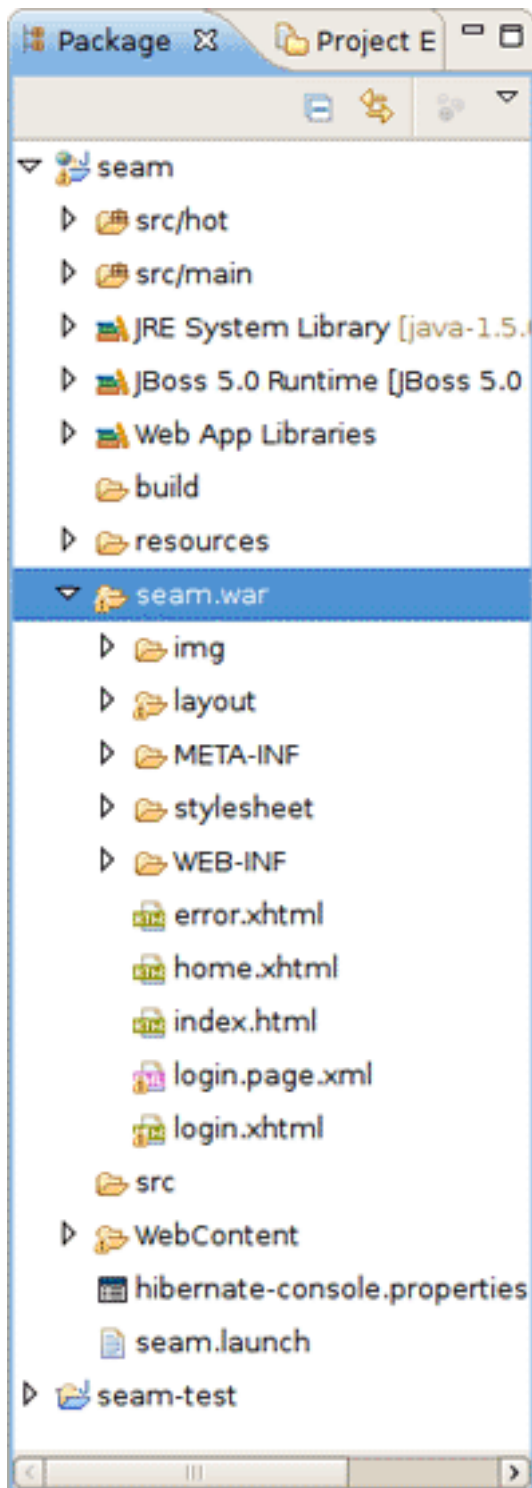
Click the **Details** button to view detailed information about the cause of the error.

In the **Package Explorer** you can view the created archive.



**Figure 7.6. The Archive in the Package Explorer**

If you use the exploded type of archiving, instead of a single file archive the result put into a folder is displayed in the **Package Explorer**.



**Figure 7.7. The Exploded Archive in the Package Explorer**

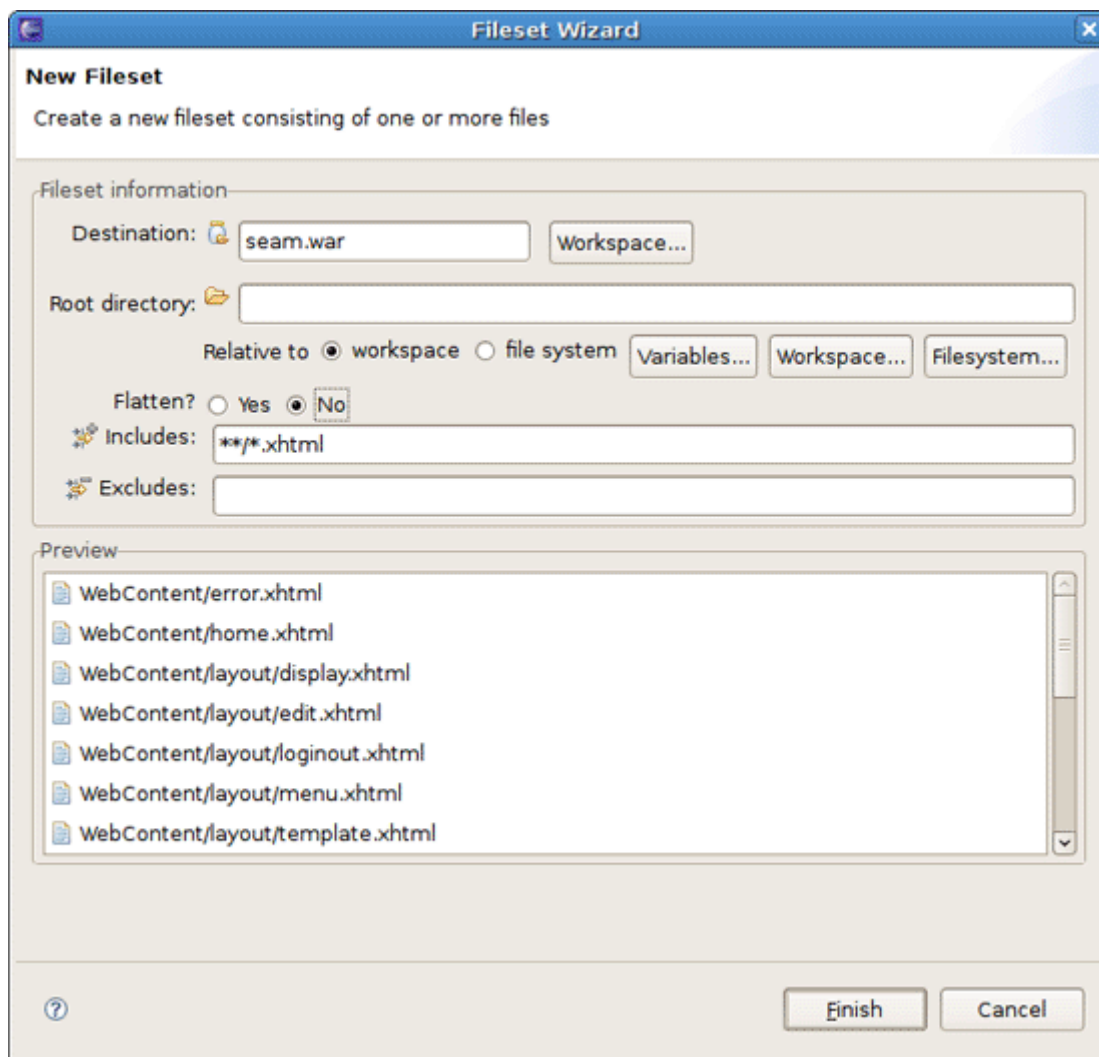
#### 7.1.2.1. Creating a Folder

To create a folder right-click on an archive or folder you want your new folder to be a child of. The only piece of required information the folder name.

### 7.1.2.2. Creating a FileSet

To create a new fileset, right click on an available target location such as an archive, a nested archive, or a folder within an archive and select the **New Fileset** option.

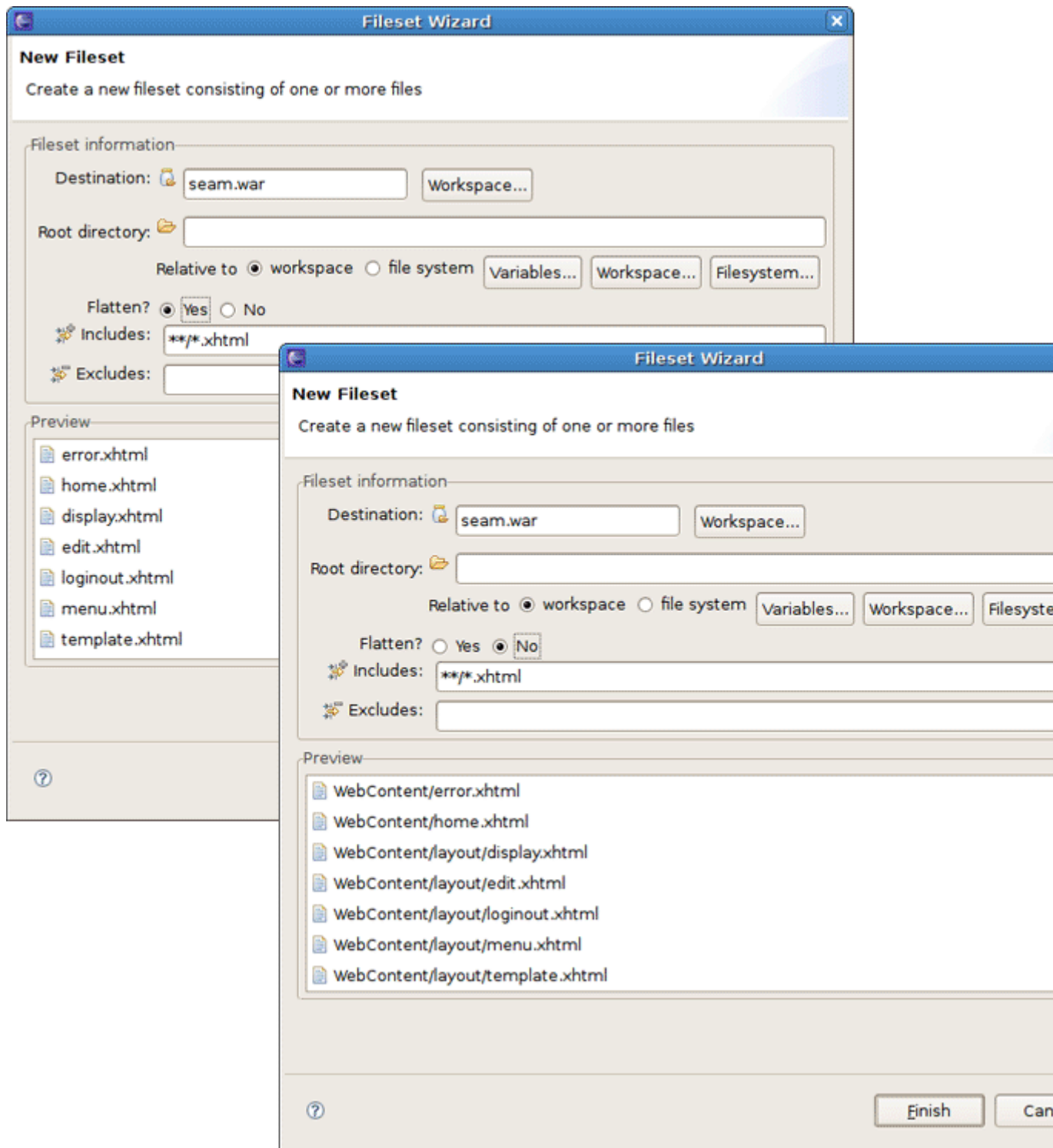
The **New Fileset** wizard requires a destination (where the files will be located) and a root directory (or where the files are coming from). The source can be anywhere in the workspace or from the filesystem at large.



**Figure 7.8. Adding a New FileSet**

Below that, the fileset requires only an **Includes** and **excludes** pattern. As you type in either of these fields, the preview viewer will list those files that are matched.

You can create a Fileset with flattening or without it. Look at the difference on the figure below.

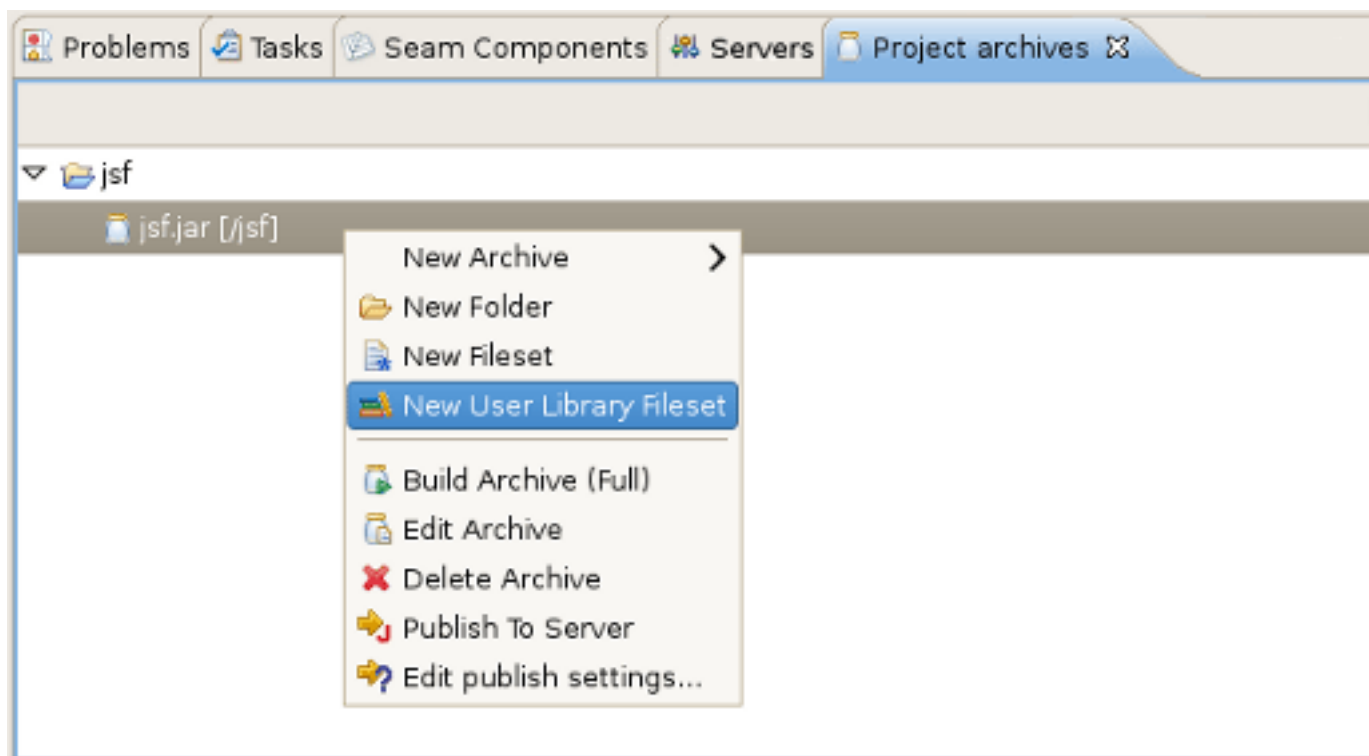


**Figure 7.9. The FileSet with flattening and without it**

### 7.1.2.3. Creating User Library FileSet

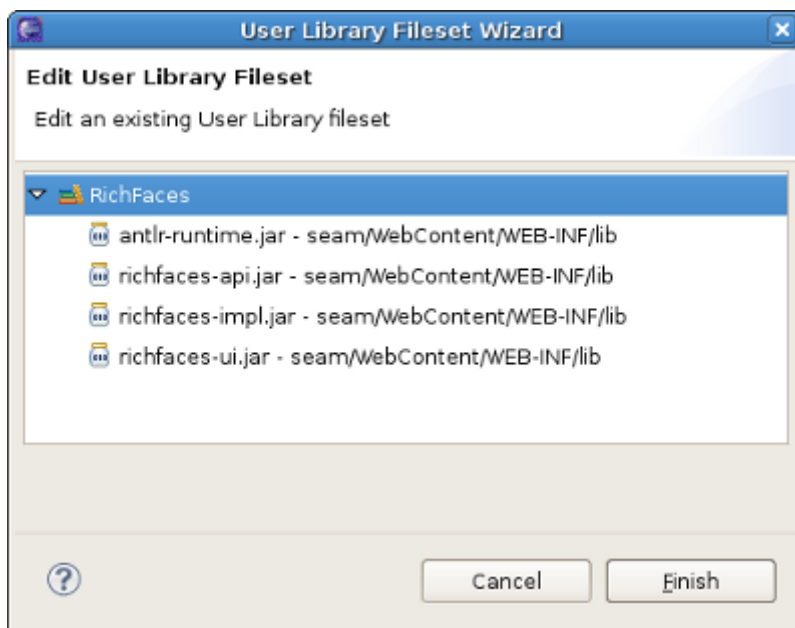
If you make use of user libraries in your projects you can also refer to these from project archives and have all the JAR and ZIP files they refer included into the archive.

To add a new user libraries file set, right-click on the necessary archive and select the **New User Libraries FileSet** option.



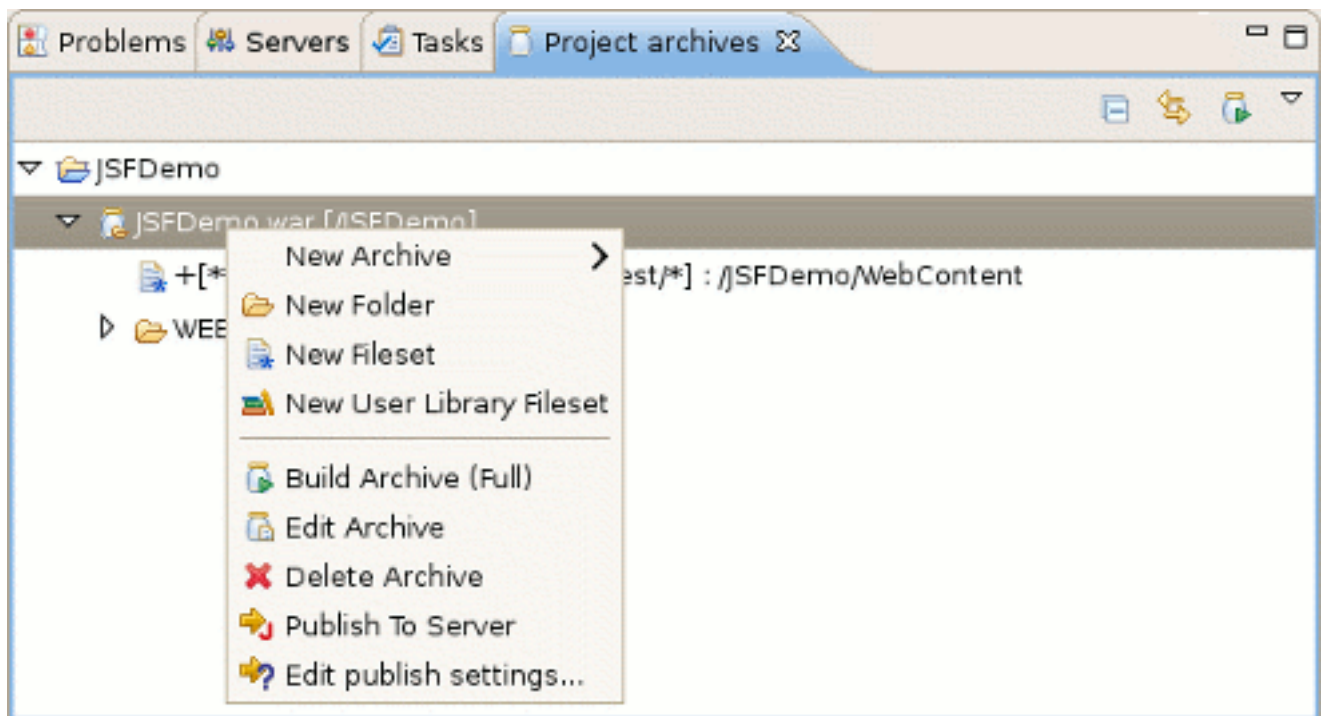
**Figure 7.10. Adding New User Library Fileset**

You can edit the existing user libraries as well using **User Libraries Fileset Wizard**. Right-click on the library fileset and select the **Edit Fileset** option.



**Figure 7.11. Editing User Library Fileset**

### 7.1.3. Archive Actions



**Figure 7.12. Context Menu on the Item**

There are a number of variable options in the context menu, but there are also several that come standard.

**Table 7.1. Context Menu on the Item**

Name	Description
Build Archive (Full)	This action is enabled only on top-level archives and initiates a full build on that archive
Edit Archive	Standard action that brings up the wizard associated with that particular node type and allows the details to be changed
Delete Archive	This option deletes the selected node
Publish To Server	This action will publish to a declared server
Edit publish settings	This option edits the archives publish settings

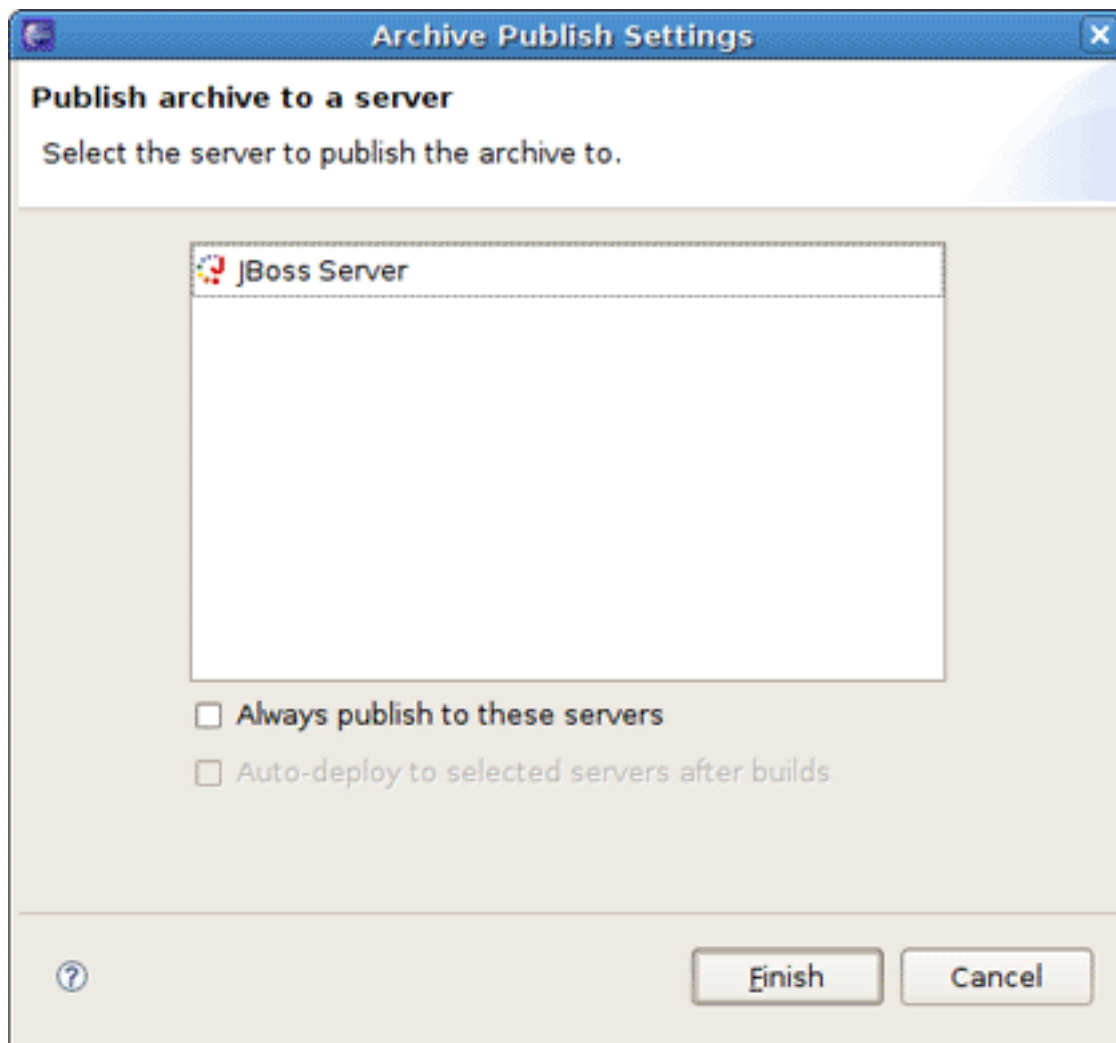


**Note:**

When editing an archive, it is also updated in all folders and other archives where it is nested.

### 7.1.4. Publishing to Server

Finally, you will need to publish your application to a server. This section describes how to do it with the help of the **Archives** View.



**Figure 7.13. Context Menu on the Item**

The dialog shown above appears after selecting the **Publish To Server** option. To publish once, select the server(s) that you want and click the **Finish** button. If you want the **Publish to Server** action on that particular Archive to always publish to that set of servers, then check the appropriate checkbox. To enable automatic publishing upon build events, check the last checkbox.

The automatic publishing feature is nice if, for example, your package's destination (where it is built) is a temporary folder and you want the archive published to several servers. If you only need your archive published to one server, it might be easier to have the archive's destination folder be the deploy folder of the server.



### 7.1.5. Relevant Resources Links

Refer to the [Ant manual](http://ant.apache.org/manual/index.html) [http://ant.apache.org/manual/index.html] to find more on how to build your applications using Ant.

We also recommend that you watch this [movie](http://docs.jboss.org/tools/movies/demos/archiving/archiving.htm) [http://docs.jboss.org/tools/movies/demos/archiving/archiving.htm] which demonstrates the powerful archiving functionality in JBoss Tools™.



# TPTP Support

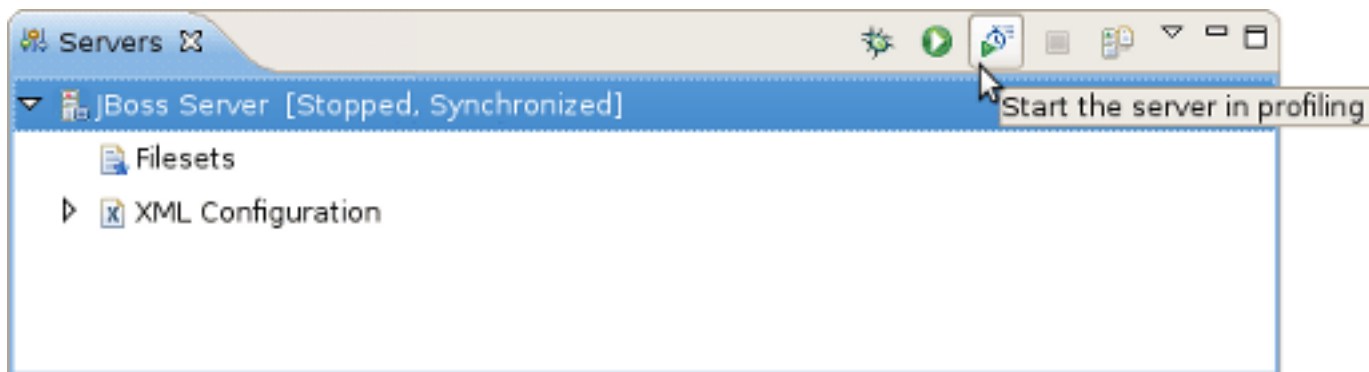
This chapter provides an overview on how to enable TPTP Profiling for JBoss AS™ adapters in JBoss Tools™.

## 8.1. TPTP Profiling

To get TPTP profiling work on JBoss Application Server™ you should do the following:

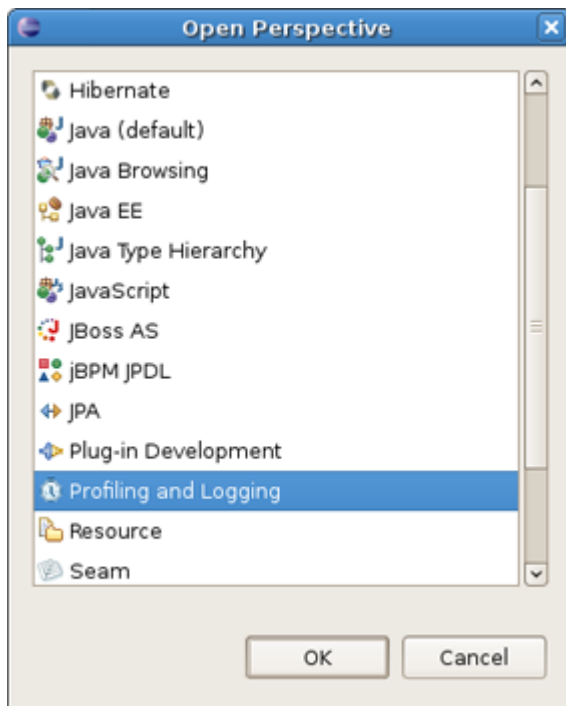
- Download [TPTP Runtime](http://www.eclipse.org/tptp/home/downloads/) [http://www.eclipse.org/tptp/home/downloads/] and install it, i. e. just add the content of *plugins/features* folders from downloaded directory to the same folders in your eclipse installation directory or use the **Help** → **Install New Software** command.
- Install JBoss TPTP Tools which provide TPTP support for JBoss AS servers (find the latest stable version of the JBoss TPTP profile feature at <http://www.jboss.org/tools/download/stable>).

And now all profile actions should work for you. To start JBoss AS™ in profiling mode use **Start the server in profiling mode** button or select **Profile As** → **Profile on Server** from the context menu of the project.



**Figure 8.1. Start the Server in Profiling mode**

To enable TPTP features in your workbench use Profiling and Logging Perspective that you can find in the list of proposed perspectives: **Window** → **Open Perspective** → **Other...**



**Figure 8.2. Profiling and Logging Perspective**

## 8.2. Relevant Resources Links

All additional information on TPTP (Test and Performance Tools Platform) can be found in the [Eclipse documentation](http://www.eclipse.org/tptp/home/downloads/4.5.0/documents/quicktour/quick_tour.html) [http://www.eclipse.org/tptp/home/downloads/4.5.0/documents/quicktour/quick\_tour.html].