



realtimepublishers.com<sup>tm</sup>

*The Developer  
Shortcut Guide<sup>tm</sup> To*



**SUSE  
LINUX**

**Novell<sup>®</sup>**

*John Featherly*

Chapter 3: JBoss Enterprise Applications.....	42
Running on JBoss .....	42
JBoss CMP .....	49
Configuring JBoss CMP Database.....	52
Web Services and JBoss .....	55
JBoss Enterprise Features .....	59
JBoss jBPM.....	59
JBoss Portal.....	62
Summary .....	66

## Copyright Statement

© 2005 Realtimedpublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimedpublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimedpublishers.com, Inc or its web site sponsors. In no event shall Realtimedpublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimedpublishers.com and the Realtimedpublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimedpublishers.com, please contact us via e-mail at [info@realtimedpublishers.com](mailto:info@realtimedpublishers.com).

[**Editor's Note:** This eBook was downloaded from Content Central. To download other eBooks on this topic, please visit <http://www.realtimepublishers.com/contentcentral/>.]

## Chapter 3: JBoss Enterprise Applications

In addition to developing applications using Eclipse and the JBoss IDE plug-in, which the previous chapter explored, the application needs to be run, tested, and debugged. This chapter will look at how to use JBoss to execute and debug applications, discuss in more detail the data source connection for Container Managed Persistence (CMP), and explore how to configure various database products as persistence engines. Finally, the chapter will look at JBoss Application Server (AS) structure and management and other JBoss features—Java Business Process Management (jBPM) and JBoss Portal.

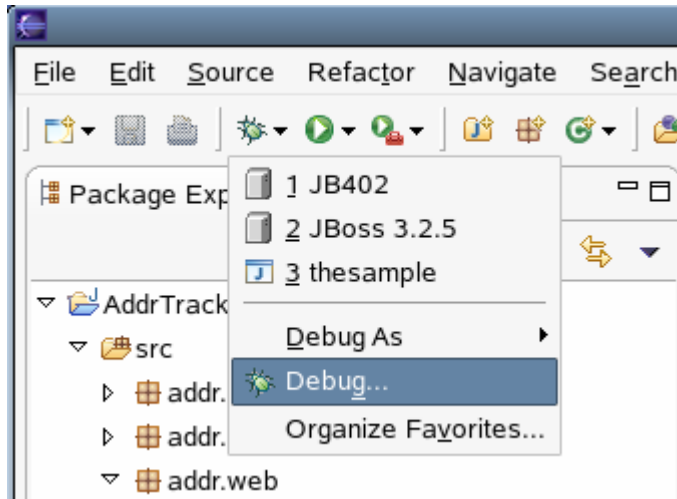
### Running on JBoss

The simplest method of running an application on JBoss is to take the application package, the .ear file (EAR stands for Enterprise ARchive), and copy it to the server folder of the JBoss application server. Chapter 2 explored how to build an .ear file. To test run the application, you simply need to start up a JBoss AS and drop in the application file.

You can use YaST to install the JBoss AS that is bundled in the SUSE Linux Professional 9.3 distribution. The 9.3 distribution contains JBoss AS version 3.2.5-6. YaST installs it to the destination `/usr/share/jboss`. SUSE has provided an `init.d` script for JBoss that you can use to start the server at boot time. You can also start the server manually with the sym-link to the `init.d` script: `/usr/sbin/rcjboss start`. These methods will typically be used on a production server; on a developer system, you will most likely want to control JBoss server startup inside the Eclipse environment. Controlling JBoss server from within the Eclipse IDE gives you debugger access to the running application.

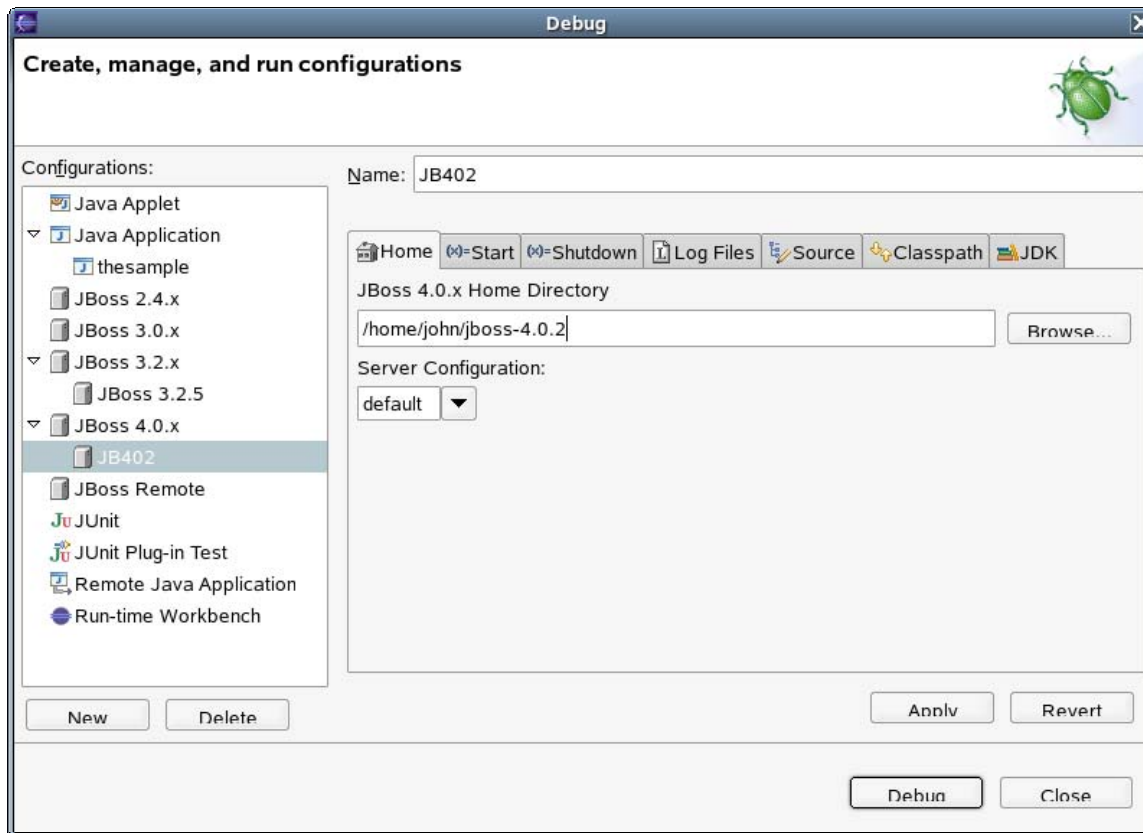
At the time of this writing, the current released version of JBoss AS is version 4.0.2. The package can be downloaded from <http://www.jboss.com/products/jbossas/downloads>. The package is platform independent, and you can choose from three archive types (all contain the same bits): `.tar.bz2`, `tar.gz`, or `.zip`. Pick any one, download it, and unpack it to a local folder (for example, `tar xzvf jboss-4.0.2.tar.gz` for the `gzip` file) on your development machine.

You can configure JBoss to run in Eclipse by creating a launch configuration in the Debug dialog box. To open the dialog box, select Debug from the Eclipse bug menu as Figure 3.1 shows.



**Figure 3.1:** The Eclipse bug menu.

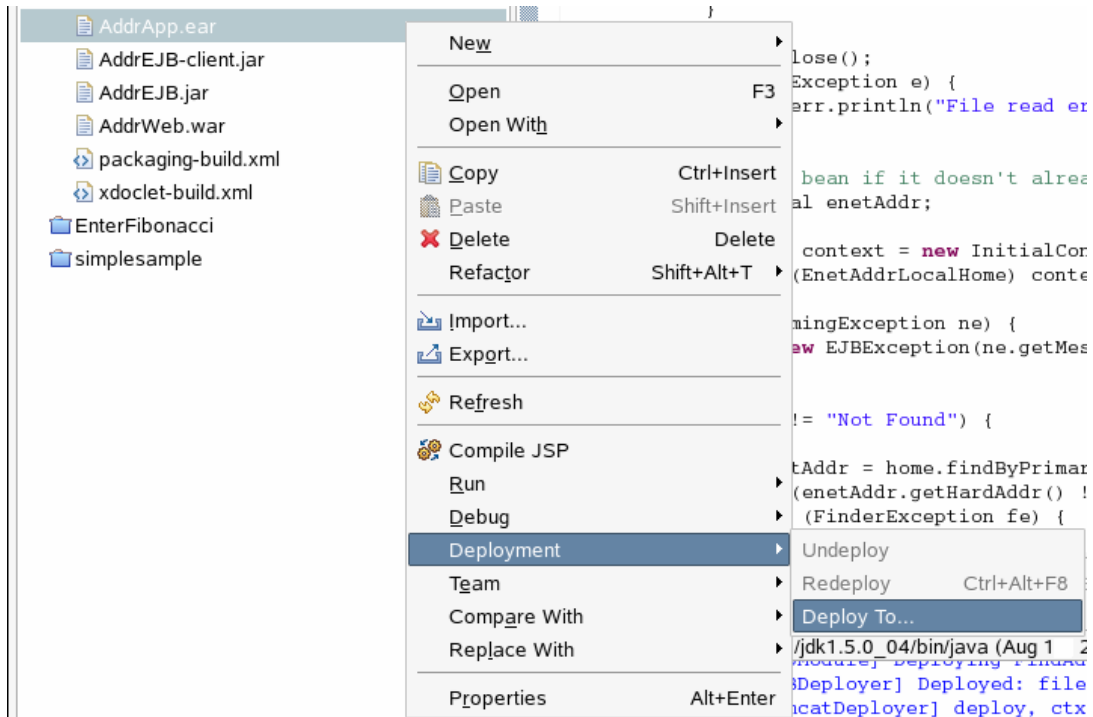
The key information to define for the configuration is the path to the folder you created when you downloaded and unpacked the JBoss AS package. Select the JBoss 4.0.x configuration group in the list on the left side of the dialog box, and click New. Enter a configuration name and the path for the JBoss home directory. You will also need to add the Java project for your application to the source list, which you can do from the Source tab in the dialog box. Figure 3.2 shows the run configuration dialog box.



**Figure 3.2: Defining the JBoss run configuration.**

JBoss allows you to create custom sever configurations, any of which will appear in the drop-down list in the dialog box that Figure 3.2 shows. Server configurations provide a mechanism to define functional server components, security, and application content into multiple sets. When you run the server, you simply specify which of the configuration definitions you want to use. Usually, you will use the predefined server configuration called “default.” Click Apply to save the configuration. Once the launch configuration has been created and saved, the JBoss server can be started by clicking Debug at the bottom right of the dialog box. The server will start up with the run time information messages appearing in the Console window in Eclipse.

With JBoss AS running, you can deploy and run your application. As mentioned at the beginning of this section, you can simply copy the application package .ear file to the deployment folder of the JBoss server (which for the default server is jboss-4.0.2/server/default/deploy). The Eclipse IDE provides a more convenient way of doing the same thing. Right-click the .ear file in the Eclipse Package Explorer, and select Deployment, Deploy To as Figure 3.3 shows.



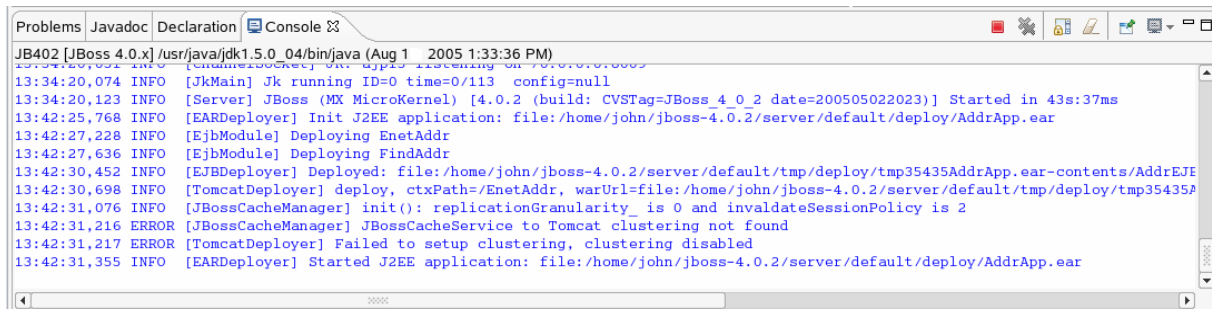
**Figure 3.3: Deploying the application .ear file.**

Select the target (launch configuration) from the available list (see Figure 3.4).



**Figure 3.4: The launch configuration list.**

The application .ear file will be copied to the deployment directory and JBoss AS will pick it up and unpack it to start it running. The JBoss deployment messages can be seen in the Eclipse Console window, as Figure 3.5 shows.



**Figure 3.5:** JBoss deployment information shown in the Eclipse Console window.

The application is now running, to access it use the URL <http://localhost:8080/EnetAddr> in a Web browser (see Figure 3.6).



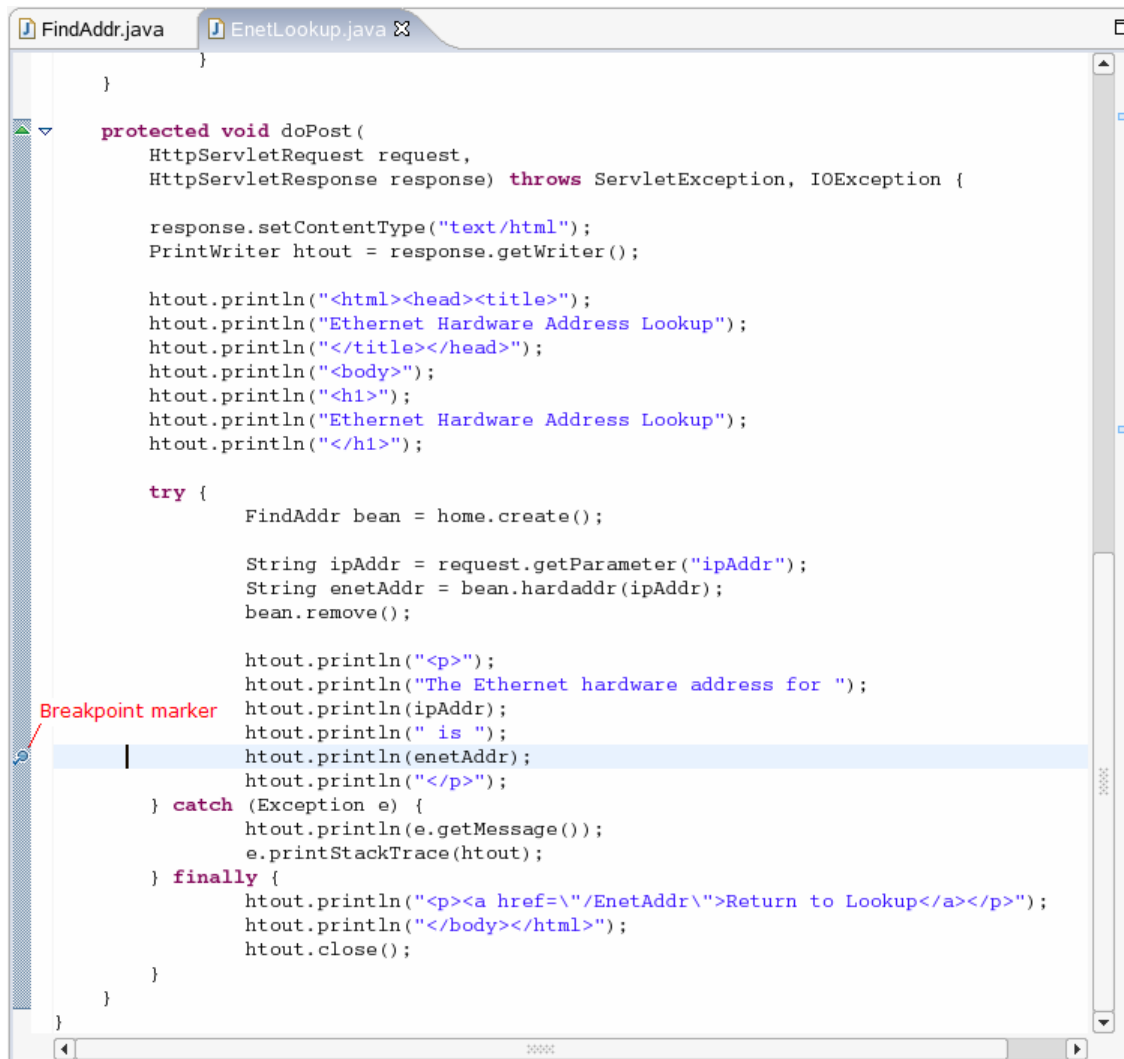
**Figure 3.6:** JBoss running the application.

With the application deployed and running, first check that everything is working by entering an IP address and clicking Lookup. Set a breakpoint at line 83 in the EnetLookup.java module, the line contains the code

```
htout.println(enetAddr)
```

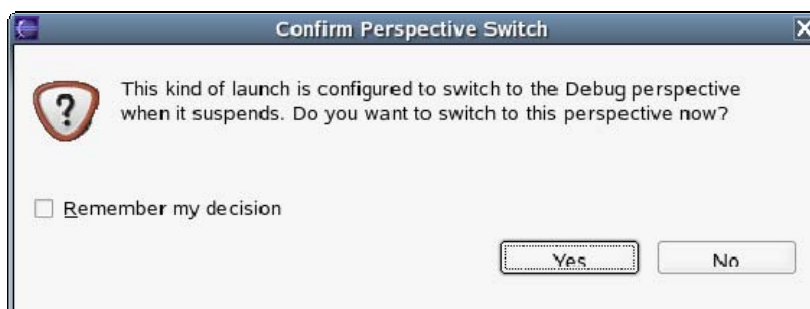
You can set a breakpoint by double-clicking in the gray margin on the left side of the code window; a breakpoint marker appears in the margin (see Figure 3.7).





**Figure 3.7: Setting a breakpoint.**

Execute a lookup by entering an IP address in the text box, and clicking LookUp. Eclipse will switch from the Java perspective to the Debug perspective. An information box will pop up to let you know Eclipse is making the switch, as Figure 3.8 shows.



**Figure 3.8: Java switching to the Debug perspective.**

In the Debug perspective, the code window is open with the breakpoint line highlighted. A debug window shows the call stack and has buttons to resume, pause, terminate, and step. There is also a window showing variables and values. Figure 3.9 shows the Debug perspective in Eclipse—notice the Variables window in the upper right.

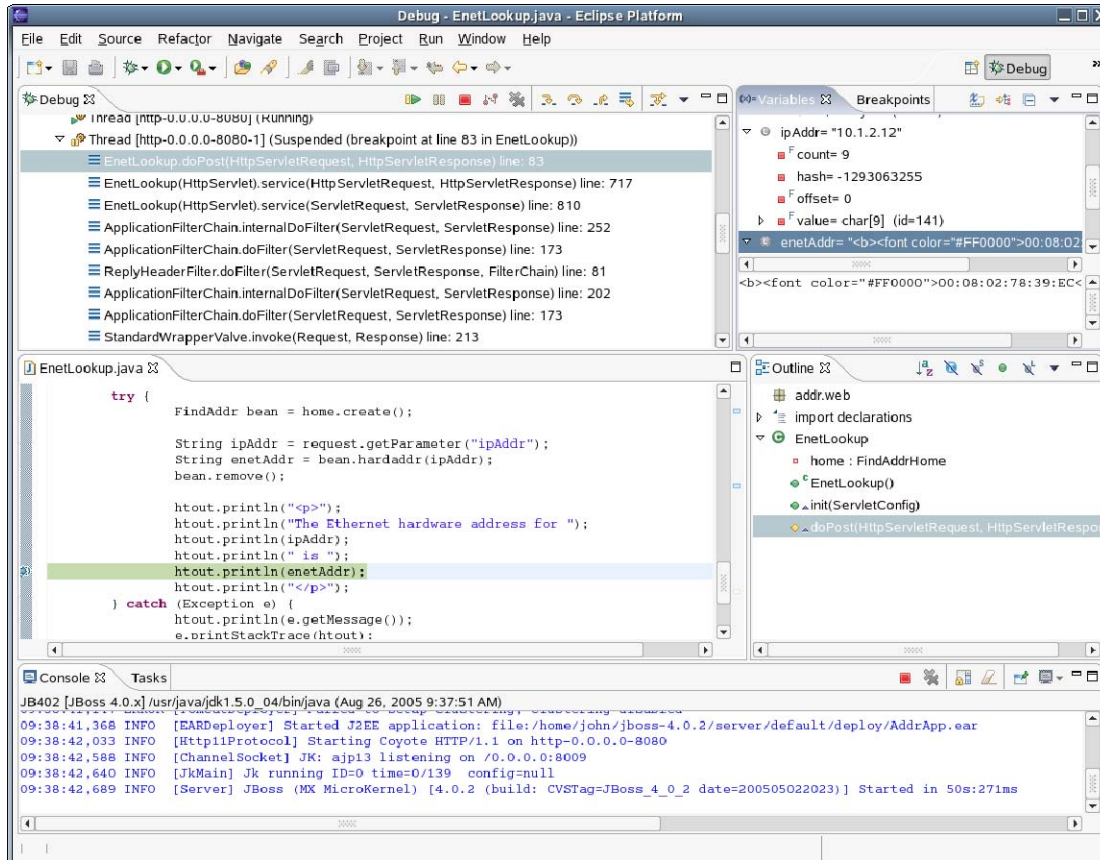


Figure 3.9: The Debug perspective.

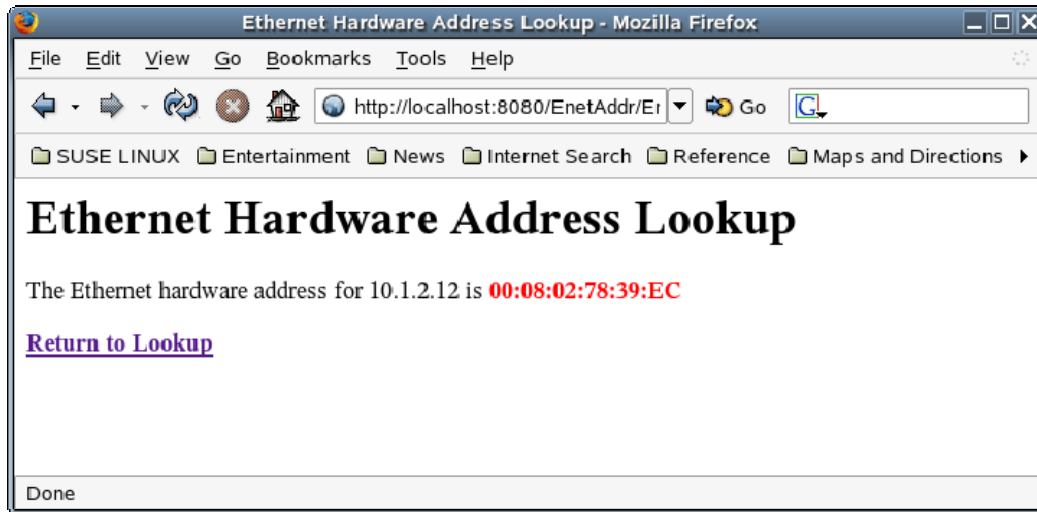
Right-click `enetAddr` and select Change Value from the menu. Next, add HTML bold and color tags:

```
<b><font color="#FF0000">
```

before and

```
</font></b>
```

after the string value. Then click Resume (small green “play” arrow) on the debug window. The application continues and displays the new result, as Figure 3.10 shows.



**Figure 3.10:** Result of changing a variable in the debugger.

Once you have the launch configuration set up, it is easy to startup JBoss AS and deploy, run, and debug applications. It is also easy to define multiple configurations for different versions of JBoss to test your application for different potential production environments.

## JBoss CMP

In the example application that has been used throughout this guide, the entity bean (EnetAddr) is set up to use CMP. Relational database systems are used as the data storage engines for CMP and the container system (JBoss AS) takes care of all the detail and internal mechanics of storing and retrieving object data. In addition to data storage, the CMP system can also reconcile some basic object data models represented by entity beans and relational data models represented by relational database systems. For more complex data models, a full object/relational mapping (O/RM) system such as Hibernate is needed. The EJB 3.0 specification defines an OR/M standard that is implemented in JBoss 4.0.2 using Hibernate3. Hibernate, in turn, is able to use just about any relational database system (RDBMS) such as MySQL, PostgreSQL, and so on for relational data storage.

The default database system JBoss uses for CMP is Hypersonic. Hypersonic has a useful database manager application that you can use to view, create, and update data and metadata. To run the database manager application, use the Hypersonic MBean in the JBoss JMX console. To access the JBoss JMX console use the URL <http://localhost:8080/jmx-console>. Figure 3.11 shows the console and the link for the Hypersonic MBean.

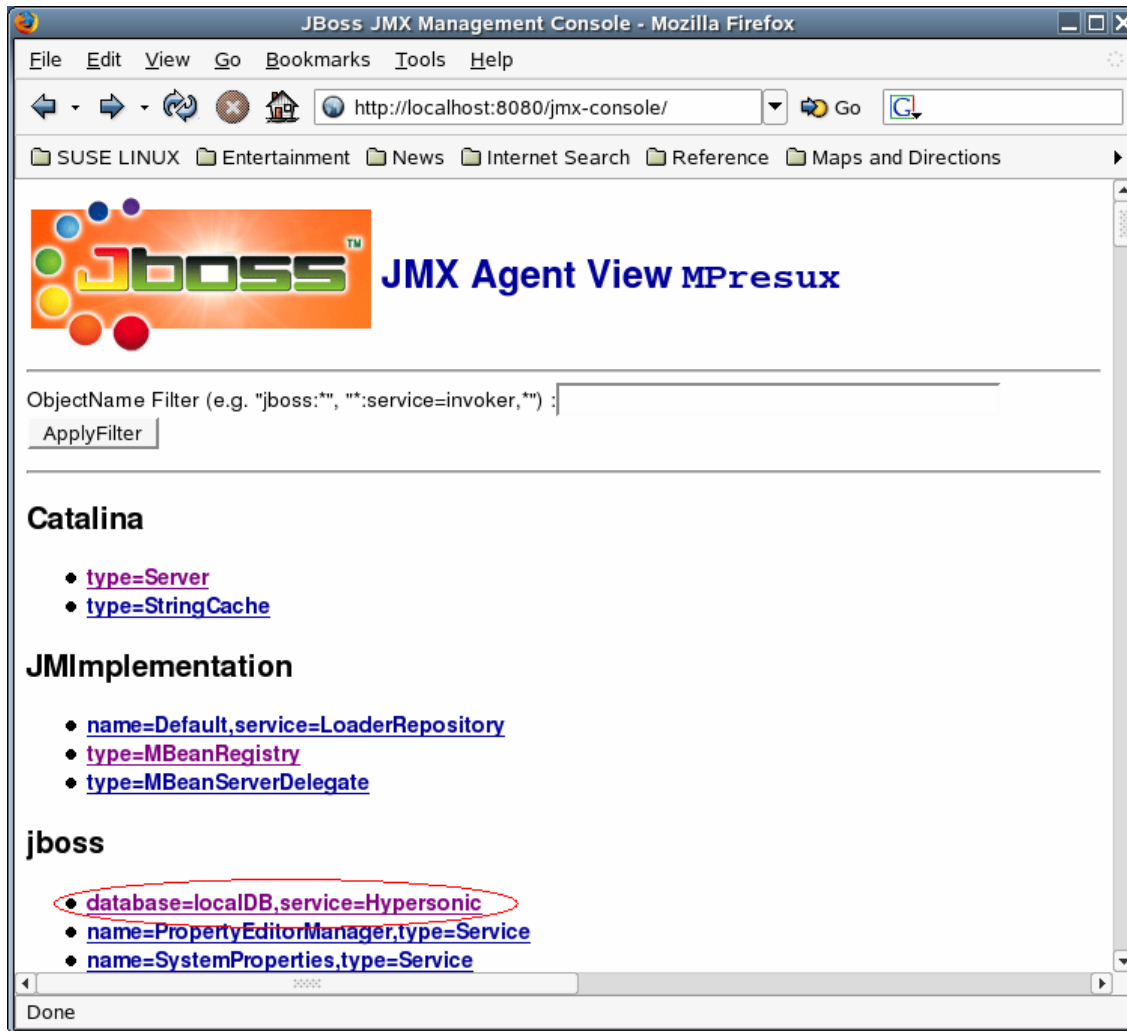
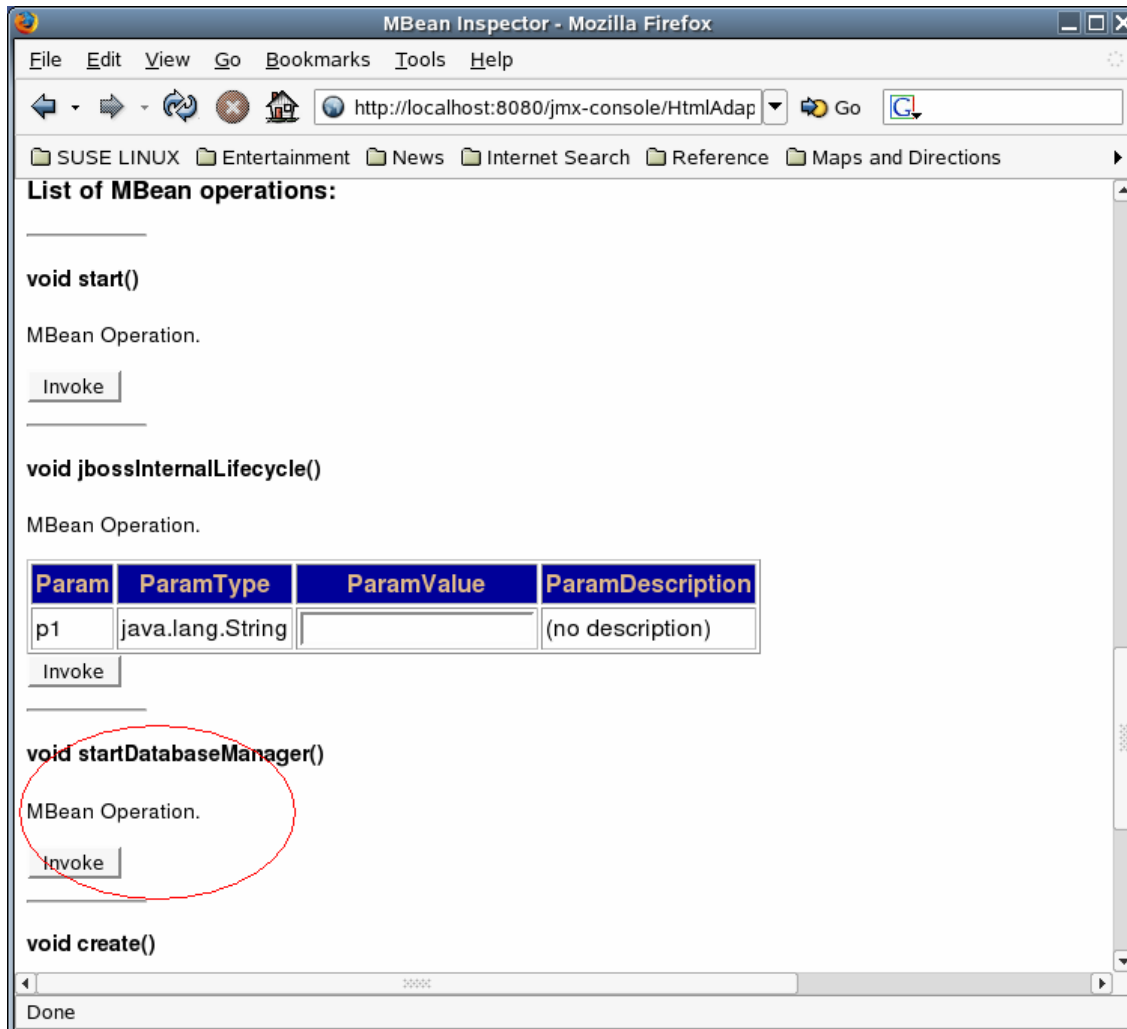


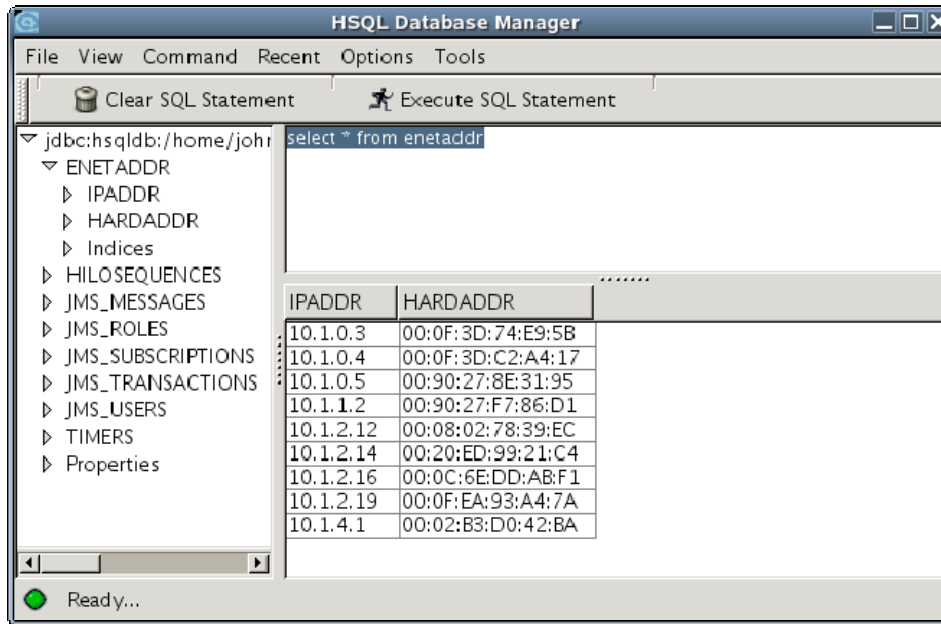
Figure 3.11: Hypersonic MBean link in JMX console.

Click the link and the MBean page for Hypersonic opens. There is a `startDatabaseManager()` operation in the list of MBean operations that opens the database manager application. Figure 3.12 shows the operation page.



**Figure 3.12: Hypersonic startDatabaseManager() operation.**

Click Invoke to start the database manager. The EnetAddr entity bean from the sample application has a table in the database with any persisted bean values stored as rows in the table. Figure 3.13 shows the database manager and the EnetAddr table.



**Figure 3.13:** EnetAddr table shown in Hypersonic.

You might need or be required to use a database system other than Hypersonic for CMP data storage. For example, let's look at how to configure MySQL as the CMP data storage engine for the application.

### Configuring JBoss CMP Database

The first step is to install and set up MySQL. The SUSE Linux Professional 9.3 distribution contains the latest MySQL 4.1 package. Install it using YaST; the database package is in the Productivity/Databases/Servers package group. Select the package `mysql` and optionally `mysql-administrator`. Also select `mysql-client` in the package group Productivity/Databases/Clients and `mysql-connector-java` in Development/Libraries/Java. After the installation is complete, start the database daemon using

```
> /usr/sbin/rcmysql start
```

Next, run the MySQL command-line tool to set up the database for your application. At a command line, enter

```
> mysql -u root -p
```

Create the `jboss` database and set the password for `jboss@localhost` as Listing 3.1 shows.

```

john@MPresux:~> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.1.10a

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+-----+
| Database |
+-----+
| mysql    |
| test     |
+-----+
2 rows in set (0.00 sec)

mysql> create database jboss;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| jboss    |
| mysql    |
| test     |
+-----+
3 rows in set (0.00 sec)

mysql> grant all privileges on jboss.* to jboss@localhost identified by
'password';
Query OK, 0 rows affected (0.01 sec)

mysql> select User,Host,Password from mysql.user;
+-----+-----+-----+

```

**Listing 3.1: Creating MySQL database for JBoss.**

Next, you need to copy the MySQL connector jar file to the JBoss server directory. YaST installs the jar file in the folder /usr/share/java. A copy command similar to the following will work:

```

> cp /usr/share/java/mysql-connector-java-3.1.6.jar \
    ~/jboss-4.0.2/server/default/lib

```

Do the copy when the JBoss server is not running. Also, before starting the JBoss server, create a data source definition by placing a file with the name mysql-ds.xml in the ~/jboss-4.0.2/server/default/deploy directory. The contents of the file are as Listing 3.2 shows.

```

<datasources>
  <local-tx-datasource>
    <jndi-name>MySQLDS</jndi-name>
    <connection-url>jdbc:mysql://localhost:3306/jboss</connection-
url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <user-name>jboss</user-name>
    <password>password</password>

```

**Listing 3.2: The mysql-ds.xml file.**

Next, you need to configure your application to use the MySQL data source. The setting is in the `jbosscmp-jdbc.xml` file, but that file is maintained by XDoclet in the application. Open the XDoclet configuration for `AddrEJB`, and change the `jboss` entry for the two properties `datasource` and `datasourceMapping`. Set the value for `datasource` to `java:/MySQLDS` and the value for `datasourceMapping` to `mysql`. Then run XDoclet and run Packaging to rebuild the application `.ear` file. Finally, right-click the `.ear` file and deploy/re-deploy it to the JBoss server. Access the application and lookup some IP addresses. When you go back into MySQL, you will see that a new table `EnetAddr` has been created and there are some data rows (see Listing 3.3).

```

john@MPresux:~> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9 to server version: 4.1.10a

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+-----+
| Database |
+-----+
| jboss   |
| mysql   |
| test    |
+-----+
3 rows in set (0.00 sec)

mysql> use jboss;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_jboss |
+-----+
| EnetAddr         |

```

**Listing 3.3: Application CMP data in MySQL.**

Other database systems such as postgresql could also be used as data sources using a similar approach to setting up the database and defining the data source.



## Web Services and JBoss

There is currently a transition state (JBoss AS version 4.0.2) for Web Services support in JBoss. JBoss Web Services (JBossWS) is a sub-project of JBoss AS to provide J2EE-compliant web services. Previous 3.x versions of JBoss used JBoss.NET web services, which will not be developed further and are not recommended for new projects. JBossWS uses Apache Axis technology to implement web services but does not directly use the Axis release. JBoss modified the Axis code to be J2EE 1.4 compliant and is implemented in `org.jboss.axis`. The modified implementation is also WS-I Basic Profile 1.0 compliant. JBoss is moving to an independent implementation using JSR-181 Java metadata, a new SOAP stack and web services toolset, in the near future.

Web service endpoints can be created for servlets (Java objects) and EJB stateless session beans. The application example has a stateless session bean—`FindAddr`. Let's look at the steps to define a web service endpoint for this EJB bean. The only new coding that has to be done is to define the interface for the endpoint. In the Eclipse Package Explorer, right-click the `addr.interfaces` package and select New, Interface. Name the interface `FindAddrEndpoint`, and click Finish. Enter the code that Listing 3.4 shows for the interface definition.

```
package addr.interfaces

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface FindAddrEndpoint extends Remote {
    public String hardaddr(String ipAddr) throws RemoteException;
}
```

**Listing 3.4:** Web service endpoint interface for `FindAddr`.

The rest of the work is now in building the deployment artifacts. As Chapter 2 explored, XDoclet is an effective tool for creating and managing deployment artifacts. There is some limited capability to use XDoclet for web services artifacts, but it is problematic and will be obsolete when JBossWS implements JSR-181. For an example, this chapter will see what is involved in creating the deployment pieces manually. You need `webservices.xml`, WSDL for the service, and JAX-RPC mapping file. The recommended method for creating the WSDL and mapping file is to use the `wscompile` tool from the Java Web Services Developer Pack (JWSDP), which you can get from Sun Microsystems at <http://java.sun.com/webservices/downloads/webservicespack.html>. To run `wscompile`, you need to have the Java class file for the endpoint interface and a small configuration file as input. For the `FindAddr` web service definition, use the `config.xml` file that Listing 3.5 shows.


```
<?xml version="1.0" encoding="UTF-8"?>

<configuration xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
  <service name="FindAddrService"
    targetNamespace="http://interfaces.addr"
    typeNamespace="http://interfaces.addr/types"
    packageName="addr.interfaces">
    <interface name="addr.interfaces.FindAddrEndpoint"
      servantName="addr.ejb.FindAddr"/>
  </service>
</configuration>
```

**Listing 3.5: Config.xml file for wscompile.**

Next, to generate the WSDL and mapping file, run wscompile using the command line:

```
wscompile.sh -classpath ../AddrTrack/bin -gen:server -
f:rpcliteral -mapping mapping.xml config.xml
```

 JBossWS supports both RPC and Document encoding, but Document style endpoints are not fully recommended until the upcoming JBossWS release supports the WS-I Attachment Profile.

Use the appropriate path to the application class files for the `-classpath` argument. Place the `mapping.xml` file in the `META-INF` folder and the `FindAddrService.wsdl` file in the `META-INF/wSDL` folder. The service endpoint then needs to be identified for the bean in the `ejb-jar.xml` file. Add the following line to the `ejb-jar.xml` file in the section for the session bean `FindAddr`:

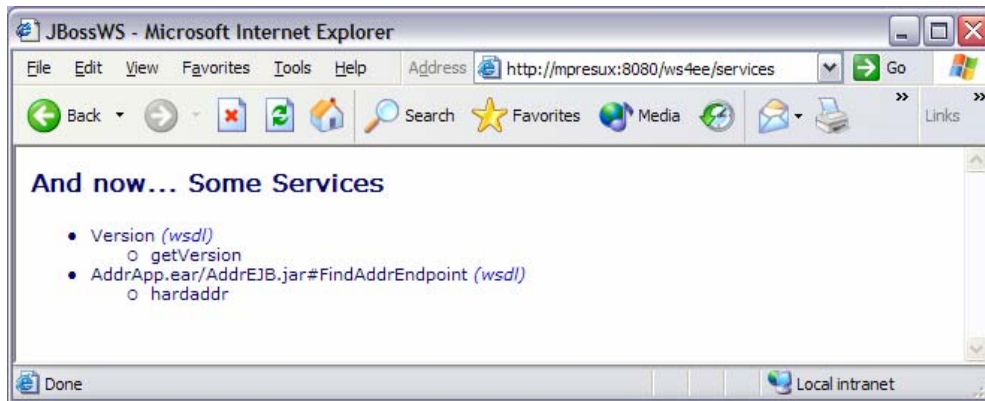
```
<service-endpoint>addr.interfaces.FindAddrEndpoint</service-
endpoint>
```

Next, create the `webservices.xml` file in the `META-INF` folder. Use the contents that Listing 3.6 shows.

```
<webservices xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://www.ibm.com/webservices/xsd/j2ee_web_services_services_1_1.xsd"
  version="1.1">
  <webservice-description>
    <webservice-description-name>
      FindAddrService
    </webservice-description-name>
    <wsdl-file>META-INF/wSDL/FindAddrService.wsdl</wsdl-file>
    <jaxrpc-mapping-file>META-INF/mapping.xml</jaxrpc-mapping-file>
    <port-component>
      <port-component-name>FindAddrEndpoint</port-component-name>
      <wsdl-port>FindAddrEndpointPort</wsdl-port>
      <service-endpoint-interface>
        addr.interfaces.FindAddrEndpoint
      </service-endpoint-interface>
    </port-component>
  </webservice-description>
</webservices>
```

**Listing 3.6: The webservices.xml file.**

Finally, to package and deploy the application with the new web service definition, add the three files FindAddrService.wsdl, webservices.xml, and mapping.xml to the package definition for AddrEJB.jar using the technique from Chapter 2. Then run packaging for the project and deploy (or re-deploy) to the running JBoss server. You can monitor the application deployment and information messages from [WSDLFilePublisher] and [AxisService] in the Eclipse Console window. Use the URL <http://localhost:8080/ws4ee/services> (or machine name in place of localhost if you browse from a different machine) to check whether the web service is registered (see Figure 3.14).



**Figure 3.14:** JBossWS service list.

You can then click on the wsdl link for hardaddr to view the published WSDL, as Figure 3.15 shows.

```

<?xml version="1.0" encoding="UTF-8" ?>
- <definitions name="FindAddrService" targetNamespace="http://interfaces.addr"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://interfaces.addr"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <types />
  - <message name="FindAddrEndpoint_hardaddrResponse">
    <part name="result" type="xsd:string" />
  </message>
  - <message name="FindAddrEndpoint_hardaddr">
    <part name="String_1" type="xsd:string" />
  </message>
  - <portType name="FindAddrEndpoint">
    - <operation name="hardaddr" parameterOrder="String_1">
      <input message="tns:FindAddrEndpoint_hardaddr" />
      <output message="tns:FindAddrEndpoint_hardaddrResponse" />
    </operation>
  </portType>
  - <binding name="FindAddrEndpointBinding" type="tns:FindAddrEndpoint">
    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http" />
    - <operation name="hardaddr">
      <soap:operation soapAction="" />
      - <input>
        <soap:body namespace="http://interfaces.addr" use="literal" />
      </input>
      - <output>
        <soap:body namespace="http://interfaces.addr" use="literal" />
      </output>
    </operation>
  </binding>
  - <service name="FindAddrService">
    - <port binding="tns:FindAddrEndpointBinding" name="FindAddrEndpointPort">
      <soap:address
        location="http://MPresux:8080/AddrApp/AddrEJB/FindAddrEndpoint" />
    </port>
  </service>
</definitions>

```

Figure 3.15: WSDL from JBoss server.

There are a few steps involved in setting up web services for JBoss applications, but those steps are fairly simple and straightforward. Based on the JBoss roadmap, you can expect this process to be even more straightforward in an upcoming release.

## JBoss Enterprise Features

In addition to the application server, JBoss offers a number of other powerful capabilities. Hibernate and EJB 3.0 have already been mentioned—jBPM and JBoss Portal are notable others.

### JBoss jBPM

JBoss jBPM is a workflow/BPM package that can be easily integrated with J2EE applications. jBPM provides a process designer integrated as an Eclipse plug-in and a process execution engine. jBPM can persist (dehydrate and store) a running process using the Hibernate storage engine. It is easy to get started using jBPM—simply download the engine and the process designer plug-in from <http://www.jboss.com/products/jbpm/downloads>.

To get started with the Eclipse plug-in, unzip the archive `jpbm-gpd-feature-3.0.1.zip` into the top directory of your eclipse installation. You might want to use a new private installation of Eclipse to test things out first. To do so, start Eclipse and create a new project by selecting File, New, Project. The New Project dialog box will open; in it, select Process Project from the JBoss jBPM category, as Figure 3.16 shows.

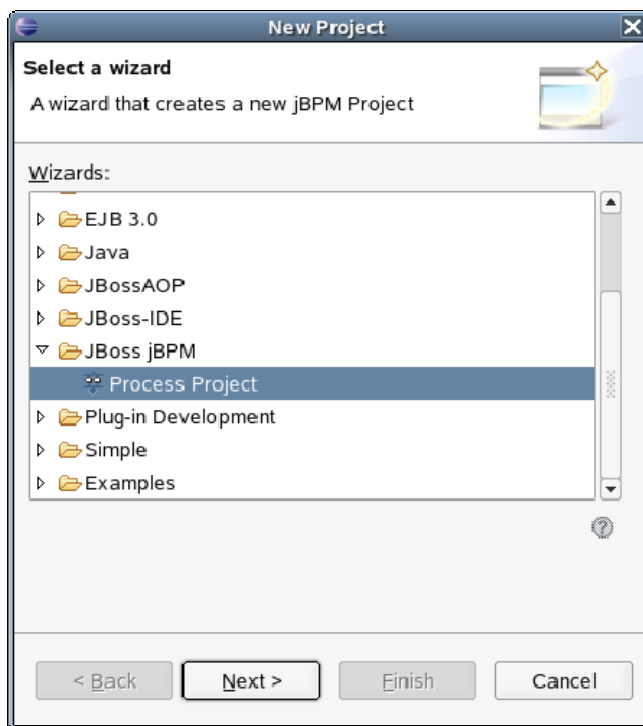
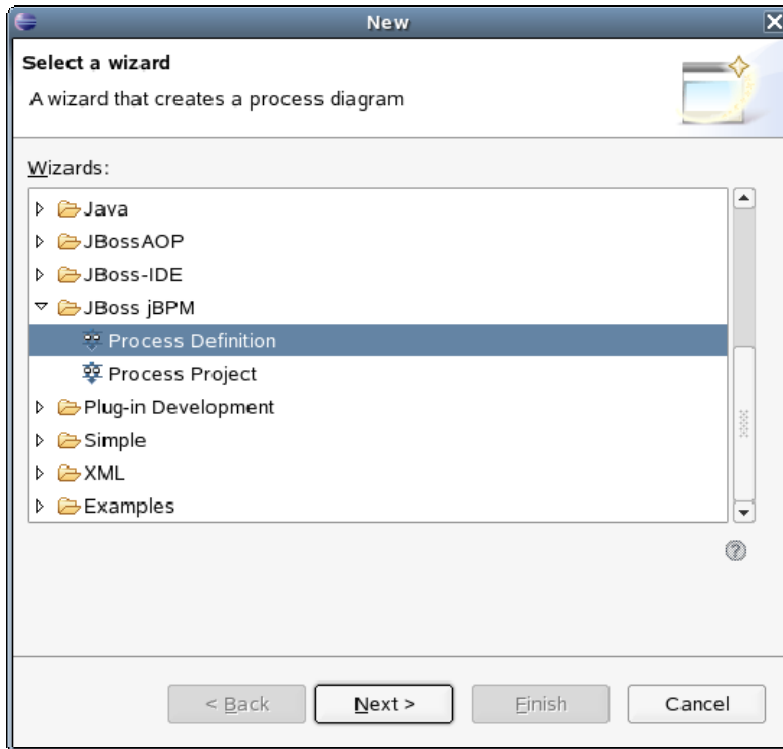


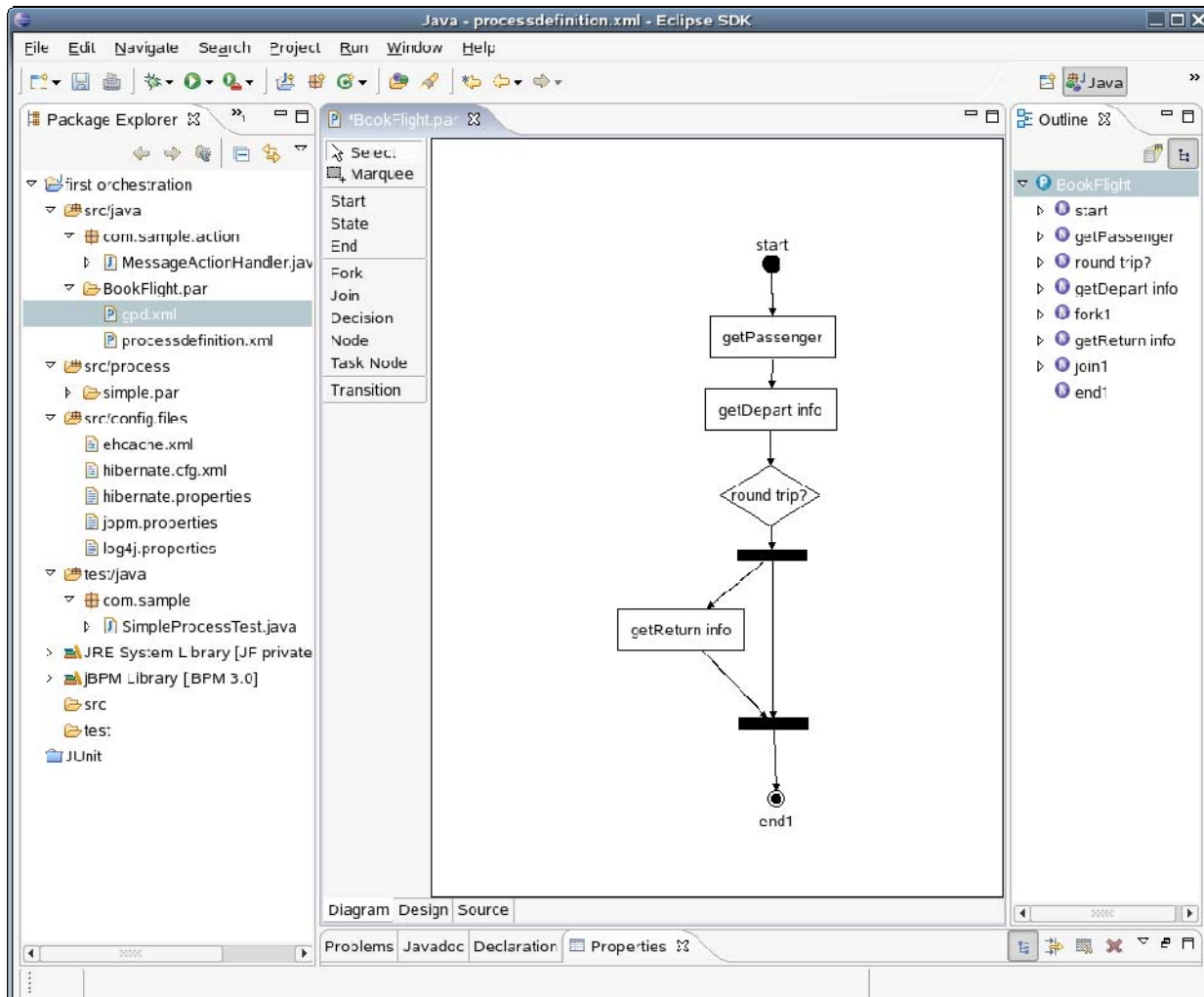
Figure 3.16: New jBPM project.

Once a project has been created, open the process designer by creating a new process definition. Right-click the project name in Package Explorer, and select New, Other; the new selection dialog box will open. Pick Process Definition from the JBoss jBPM category (see Figure 3.17).



**Figure 3.17:** The new jBPM process definition dialog box.

The process designer opens the process definition in a graphical design window with a toolbox to allow you to place graphical representations of the process steps on the design surface. Figure 3.18 shows the results of an example process definition for flight booking.



**Figure 3.18: JBoss jBPM process design.**

The jBPM engine distribution package includes several samples predefined as Eclipse projects. To investigate the samples, import the project into Eclipse using File, Import; select the Existing Projects into Workspace wizard, then browse to the jBPM folder in the unpacked download.

Of course, the jBPM needs to have an execution environment to run. The distribution includes a predefined JBoss AS 4.0.2 server that you can start and manage just like any JBoss application server. Start it from a command line using

```
~/jbpn-top/jbpn-server/bin/run.sh -c jbpn
```

You can also create a launch configuration inside Eclipse just as earlier in this chapter. There is a preliminary jBPM console application bundled in the server. Once the server is running, access the console using <http://localhost:8080/jbpn>. Check the JBoss Web site at <http://www.jboss.org/products/jbpn> for downloads, documentation, and further information about jBPM.

## JBoss Portal

JBoss Portal is a full-featured Web portal product that integrates with JBoss AS. You will find all the expected features such as user management, content management, security management, themes, and forums. To get started using JBoss Portal download it from <http://www.jboss.com/products/jbossportal/downloads>.

JBoss Portal uses JBoss AS and Hibernate3, so you should have a 4.0.2 JBoss application server set up on which to install JBoss Portal. Hibernate can use any database engine. The following process walks you through a sample setup using MySQL. Setup MySQL data connector the same way you did earlier in this chapter. Install MySQL and MySQL Connector/J, copy the connector jar file to the server lib folder, and create a database and user in MySQL for JBoss Portal. Listing 3.7 shows a typical MySQL setup.

```
john@MPresux:~> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10 to server version: 4.1.10a

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+-----+
| Database |
+-----+
| jboss    |
| mysql    |
| test     |
+-----+
3 rows in set (0.07 sec)

mysql> create database jbossportal;
Query OK, 1 row affected (0.07 sec)

mysql> show databases;
+-----+
| Database      |
+-----+
| jboss         |
| jbossportal   |
+-----+
```

**Listing 3.7: MySQL setup for JBoss Portal.**

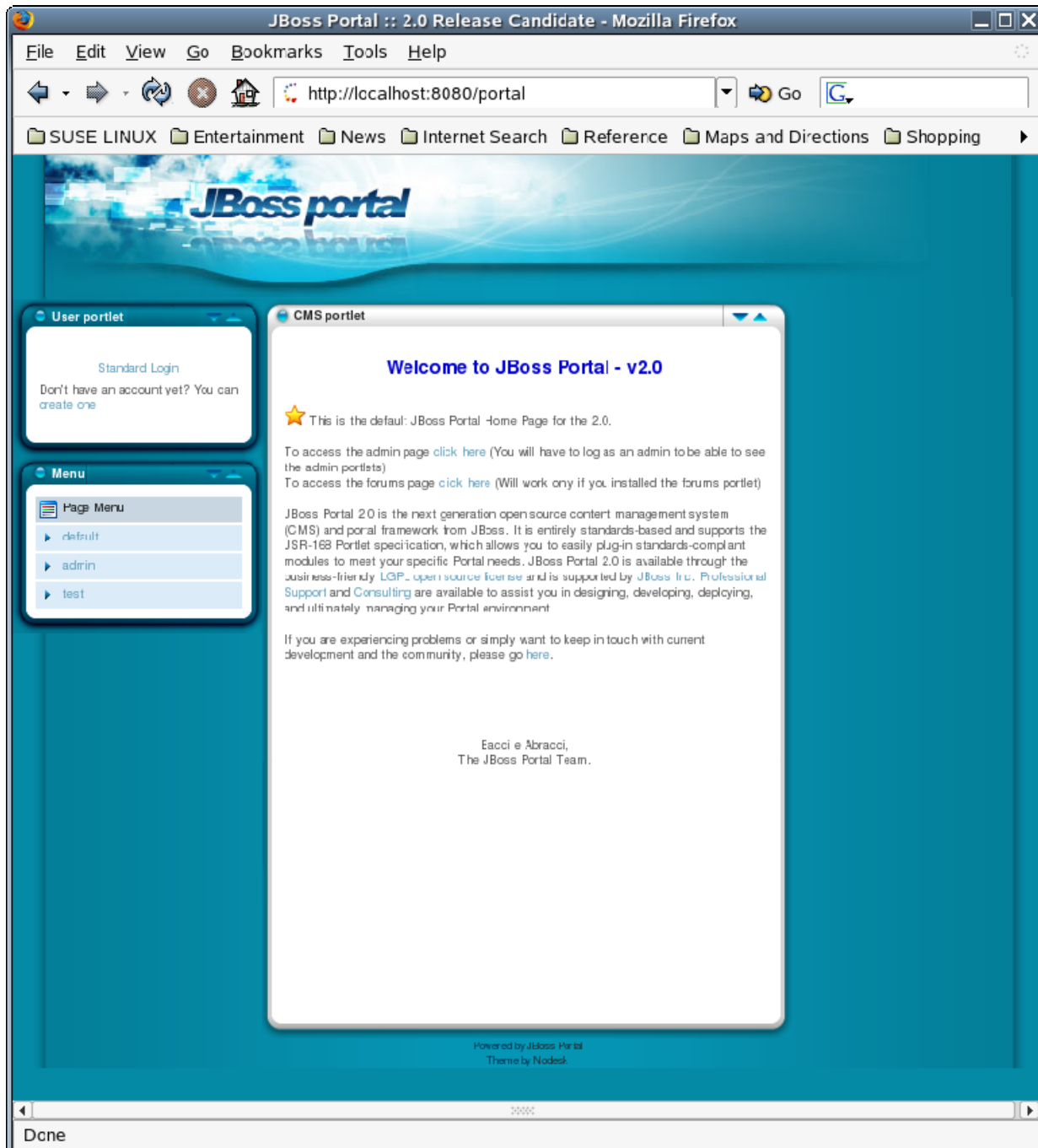
The corresponding MySQL data source definition is done in the file `portal-mysql-ds.xml`. Use the contents that Listing 3.8 shows to match the MySQL setup from Listing 3.7.



```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>PortalDS</jndi-name>
    <connection-url>
jdbc:mysql://localhost:3306/jbossportal?useServerPrepStmts=false
    </connection-url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <user-name>portal</user-name>
    <password>pwdportal</password>
  </local-tx-datasource>
```

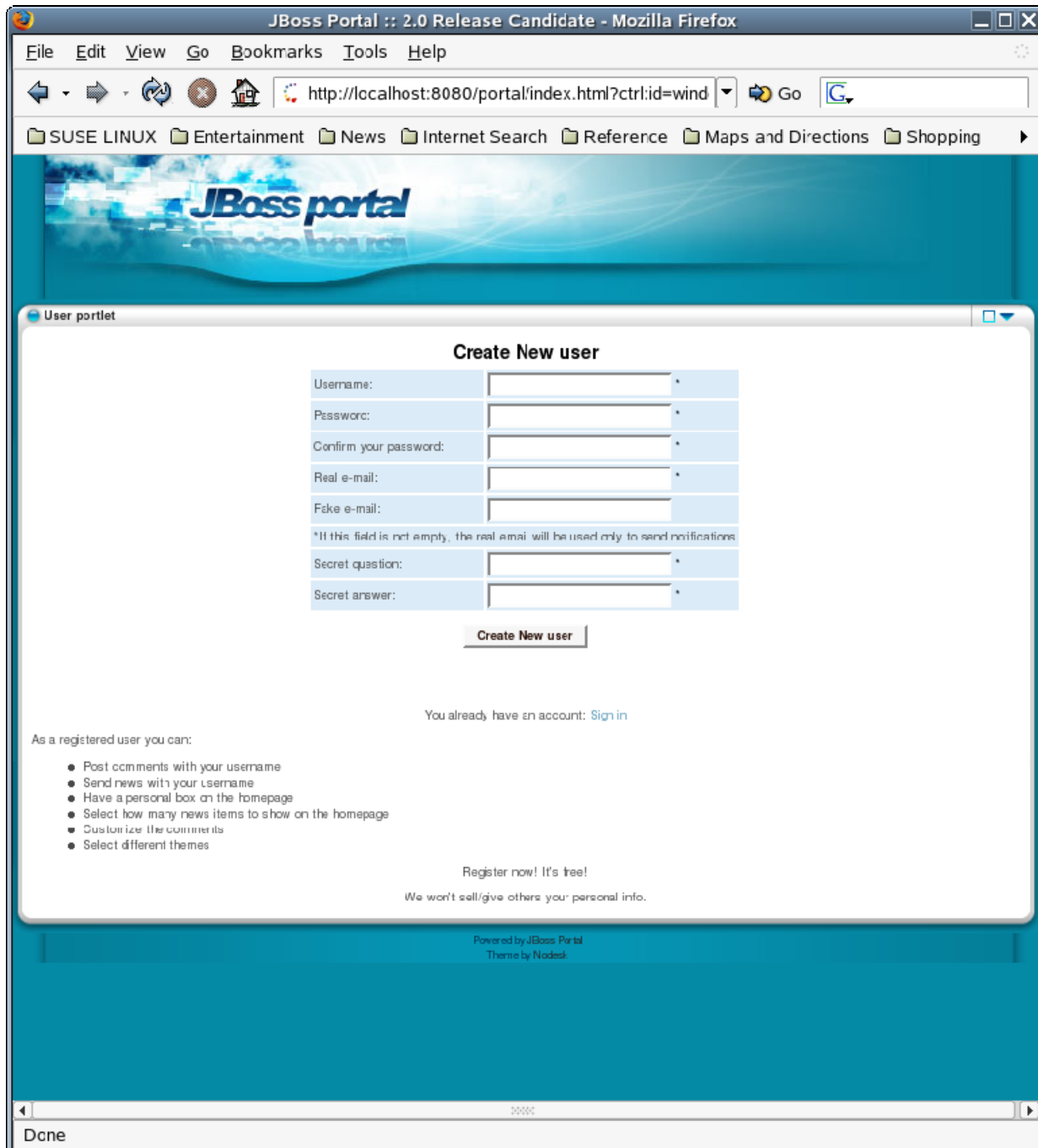
**Listing 3.8: Data source definition portal-mysql-ds.xml.**

Start the JBoss Portal in the usual way by launching the hosting JBoss AS with run.sh. Access the portal home page using the default URL <http://localhost:8080/portal>, see Figure 3.19.



**Figure 3.19:** The JBoss Portal.

JBoss Portal provides a complete set of user management tools including user self registration. Figure 3.20 shows the new user registration page.



**Figure 3.20:** The JBoss Portal new user registration.

The Portal data tables in MySQL will be created in the `jbossportal` database when JBoss Portal first runs. Listing 3.9 shows the initial list of tables in MySQL.

```
mysql> use jbossportal;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_jbossportal |
+-----+
| jbp_role_membership   |
| jbp_roles              |
| jbp_user_pref          |
| jbp_user_pref_prop_value |
| jbp_user_pref_set     |
+-----+
```

**Listing 3.9: JBoss Portal MySQL Data Tables.**

Once you have JBoss Portal up and running, you will need to configure content, user settings, and so on. Refer to the Portal documents available at <http://www.jboss.com/products/jbossportal/docs> for more information and user documents.

## Summary

This chapter explored the techniques for deploying, running, and debugging J2EE applications on JBoss AS. In addition, the chapter covered web services access to a J2EE application session bean and the method for configuring alternative database systems for JBoss CMP.

Hopefully, this chapter's discussion as well as the guide as a whole has helped you get a quick start on writing open source-based enterprise applications. Good luck!

## Content Central

[Content Central](#) is your complete source for IT learning. Whether you need the most current information for managing your Windows enterprise, implementing security measures on your network, learning about new development tools for Windows and Linux, or deploying new enterprise software solutions, [Content Central](#) offers the latest instruction on the topics that are most important to the IT professional. Browse our extensive collection of eBooks and video guides and start building your own personal IT library today!

## Download Additional eBooks!

If you found this eBook to be informative, then please visit Content Central and download other eBooks on this topic. If you are not already a registered user of Content Central, please take a moment to register in order to gain free access to other great IT eBooks and video guides. Please visit: <http://www.realtimepublishers.com/contentcentral/>.