Deploying Java CAPS Projects

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–3387–10
June 2008

080711@20490

Contents

CHAPTER 1

Deploying Java CAPS Projects

The topics listed here provide information about how to deploy Sun Java™ Composite Application Platform Suite (Java CAPS) Repository-based projects.

If you have any questions or problems, see the Java CAPS web site at `http://goldstar.stc.com/support`.

Creating a Deployment Profile

Before creating a Deployment Profile, you must have created an appropriate runtime Environment to support the Project.

Note – If the Project includes web services, then each of a Project's active Deployment Profiles must target a separate SOAP/HTTP External System; otherwise, the web containers can cause duplicate servlet names to appear in the domain.

Note – Deployment Profile names should contain only alphanumeric characters (letters and numbers), dashes, and underscores.

▼ To create a Deployment Profile

1. **In the NetBeans Project window, right-click on the Project to display its context menu.**

2. **From the menu, select `New` > `Deployment Profile` to display the *Create Deployment Profile* dialog.**

3. **Enter the Deployment Profile Name you want to use (or accept the default).**

   Note – The resulting application (`.ear`) file will default to a concatenation of the Deployment Profile name and the Project name (see "Naming the Application File" on page 6).

   ---

4. **Select the Environment to which you want to deploy the Project.**

5. **Select the Connectivity Map(s) to include in the deployment.**

6. **The Deployment Profile Editor now appears, showing all deployable Project components.**

7. **Drag the Project components from the left panel and drop them into the appropriate Environment components in the right panel.**
   - Drag the topics and queues to the appropriate message server, being careful not to split inbound and outbound message destinations between separate servers.
   - Drag the Collaborations to the appropriate application servers.
   - Drag the Adapter components to the appropriate external systems.

   Note – You can select multiple components of the same type that are destined for a single external system, dragging them to that external system in one operation.

   ---

8. **When the Environment components are fully populated, the left panel will be blank. You should now `Save` the profile.**

   Note – You can deploy two services that are not directly dependent upon each other across two domains; however, two processes that need to have direct access to one another must be deployed to the same domain.

Naming the Application File

You have the option of specifying the filename for the `.ear` file that is created from the Deployment Profile, which can make it easier to identify the file at a later time. By default, the

`.ear` filename is a concatenation of the Deployment Profile name and the Project name. However, if the resulting name is too long or does not fit into your file-naming scheme, you can rename the file.

You can also name an `.ear` file from the command line, without having the NetBeans IDE running. See also "Building an Application File From the Command Line" on page 12.

Note – If you want to change the name of an `.ear` file that already has been deployed, you must undeploy it before changing its name.

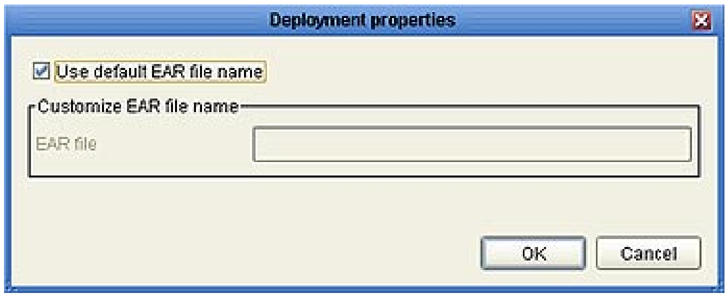
Note – If you use the same custom `.ear` file name in multiple Deployment Profiles within the same Project and Environment, previously-built `.ear` files will be overwritten by more recently-built files, since custom `.ear` file names are stored in a common directory. In this situation, you must ensure that the file names you assign are unique.

Note – Filenames can contain only alphanumeric characters, hyphens, and underscores. Spaces are not allowed.

▼ To specify the filename of the EAR file in the NetBeans IDE

1. **In the NetBeans Project window, right-click the Deployment Profile to display its context menu.**

2. **Select `Properties` to display the *Deployment Properties* dialog.**



3. **Clear the check box for `Use default EAR file name` (it is checked by default).**

4. **Enter the name you want into the `EAR file` text box, and click `OK`.**

▼ To specify the filename of the EAR file from the command line (Windows)

1. **Locate the following file, and select** `Edit` **from its context menu:**

   ```
   <CAPS_install_dir>\commandlinecodegen\ant.bat
   ```

2. **Locate the line beginning with SET ANT_OPTS, and append the following entry:**

   -Ddeployment.ear.file.name=filename

   where *filename* is the name you want for the `.ear` file; for example,

   ```
   SET ANT_OPTS=-Dantlrmaxaltblklines=3000 -Dcompile=injar -Xms512M -Xmx768m
   -XX:PermSize=256m -XX:MaxPermSize=256m -Ddeployment.ear.file.name=test_ear
   ```

3. `Save` **the file.**

▼ To specify the filename of the EAR file from the command line (AIX/Linux/MacOS/Solaris)

1. **Locate the following file, and open it in a text editor:**

   ```
   <CAPS_install_dir>/commandlinecodegen/ant
   ```

2. **Locate the following if-then-else statement:**

   ```
   if [ "$UNAME_SYSTEM" = "AIX" ] || [ "$UNAME_SYSTEM" = "aix" ];
   then ANT_OPTS='-Dantlrmaxaltblklines=3000 -Xms512M -Xmx768m'
   else ANT_OPTS='-Dantlrmaxaltblklines=3000 -Xms512M -Xmx768m
   -XX:PermSize=256m -XX:MaxPermSize=256m'
   ```

3. **Insert the following entry:**

   `-Ddeployment.ear.file.name=`*filename*

   where *filename* is the name you want for the `.ear` file, as shown in the following example:

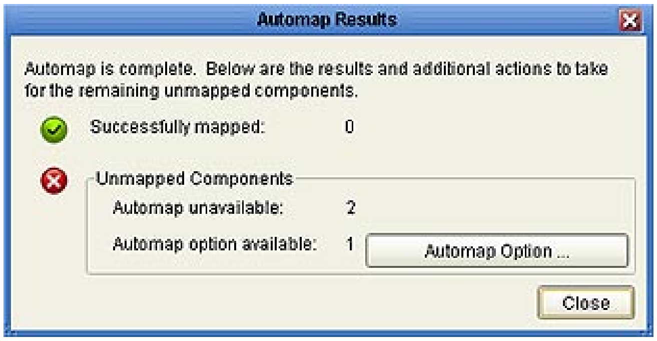
   ```
   if [ "$UNAME_SYSTEM" = "AIX" ] || [ "$UNAME_SYSTEM" = "aix" ];
   then ANT_OPTS='-Dantlrmaxaltblklines=3000 -Xms512M -Xmx768m
   -Ddeployment.ear.file.name=test_ear'
   else ANT_OPTS='-Dantlrmaxaltblklines=3000 -Xms512M -Xmx768m -XX:PermSize=256m
   -XX:MaxPermSize=256m -Ddeployment.ear.file.name=test_ear'
   ```

4. `Save` **the file.**

Automapping

When a one-to-one correspondence exists between the available objects and the containers in the Environment to which you want to deploy them, clicking the `Automap` icon automatically deploys the components to their matching containers. (This feature only works with external systems for which it is enabled.)

After the Automap feature executes, a dialog box is displayed showing the results. The lack of an appropriate container, or ambiguity between potential containers, will cause a component to remain unmapped.


Automap Results

Automap is complete. Below are the results and additional actions to take for the remaining unmapped components.

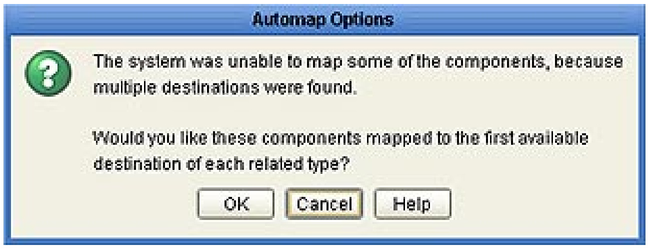
Successfully mapped: 0

Unmapped Components

Automap unavailable: 2

Automap option available: 1

In the case of an ambiguity, for example, mapping options are presented in an Automap Options dialog box that is displayed when you click `Automap Option`. You may click `OK` to accept the Automapping option, or click `Cancel` and map the deployable components manually.


Automap Options

The system was unable to map some of the components, because multiple destinations were found.

Would you like these components mapped to the first available destination of each related type?

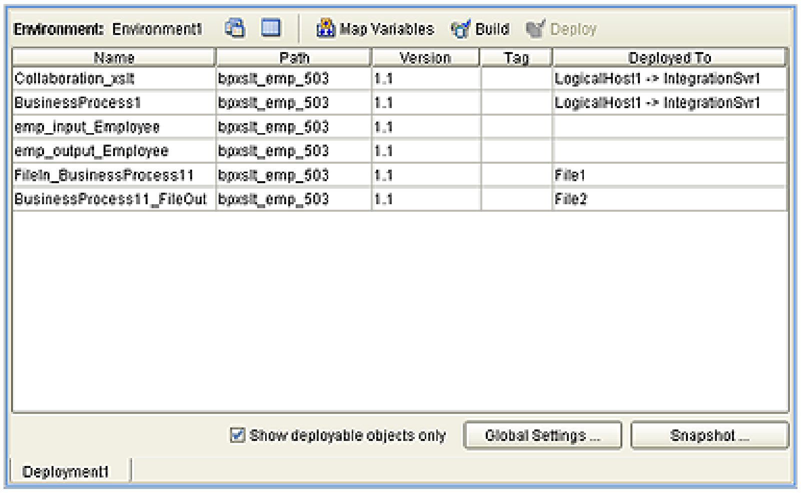
Mapping Variables

Project variables function as placeholders, having values that are determined when you create a specific Deployment Profile. These values can be literals or Environmental constants. Clicking the `Map Variables` button displays the Deployment Profile Mappings panel, where you can assign names and values.

Version Control

The Deployment Profile Editor allows you to control which version of the various Project components get mapped in the Deployment Profile. The editor provides a spreadsheet-like view to display the Project components and their versions. This view includes the component name, project path, current version in the users' workspace, tag assigned to the version (if there is one), and to what external system the component is mapped. This view is enabled by clicking the Spreadsheet View icon in the Deployment Profile Editor toolbar.

The spreadsheet view has an option that is checked by default to show only those components that are displayed when mapping components to external systems. If not checked, those components not ordinarily seen in the Deployment Profile (for example, Connectivity Maps and OTDs) will be listed as well. Listing them allows you to also specify the version of those components.

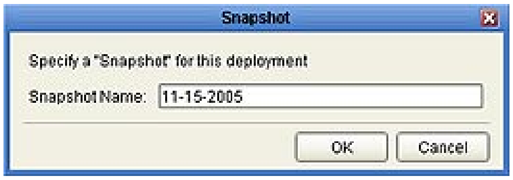


Name	Path	Version	Tag	Deployed To
Collaboration_xslt	bpxslt_emp_503	1.1		LogicalHost1 -> IntegrationSvr1
BusinessProcess1	bpxslt_emp_503	1.1		LogicalHost1 -> IntegrationSvr1
emp_input_Employee	bpxslt_emp_503	1.1		
emp_output_Employee	bpxslt_emp_503	1.1		
FileIn_BusinessProcess11	bpxslt_emp_503	1.1		File1
BusinessProcess11_FileOut	bpxslt_emp_503	1.1		File2

The spreadsheet view allows you to specify the version for each component by selecting a version or tag from a drop-down list. Selecting versions or tags does not affect your workspace, but is only for setting up the versions to appear in a snapshot; only when the snapshot is retrieved will your workspace be modified.

Creating Snapshots

The Deployment Profile Editor allows you to take a snapshot of the configuration currently in your workspace, as shown in the spreadsheet view. Clicking the `Snapshot` button presents a dialog in which you enter a name for the snapshot. Clicking `OK` then saves the snapshot.

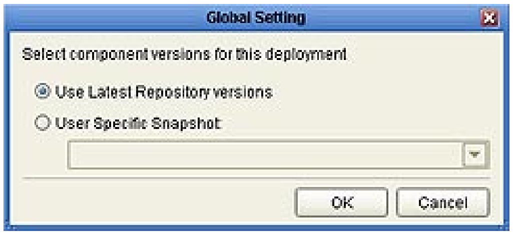


Specify a "Snapshot" for this deployment

Snapshot Name: 11-15-2005

Retrieving Snapshots

The Deployment Profile Editor gives you the option of deploying the latest component versions from the Repository or those versions recorded in a snapshot of your workspace at a previous time. Clicking the `Global Settings` button presents a dialog in which you can specify your choice and, if appropriate, name the desired snapshot. Clicking `OK` then carries out the operation.



Select component versions for this deployment

● Use Latest Repository versions

○ User Specific Snapshot

In either case, if your chosen configuration is different from what is currently in your workspace, you will receive another dialog informing you that the action will modify what is in your workspace and offering you the opportunity to not perform the operation.

- If you select a specific snapshot which is different from what is currently in your workspace, the versions of those components used by the Deployment Profile that are part of the specified snapshot will be retrieved into your workspace as read-only versions.
- If you select to use the latest Repository versions while another configuration is currently in your workspace, those components used by the Deployment Profile that are currently in your workspace will be removed and replaced by the latest versions from the Repository.

Building an Application File

The end product of the Project design process is an *application file* that can be deployed to an application server. This enterprise archive (`.ear`) file contains a collection of `.jar` files, classes, and resources. In the case of a web application, the corresponding web archive (`.war`) file contains a group of `.jar` files, classes, and resources that can be packaged and accessed as a single servlet context.

Once a Deployment Profile has been defined for your Project, you can build the `.ear` or `.war` file by clicking the `Build` button in the Deployment Editor toolbar. You can also generate an `.ear` file from the command line, without having NetBeans running.

The resulting `.ear` or `.war` file is stored in the following directory:

CAPS_install_dir`\.netbeans\caps\builds\`*DeploymentNameProjectName*`\`*logicalhost*`\`*appserver*

Note – By default, the `.ear` file does not contain the associated Java source files. For the Java source files to appear in the `.ear` file, you need to set the value of the NetBeans `run.mode` property to `debug` before building the file. You do this by setting a command-line switch in the following file:

CAPS_install_dir`\netbeans\etc\netbeans.conf`

Add "`-J-Drun.mode=debug`" to the series of entries following `netbeans_default_options=`. Note that the resulting `.ear` file can be quite large.

Building an Application File From the Command Line

You can generate an `.ear` file from the command line, without having the NetBeans IDE running, by using the *commandline codegen* tool, which you must download from the Repository using the Java CAPS Installer. When using this tool, you have three options for specifying the build properties for the `.ear` file:

1. You can specify the properties in the `build.properties` file.
2. You can specify the properties explicitly when issuing the command.
3. You can specify properties globally in the `build.properties` file, and override them selectively by specifying those to be overridden when issuing the command.

Note – All relevant components must be saved in the NetBeans IDE prior to running commandline codegen.

By default, the resulting `.ear` file will be located in the following directory:

commandlinecodegen-install`\localrepository\DEST\builds\`*ear-file*\logicalhost\appserver

---

Note – You can configure the repository location using the command-line parameter `commandline.rep.dir`. The default value is *localrepository*.

TABLE 1–1 Commandline Codegen Properties

Property Name	Description
commandline.rep.url	Repository URL (required). The format is: *http://host:port/repositoryname*
commandline.rep.user	Repository user name (required).
commandline.rep.pass	Repository user password (required).
commandline.rep.dir	Repository root directory (required). The default value is *localrepository*.
commandline.rep.projectName	Project name (required).
commandline.rep.projectDeployName	Deployment Profile name (required).
commandline.rep.projectDeploymentTag	Project snapshot name (optional).
commandline.rep.projectBranchName	Repository Branch for the Project (optional).

Note – Ensure there are no leading or trailing spaces in the values you enter for variables.

▼ To set up your system to use commandline codegen

1 **Set the `JAVA_HOME` environmental variable on your computer to the location where JDK version 1.5 is installed, for example:**

```
set JAVA_HOME=c:\Program Files\Java\jdk1.5.0_14
```

2 **Install Apache Ant, version 1.6 or later, on your computer. You can download this program from the following URL:**

`http://ant.apache.org/bindownload.cgi`

3 **Log in to the Java CAPS Uploader, click the Downloads tab, and install commandline codegen. For additional information, see "Installing Java CAPS Components Using the Java CAPS Uploader" in** *Using the Java CAPS 6 Installation GUI*

4 **At the command-line prompt, set the `ANT_HOME` environmental variable, for example:**

```
set ANT_HOME=c:\Program Files\Apache\apache-ant-1.6.5
```

Note – You must set this variable in the command window, in the directory in which you will be running the command.

Windows Platform Procedures

▼ To build an application file based on a property file only (Windows)

1. **Locate the file `build.properties` in your `commandlinecodegen` directory.**

2. **Open the file using a text editor such as *Notepad*. The contents are as follows:**

   ```
   commandline.rep.url=http://host:port/repositoryname
   commandline.rep.user=
   commandline.rep.pass=
   commandline.rep.dir=localrepository
   commandline.rep.projectName=
   commandline.rep.projectDeployName=
   commandline.rep.projectDeploymentTag=
   commandline.rep.projectBranchName=
   ```

3. **Fill in the property values as described in Table 1–1 (optional properties may be left blank).**

4. **Run commandline codegen by issuing the following command from the command-line prompt in your `commandlinecodegen` directory:**

   ```
   ant -propertyfile build.properties
   ```

▼ To build an application file by passing parameters in the command line only (Windows)

● **Run commandline codegen by issuing the following command from the command prompt in your `commandlinecodegen` directory (see Table 1–1):**

```
ant "-D(propertyname1)=(propertyvalue1)" "-D(propertyname2)=
(propertyvalue2)" ... "-D{propertynameN}={propertyvalueN}"
```

Note – When using the command-line properties method, you must specify values for all required properties. You also may specify values for any optional properties as is appropriate.

▼ To build an application file based on both a property file and by passing parameters in the command line (Windows)

1. **Locate the file `build.properties` in your `commandlinecodegen` directory.**

2. **Open the file using a text editor such as *Notepad*. The contents are as follows:**

   ```
   commandline.rep.url=http://host:port/repositoryname
   commandline.rep.user=
   commandline.rep.pass=
   commandline.rep.dir=localrepository
   commandline.rep.projectName=
   commandline.rep.projectDeployName=
   commandline.rep.projectDeploymentTag=
   commandline.rep.projectBranchName=
   ```

3. **Fill in the property values as described in Table 1–1.**

4. **Run commandline codegen by issuing the following command from the command-line prompt in your `commandlinecodegen` directory:**

   ```
   ant -propertyfile build.properties "-D{propertyname1}={propertyvalue1}"
   "-D{propertyname2}={propertyvalue2}" ... "-D{propertynameN}={propertyvalueN}"
   ```

Note – When using a combination of a property file and command-line properties, the command-line properties override those specified in the property file. You need to list only those properties that you want to override.

UNIX Platform Procedures

Note – The `Export` command works for **ksh** and **bash**. For **csh**, use `setenv` instead.

▼ To build an application file based on a property file only (AIX/Linux/MacOS/Solaris)

1. **At the command prompt, run the commands:**

   ```
   Export ANT_HOME=path
   Export JAVA_HOME=path
   dos2unix ant ant
   dos2unix build.xml build.xml
   dos2unix build.properties build.properties
   ```

2. **Locate the file `build.properties` in your `commandlinecodegen` directory.**

3 **Open the file using a text editor. The contents are as follows:**

```
commandline.rep.url=http://host:port/repositoryname
commandline.rep.user=
commandline.rep.pass=
commandline.rep.dir=localrepository
commandline.rep.projectName=
commandline.rep.projectDeployName=
commandline.rep.projectDeploymentTag=
commandline.rep.projectBranchName=
```

4 **Fill in the property values as described in Table 1–1 (optional properties may be left blank).**

5 **Run commandline codegen by issuing the following command from the command-line prompt in your `commandlinecodegen` directory:**

```
sh ant –propertyfile build.properties
```

▼ To build an application file by passing parameters in the command line only (AIX/Linux/MacOS/Solaris)

1 **At the command prompt, run the commands:**

At the command prompt, run the commands:

```
Export ANT_HOME=path
Export JAVA_HOME=path
dos2unix ant ant
dos2unix build.xml build.xml
dos2unix build.properties build.properties
```

2 **Run commandline codegen by issuing the following command from the command prompt in your `commandlinecodegen` directory (see Table 1–1):**

```
sh ant "-D{propertyname1}={propertyvalue1}" "-D{propertyname2}=
{propertyvalue2}" ... "-D{propertynameN}={propertyvalueN}"
```

Note – When using the command-line properties method, you must specify values for all required properties. You also may specify values for any optional properties as is appropriate.

▼ To build an application file based on both a property file and by passing parameters in the command line (AIX/Linux/MacOS/Solaris)

1 **At the command prompt, run the commands:**

```
Export ANT_HOME=path
Export JAVA_HOME=path
dos2unix ant ant
```

```
dos2unix build.xml build.xml
dos2unix build.properties build.properties
```

2 **Locate the file `build.properties` in your `commandlinecodegen` directory.**

3 **Open the file using a text editor such as *Notepad*. The contents are as follows:**

```
commandline.rep.url=http://host:port/repositoryname
commandline.rep.user=
commandline.rep.pass=
commandline.rep.dir=localrepository
commandline.rep.projectName=
commandline.rep.projectDeployName=
commandline.rep.projectDeploymentTag=
commandline.rep.projectBranchName=
```

4 **Fill in the property values as described in Table 1–1.**

5 **Run commandline codegen by issuing the following command from the command-line prompt in your `commandlinecodegen` directory:**

```
sh ant -propertyfile build.properties "-D{propertyname1}={propertyvalue1}"
 "-D{propertyname2}={propertyvalue2}" ... "-D{propertynameN}={propertyvalueN}"
```

When using a combination of a property file and command-line properties, the command-line properties override those specified in the property file. You need to list only those properties that you want to override.

Deploying Application Files from the NetBeans IDE

You can deploy a Java CAPS Repository-based project to Sun Java System Application Server by using the NetBeans IDE.

These instructions assume that you have built an application file in the NetBeans IDE. See "Building an Application File" on page 12.

Note – If you deploy by using the NetBeans IDE, then you cannot specify the server instance. Therefore, if the domain has multiple server instances, the application is deployed to all of the instances.

▼ To Deploy an Application File from the NetBeans IDE

1 **Ensure that the application server is running.**

2 **In the toolbar of the Deployment Profile Editor, click `Deploy`.**

Deploying Application Files from Enterprise Manager

You can deploy a Java CAPS Repository-based project to Sun Java System Application Server by using Enterprise Manager.

You can access Enterprise Manager with any of the following browsers:

- Internet Explorer 6 Service Pack 2
- Internet Explorer 7
- Mozilla Firefox 2.0 and above

These instructions assume that you have built an application file. See "Building an Application File" on page 12.

Note – If you deploy by using Enterprise Manager, then you cannot specify the server instance. Therefore, if the domain has multiple server instances, the application is deployed to all of the instances.

Adding the Application Server Domain to Enterprise Manager

In order to manage a Sun Java System Application Server domain in Enterprise Manager, you must first add the domain.

▼ To Add the Application Server Domain to Enterprise Manager

1 **Ensure that the application server is running.**

2 **In the Explorer panel of Enterprise Manager, click the Java EE node.**
The Manage Servers tab appears.

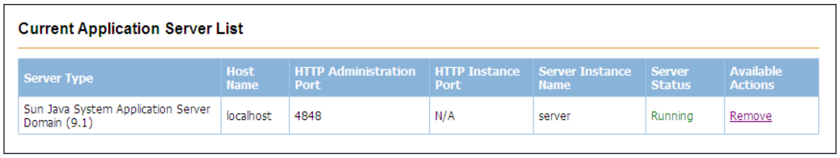
3 **Enter the connection information.**

Field	Description
Server Type	The type of application server. Set this field to Sun Java System Application Server Domain (9.1).
Host Name	The fully qualified host name (for example, `myhost.company.com`) or IP address of the computer on which the application server is running.

Field	Description
HTTP Administration Port	The port number of the Domain Administration Server.
User Name	The user name required to access the domain.
Password	The password required to access the domain.

4 **Click Connect to Server.**

The application server domain is added to the Current Application Server List table.



Current Application Server List

Server Type	Host Name	HTTP Administration Port	HTTP Instance Port	Server Instance Name	Server Status	Available Actions
Sun Java System Application Server Domain (9.1)	localhost	4848	N/A	server	Running	Remove

Deploying the Application File

Once you have added the Sun Java System Application Server domain to Enterprise Manager, you can deploy the application file.

▼ To Deploy the Application File

1 **Ensure that the application server is running.**

2 **In the Details panel of Enterprise Manager, click the Deploy Applications tab.**

3 **Click Browse and select the application file.**

4 **Select the Deploy and Enable check boxes next to any appropriate server.**

There might be more than one server running.

5 **Click Deploy.**

The Results area indicates the status of the deployment.

Deploying Application Files from the Admin Console

You can deploy a Java CAPS Repository-based project to Sun Java System Application Server by using the Sun Java System Application Server Admin Console.

These instructions assume that you have built an application file. See "Building an Application File" on page 12.

For detailed information about the Admin Console, see the *Sun Java System Application Server 9.1 Administration Guide*.

▼ To Deploy an Application File from the Admin Console

1. **Ensure that the application server is running.**

2. **Log in to the Sun Java System Application Server Admin Console.**

3. **In the left pane, expand the Applications node and then click Enterprise Applications.**

   The Enterprise Applications page appears.

4. **Click Deploy.**

   The Deploy Enterprise Applications/Modules page appears.

5. **Select the Packaged File to be Uploaded to the Server option.**

6. **Click Browse and select the application file.**

7. **Click OK.**

   The Enterprise Applications page appears. The application is now in the list of deployed applications.

Deploying Application Files from the Command Line

You can deploy a Java CAPS Repository-based project to Sun Java System Application Server from the following command-line utilities:

- `asadmin`
- `asant`

These instructions assume that you have built an application file. See "Building an Application File" on page 12.

Deploying from the `asadmin` Utility

The application server includes a command-line utility called `asadmin` for performing administrative tasks.

To deploy an application file from the `asadmin` utility, use the `deploy` command. For example:

```
asadmin deploy C:\JavaCAPS6\dpFTQFprjFTQF.ear
Command deploy executed successfully
```

For detailed information about the `asadmin` utility, see the *Sun Java System Application Server 9.1 Administration Guide*.

Deploying from the `asant` Utility

The application server includes a command-line utility called `asant` for invoking Apache Ant.

To deploy an application file from the `asant` utility, use the `sun-appserver-deploy` task. The following example is a simple application deployment script. The `file` and `passwordfile` attributes are shown. Other attributes are implied.

```
<sun-appserver-deploy
file="${assemble}/dpFTQFprjFTQF.ear"
passwordfile="${passwordfile}" />
```

For detailed information about the `asant` utility, see the *Sun Java System Application Server 9.1 Developer's Guide*.

Index