Peregrine

Get-Services Installation Guide

For Windows, Solaris, AIX, and Linux

PART NO: GET-SER-4.0-ENG-01028-00250

Peregrine SYSTEMS

If you need technical support for this product, or would like to request documentation for a product for which you are licensed, contact Peregrine Systems, Inc. Customer Support by e-mail at support@peregrine.com.

If you have comments or suggestions about this documentation, contact Peregrine Systems, Inc. Technical Publications by e-mail at doc_comments@peregrine.com.

This edition applies to version 4.0 of the licensed program.

Peregrine Systems, Inc.
Worldwide Corporate Headquarters
3611 Valley Centre Drive San Diego, CA 92130
Tel 800.638.5231 or 858.481.5000
Fax 858.481.1751
www.peregrine.com

Contents

About this Guide

Get-Services is an application that provides a web-based interface to Peregrine ServiceCenter®. Get-Services enables users to report problems in their work environment by opening incident tickets in the appropriate back-end system.

This book provides step-by-step instructions for installing Get-Services. This guide enables you to:

- Install the Peregrine OAA Platform and Get-Services.
- Configure Get-Services for ServiceCenter.

Book audience

This guide is for Get-Services administrators who configure and maintain the application. To use this guide effectively, you need to have knowledge of the following:

- XML and ECMAScript (or JScript/JavaScript)
- Operating guides, reference manuals, and other documentation for your PC hardware and operating system
- ServiceCenter administration and functionality

Related documentation

Refer to the following documentation for additional information:

- *Get-Services Administration Guide* describes the Peregrine OAA platform and Get-Services administration.
- *Get-Services Release Notes* covers any late breaking documentation or known issues with Get-Services. These are constantly updated and posted to the Customer Support web site. See *Contacting customer support* on page 10 for details on accessing the Customer Support website.

Associated applications

This guide does not contain information about products that may be used with Get-Services, such as Peregrine OAA, ServiceCenter, or Password Management. Refer to the appropriate product documentation for information about installing, configuring, and using these associated applications.

Note: ServiceCenter must be installed and configured before you can install and configure Get-Services. Peregrine OAA installs with Get-Services, and only the installation of Peregrine OAA for Get-Services is included in this guide.

Terminology

The terminology used in this guide and in the Get-Services interface is based on ServiceCenter 4.x and 5.x.

Typographical conventions

This guide uses typeface conventions to indicate special terms and actions. These conventions and their meanings are:

Convention	Meaning
Bold	Information that you must type exactly as shown appears in **bold**. The names of buttons, menus, and menu options also appear in **bold**.
Italics	Variables and values that you must provide appear in *italics*. New terms also appear in *italics*.

Monospace	Code or script examples, output, and system messages appear in a monospace font.
	`var msgTicket = new Message( "Problem" );`
	`…`
	`msgTicket.set( "_event", "epmc" );`
	
	An ellipsis (...) is used to indicate that portions of a script have been omitted because they are not needed for the current topic. Samples of code are not entire files, but they are representative of the information discussed in a particular section.

Sans Serif	Filenames, such as login.asp, appear in a sans serif font.

Special elements

This book uses special elements to help you locate information. These special elements and their uses are in the following table:

Element	Usage
Important:	Information that is required to complete a task
Note:	Information that is of general interest
Tip:	Information that can make a task easier or faster
Warning:	Information that is needed when there is a risk of losing data

Organization of the guide

The following table shows you where in this guide to find the information you need.

This section	Provides information about
Chapter 1, Get-Services Installation Overview	Installation requirements, types of installations, and back-end databases.
Chapter 2, Installing on Windows	Installing and configuring application servers and Web servers on a Windows operating system.
Chapter 3, Installing on AIX, Linux, or Solaris	Installing and configuring application servers and Web servers on a Unix operating system.
Chapter 4, Load Balancing	Creating and configuring multiple instances of servers.
Chapter 5, ServiceCenter Administration	Uploading files and configuring ServiceCenter to work with Get-Services.
Chapter 6, Configuring the Adapters	Configuring the Get-Services Admin module for ServiceCenter.
Chapter 7, Troubleshooting	Troubleshooting installation problems with Apache Web server, Tomcat, OAA, and ServiceCenter.

Contacting customer support

For further information and assistance with this release, contact Peregrine Systems' Customer Support.

Peregrine CenterPoint Web site

Current details of local support offices are available through the following main contacts or through the Peregrine CenterPoint Web site at:

http://support.peregrine.com

You need your current login user name and password to access this Web page.

To contact Peregrine customer support

1. Log in to the Web site with your login user name and password.
2. Click **Go** beside the CenterPoint support area.
3. From **Contents** on the left, select **Whom Do I Call?** to display the **Peregrine Worldwide Contact Information**.

Documentation Web site

A complete listing of the available documentation is on Peregrine's CenterPoint Web site at:

http://support.peregrine.com

Important: Release Notes for this product are continually updated after the release of the product. Visit the Peregrine Customer Support Web site to ensure that you have the most current version of the Release Notes.

CHAPTER 1 Get-Services Installation Overview

This chapter covers the following topics for Get-Services:

Installation requirements

This section outlines the recommended minimum configuration for proper installation and configuration of Get-Services. Before beginning installation, ensure that you have the following.

Requirement	for Windows	for Unix
Java run-time environment	Java 2 SDK Standard Edition v1.3.1_05. *Available on the Get-Services Installation CD.*	Java 2 SDK Standard Edition v1.3.1_05. *Available on the Get-Services Installation CD.*
Application server	Any one of the following: ■ Tomcat 4.1.12 *Available on the Get-Services Installation CD.* ■ WebSphere 4.0.2 or later ■ WebLogic 6.1 SP3 or later ■ JRun 3.1	Any one of the following: ■ Tomcat 4.1.12 *Available on the Get-Services Installation CD.* ■ WebSphere 4.0.2 or later ■ WebLogic 6.1 SP3 or later ■ JRun 3.1
Back-end database(s)	ServiceCenter 4.0.x or later	ServiceCenter 4.0.x or later
Operating systems	■ Windows 2000 Server SP2	■ AIX 5.1 ■ Red Hat Linux 7.3 ■ Solaris 2.7 or Solaris 2.8
Web Server	One of the following: ■ Apache 2.0.43 *Available on the Get-Services installation CD.* ■ Microsoft IIS Server 5.0 ■ IBM HTTP Server 1.3.19 *Available on the WebSphere installation CD and from the IBM support Website*	One of the following: ■ Apache 2.0.43 *Available on the Get-Services installation CD.* ■ IBM HTTP Server 1.3.19 *Available on the WebSphere installation CD and from the IBM support Website*
System Processor	Pentium, 400 MHz or faster	Pentium, 400 MHz or faster

Requirement	for Windows	for Unix
RAM	512 MB or more.	512 MB or more
Hard disk space	100 MB for Get-Services.	100 MB for Get-Services.

Types of installations

The Get-Services installer offers two basic types of installation:

- Typical installation
- Custom installation

A *typical installation* installs a fixed configuration of Get-Services all on one server. Typical installations are intended to set up development environments (see below).

A *custom installation* allows you to choose the exact components installed on a given server. Custom installations are intended for users who will be using alternate application servers, web servers, or to set up a production environment.

The Get-Services custom installation can be optimized for two types of environments:

- Development environment
- Production environment

A *development environment* installation places all needed software and data on one server. It is intended for Get-Services implementers to review application functionality and test customizations before deploying to a production environment. By default, the Get-Services installer uses the development environment installation.

A *production environment* installation is optimized for performance and scalability. Each software function, such as an application server and Web server, are installed on different servers. In addition, there may be multiple instances of any one part of the basic Get-Services environment. Given the amount of flexiblity involved in such an installation, users must manually set up a production environment.

Back-end systems

In order to use Get-Services you must have a properly configured back-end systemGet-Services utilizes the back-end system to:Get-Services uses ServiceCenter as its back-end database. You can install Get-Services 4.0 with ServiceCenter 4.x or ServiceCenter 5.x.

- Authenticate users and define access rights
- Process application workflows and store data
- Store personalization settings for the web application

Get-Services uses ServiceCenter or AssetCenter as a back-end system. Refer to *Installation requirements* on page 14 for a complete list of the AssetCenter and ServiceCenter versions compatible with Get-Services.

2 Installing on Windows

CHAPTER

This chapter covers the following topics:

Choosing an installation environment

You can install Get-Services in one of two installation environments:

- Development environment
- Production environment

The Get-Services development environment is intended for you to evaluate product features and customize your installation prior to deployment in a production environment. In a development environment, you install all software required for Get-Services on one computer system.

You have two choices of development environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Services deployed on Tomcat 4.1.12 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Services

The Get-Services production environment is intended to maximize server performance and scalability, and to deploy any customizations you have made. In a production environment, you install the various components of Get-Services on different servers to maximize performance.

You have two choices of production environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Services deployed on multiple instances of Tomcat 4.1.12 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Services

Development Environment

The following procedures describe how to install Get-Services in a development environment.

To install Get-Services in a typical development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Run the Get-Services installer and select the Typical installation option. See *Typical installation option* on page 63.

Step 4 Configure the back-end databases and create Get-Services users.

To install Get-Services in a custom development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Install alternate application and Web servers.

Step 4 Configure the alternate application server for Get-Services. See *Configuring alternate application servers* on page 25.

Step 5 Run the Get-Services installer and select the Custom installation option. See *Custom installation option* on page 70.

Step 6 Configure the back-end databases and create Get-Services users.

Production Environment

The following procedures describe how to install Get-Services in a production environment.

To install Get-Services in a typical production environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services on a separate server.

Step 3 Run the Get-Services installer and select the Typical installation option. See *Typical installation option* on page 63.

Step 4 Configure multiple instances of Tomcat for load balancing on the Apache Web server.

Step 5 Configure the back-end databases and create Get-Services users.

To install Get-Services in a custom development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Install the alternate application server and Web server on separate servers.

Step 4 Configure the alternate application server for Get-Services. See *Configuring alternate application servers* on page 25.

Step 5 Run the Get-Services installer and select the Custom installation option. See *Custom installation option* on page 70.

Step 6 Configure the Web servers and application servers for load balancing.

Step 7 Configure the back-end databases and create Get-Services users.

Migrating Get-Services from previous versions

To migrate older versions of Get-It or Get-Services to Get-Services 4.0 requires both a manual data migration process and the recreation of any interface customizations you have made. The following steps describe the migration process.

Important: You should backup all Get-Services and back-end system data prior to performing any steps for migration.

To migrate previous versions to Get-Services 4.0:

Step 1 Review the customizations of previous version and determine which customizations need to be recreated in Get-Services 4.0. See *Recreating customizations in Get-Services 4.0* on page 21.

Step 2 Install Get-Services 4.0 on a new system. See *Choosing an installation environment* on page 18.

Step 3 Apply any required configuration changes to the back-end database you want to migrate to Get-Services 4.0. See *Configuring an existing back-end database for Get-Services 4.0* on page 23.

Recreating customizations in Get-Services 4.0

You cannot directly migrate customizations implemented in previous versions to Get-Services 4.0. Instead, you must recreate your changes using the new features and methods available in Get-Services 4.0.

The following sections describe how to recreate your customizations from previous versions.

No customizations

If you have made no customizations to Get-Services, you can simply install Get-Services 4.0 on a new system and migrate your data from your existing back-end database.

Customized JSP files

In previous versions, customers had to directly modify JSP files in order to add or remove certain functionality. The following table describes how to recreate some of the more common JSP file modifications.

JSP file modification	New method to use
Remove the user self-registration option from login page	Enable or disable the user registration option from the Administration Settings page
Remove the change password option from the login page	Enable or disable the change password option from the Administration Settings page

Personalized pages

Get-Services 4.0 offers many more pages that you can personalize directly from the Web interface. If you personalized pages in a previous version, you must recreate your personalized pages in Get-Services 4.0 using DocExplorer.

You can use personalization to:

- Add or remove fields from a page
- Save a personalized search results or details on your portal page

Customized skins, stylesheets, and themes

Get-Services 4.0 has combined all interface images and stylesheets into themes. Users can no longer select separate skins and stylesheets. The new themes consist of skins (which themselves are composed of image files, frame definitions, and layer files), cascading stylesheet definitions, and XSL templates.

Although you may copy over older custom themes to Get-Services 4.0, you may experience rendering errors due to the new images, CSS definitions, frame definitions, and layers. It is recommended that you recreate any custom themes using the Get-Services 4.0 version of the classic theme as your template.

Alternate login pages and authentication methods

If you used a custom login page or an alternate authentication method in a previous version, you can re-use or recreate these customizations using the updated instruction. You can find information about alternate security methods in the *Get-Services Administration Guide*.

Customizations made with a previous tailoring kit

Many customizations that required a tailoring kit in previous versions can now be done directly from the Get-Services Web interface. The following table describes how to recreate some of the more common tailoring kit changes.

Tailoring kit modification	New method to use
Added or removed fields form a form	Add or remove fields from Personalization
Added a new language or locale to the Get-Services interface	Create and edit language strings files directly. You may also purchase officially supported language packs from Peregrine Systems
Made changes to the common, portal, or Peregrine Studio packages	These packages are no longer available for tailoring, however most common interface settings can now be customized from the Administration Settings page.
Made changes to schemas or ECMA server-side scripts	Review new functionality and determine if you still need the customized scripts and schemas. If you do need the customizations, you will to recreate them in the current version of the Get-Services tailoring kit.

Configuring an existing back-end database for Get-Services 4.0

The following table lists the options available for data migration.

Get-Services 2.3 to Get-Services 4.0

Back-end version	Migration required
ServiceCenter 3.0	Upgrade to ServiceCenter 4.x or 5.0.x

Back-end version	Migration required
ServiceCenter 4.x	Apply Get-Services 4.0 unload files to existing ServiceCenter 4.x
Service 5.0.x	Apply Get-Services 4.0 unload files to existing ServiceCenter 5.0.x

Configuring alternate application servers

You must install a Java-enabled application server to support your Peregrine Web applications. Peregrine OAA supports the following alternate application servers:

- *Tomcat 4.1.12 connecting to IIS 5.0*
- *WebSphere 4.0.2*
- *WebLogic 6.1 SP3 or SP4*
- *JRun 3.1*

The Get-Services typical installation option installs Tomcat 4.1.12 and connects it to an Apache 2.0 web server. You can also install Tomcat 4.1.12 using the custom installation option.

Important: If you want to use an application server other than Tomcat 4.1.12, then you must configure your application and Web servers *prior* to running the Get-Services installer.

See the following sections for instructions configuring alternate application servers for Get-Services.

Tomcat 4.1.12 connecting to IIS 5.0

You can use the Get-Services installer to install the Tomcat application server. If you use the typical installation option, the Get-Services installer configures Tomcat for the Apache Web server. In order to configure the Tomcat for the IIS Web server, you must perform a custom installation and configure IIS using the following instructions.

To configure Tomcat to connect to an IIS 5.0 Web server:

Step 1 Run the Get-Services installer and select the Custom installation option. See *Custom installation option* on page 70.

Step 2 Configure the ISAPI Plugin for IIS. See *Configuring the ISAPI Plugin for IIS* on page 27.

Step 3 Configure IIS to use isapi_redirector2.dll as an ISAPI Filter. See *Configuring the isapi_redirector2.dll as an ISAPI filter* on page 27.

Step 4 Create and configure a jakarta virtual directory in IIS. See *Configuring a jakarta virtual directory in IIS* on page 28.

Step 5 Create and configure an oaa virtual directory in IIS. See *Configuring an oaa virtual directory in IIS* on page 28.

Step 6 Edit the workers2.properties file of the first or master Tomcat instance to set the values for each additional Tomcat instance. See *Editing the workers2.properties file for IIS* on page 29.

Step 7 Edit the server.xml file. See *Editing the server.xml file for IIS* on page 30.

Step 8 Edit the jk2.properties file. See *Editing the jk2.properties file for IIS* on page 31.

Step 9 Install Tomcat as a service using installservice.bat (Optional). This file can be found in the Tomcat\bin directory. See *Installing Tomcat as a service* on page 32.

Step 10 Pre-compile JSP files for your production environment. See *Precompiling JSP files for a Tomcat production environment* on page 32.

Configuring the ISAPI Plugin for IIS

The ISAPI plugin for IIS establishes a connection between Tomcat and the IIS Web server. Before configuring IIS to use this connector, you must update the registry file entry for the connector to ensure that it has the proper paths listed for the Tomcat application server.

The Get-Services installer automatically places a copy of the ISAPI plugin for IIS in the following folder:

c:\Program Files\Peregrine\Common\Tomcat4\bin

Use the following procedures to configure the plugin for your intranet environment.

To configure the ISAPI plugin for IIS:

1. Open the file jk2.reg in a text editor. This file is located at:

   C:\Program Files\Peregrine\Common\Tomcat4\conf

2. Verify that the "serverRoot" and "workersFile" values list the proper installation path to Tomcat. By default these values are:

   ```
   "ServerRoot"="C:\\Program Files\\Peregrine\\Common\\Tomcat4"
   "workersFile"="C:\\Program Files\\Peregrine\\Common\\Tomcat4\\conf\\
   workers2.properties"
   ```

3. Save and close the jk2.reg file.
4. Double-click on the jk2.reg file from Windows Explorer.

   Windows adds the registry settings to the Windows registry.

Configuring the isapi_redirector2.dll as an ISAPI filter

To establish a connection between Tomcat and IIS, you will need to install isapi_redirector2.dll as an ISAPI filter.

To install isapi_redirect2.dll as an ISAPI filter:

1. Open the Internet Services management console.
2. Right-click the **Default Web Site** node and then click **Properties**.
3. Click the **ISAPI Filters** tab.
4. Click **Add**.
5. Enter the following information:

a **Filter Name**: jakarta. The filter name must match the name you defined the jk2.reg registry file. By default the filter name is jakarta.

b **Executable**: isapi_redirector2.dll. By default this file is located at:

C:\Program Files\Peregrine\Common\Tomcat4\bin\isapi_redirector2.dll

6 Click **OK**.

7 Close the Internet Services management console.

Configuring a jakarta virtual directory in IIS

The ISAPI plugin for IIS requires a specific IIS virtual directory in order to run. Use the following guidelines to create the IIS virtual directory. For specific instructions about configuring IIS, refer to your Windows Help.

Requirements for a jakarta virtual directory

Requirement	Setting
Create virtual directory	jakarta
Directory access rights	execute
Map to physical path	<Tomcat>\bin\isapi_redirector2.dll

For <*Tomcat*>, enter the path to your tomcat installation. By default the path is:

C:\Program Files\Peregrine\Common\Tomcat4

Configuring an oaa virtual directory in IIS

To run Get-Services from IIS, you need to create a virtual directory that maps to your Tomcat deployment folder.

Requirements for an oaa virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Directory access rights	anonymous
Map to physical path	<Tomcat>\webapps\oaa
Set Execute Permissions to	Scripts and Executables
Remove "allow anonymous access" to	default.asp, login.asp, e_login_main_start.asp
Set security access to only allow "System" and "Authenticated Users" to	default.asp, login.asp, e_login_main_start.asp

For <*oaa*>, enter the name of the virtual directory you wan to use for Get-Services. Whatever name you enter here you will need to replicate in your application server configuration.

For <*Tomcat*>, enter the path to your tomcat installation. By default the path is:

C:\Program Files\Peregrine\Common\Tomcat4

Editing the workers2.properties file for IIS

For each server on which Tomcat is installed, there is only one workers2.properties file. Tomcat installs the workers.properties file in the conf directory of your primary Tomcat instance. This file will be shared by all other Tomcat instances on that particular server.

The workers2.properties file specifies the worker threads that the Web server connector will create in order to communicate with your Tomcat instances.

To edit the worker2.properties file:

1. Open the workers2.properties file (located in the config directory of your Tomcat installation) in any text editor.
2. Create a channel.socket entry.

   Example:

   ```
   [channel.socket:<server>:<port>]
   ```

```
info=Description of Tomcat instance
debug=0
tomcatId=<server>:<port>
lb_factor=1
disabled=0
```

For <*server*>, enter the server name of where Tomcat is installed.

For <*port*>, enter the communications port on which Tomcat is listening.

3 Verify that the uri settings lists the proper IIS virtual directory. By default the virtual directory should be **oaa**.

If you have defined a different virtual directory other than **oaa** to run Get-Services, you will need to change the **uri** values listed here.

Example:

```
[uri:/oaa/servlet/*]
info=Prefix mapping

[uri:/oaa/*.jsp]
info=Extension mapping
```

4 Save the file.

Editing the server.xml file for IIS

You must edit this file to connect Tomcat to IIS as well as to find the Get-Services Web application files.

To edit the server.xml file:

1 Open the file server.xml in any text editor. Be default this file is located at:

C:\Program Files\Peregrine\Common\Tomcat4\conf

2 Update the port number attribute of the <Server> element to a unique value that will not conflict with other port numbers on your network.

Example:

```
<Server port="8005" shutdown="SHUTDOWN" debug="0">
```

3 Update the port number attribute of the Coyote Connector <Connector> element to a unique value that will not conflict with other port numbers on your network.

Example:

```
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8009" minProcessors="5" maxProcessors="75" enableLookups="true"
redirectPort="8443" acceptCount="10" debug="0"
connectionTimeout="20000" useURIValidationHack="false"
protocolHandlerClassName="org.apache.jk.server.JkCoyoteHandler" />
```

4 Create a <Context> element entry from Tomcat to the Get-Services deployment directory.

Add the entry just above the "examples" Context entry.

Example:

```
<Context path="/oaa"
docBase="<Tomcat>/webapps/oaa"
crossContext="false"
debug="0"
reloadable="false" >
</Context>
```

For the docBase attribute, set *<Tomcat>* to the absolute path of the first or master Tomcat instance.

5 Update the <Engine> element with the server name and communications port used by Tomcat.

List the server information in the jvmRoute attribute.

Example:

```
<Engine jvmRoute="localhost:8009" name="Standalone"
defaultHost="local host" debug="0">
```

6 Update the <Host> element with the webapps directory used by the first or master Tomcat instance.

List the server information in the appBase attribute.

Example:

```
<Host name="localhost" debug="0" appBase="<Tomcat>/webapps"
unpackWARs="true" autoDeploy="true">
```

For the appBase attribute, set *<Tomcat>* to the absolute path of the first or master Tomcat instance.

7 Save the file server.xml.

Editing the jk2.properties file for IIS

You will need to modify the jk2.properties file to set the jk2 communication port.

To edit the jk2.properties file:

1 Open the jk2.properties file in any text editor. Be default this file is located at:

C:\Program Files\Peregrine\Common\Tomcat4\conf

2 Insert a line for the channelSocket port. The port number must match the port number defined in workers2.properties file.

Example:

```
channelSocket.port=8009
```

3 Save the file.

Installing Tomcat as a service

After you have edited the Tomcat files, you can install Tomcat as Windows services using installservice.bat.

To install Tomcat as a service:

1 Open a DOS command prompt and change directories to your Tomcat bin directory.

2 Enter the following command to create each Tomcat instance:

```
installservice <service name> <tomcat_home> <jvm_dll_path>
```

Where *<service name>* is the name you wish to give the Tomcat service, *<tomcat_home>* is the Tomcat install directory of the instance for which you are creating the service, and *<jvm_dll_path>* is the Java SDK install directory.

The second and third parameters are optional if you have already set the CATALINA_HOME and JAVA_HOME environment variables.

Example:

```
installservice Tomcat8009 C:\Program Files\Peregrine\Common\Tomcat4
C:\Program Files\Peregrine\Common\jdk1.3.1_05
```

3 Repeat step 1 through step 2 for each Tomcat service you wish to create.

Precompiling JSP files for a Tomcat production environment

If you plan to use Tomcat 4.1.12 in a production environment or on a system with multiple processors, you will need to pre-compile the JSP files deployed for Get-Services. This extra configuration step is the result of a known issue with Tomcat. See the following URL for more details about this issue:

http://nagoya.apache.org/bugzilla/show_bug.cgi?id=14077

To pre-compile JSP files for Tomcat 4.1.12:

1. Stop the Tomcat application server.
2. Open a command prompt.
3. Enter one of the following commands based upon your operating system:

Operating system	Command required
Windows	set JASPER_HOME=<*Tomcat directory*>

For <*Tomcat directory*> enter the absolute path to your Tomcat installation.

4. Change directories to the Tomcat bin folder.

Operating system	Command required
Windows	cd %jasper_home%\bin

5. Run the precompile batch file.

Operating system	Command required
Windows	precompile <*Web app name*> <*Tomcat instance name*>

For <*Web app name*> enter the name of the Get-Services deployment folder. You may omit this name if you are using the default folder named oaa.

For <*Tomcat instance name*> enter the name of the Tomcat instance you have installed. You may omit this name if you are using the default instance named Standalone.

The batch file displays the progress of the conversion. When it is complete, the command prompt returns.

6. Start the Tomcat application server.

WebSphere 4.0.2

Use the following procedures to configure WebSphere to run Get-Services on Windows.

To configure WebSphere 4.02:

Step 1 Install WebSphere 4.02. Your version of WebSphere 4.0.2 includes the IBM HTTP Server.

Step 2 Deploy the Portal WAR file to WebSphere to create the necessary folder structure for Get-Services. See *Deploying the Portal WAR file to WebSphere* on page 34.

Step 3 Set the JVM Java heap size for each WebSphere instance running Get-Services. See *Setting the Java heap size* on page 36.

Step 4 Create the virtual directory you want to use for Get-Services in your Web server. See *Configuring a virtual directory for IBM HTTP Server* on page 38.

Step 5 Run the Get-Services installer.

If you plan on setting up a WebSphere Portal Server or a WebSphere Translation Server, see*Installing WebSphere Portal Server* on page 39 or *Configuring WebSphere Translation Server for Get-Services* on page 48.

Deploying the Portal WAR file to WebSphere

The Portal WAR file creates the folder structure necessary to deploy Get-Services in your application server. After you have deployed this file to WebSphere you will be ready to run the Get-Services installer.

To deploy the Portal WAR file to WebSphere:

1. Verify that the WebSphere Admin Server has been started.
2. Open the WebSphere Advanced Administrator's Console (**Start** > **Programs** > **IBM WebSphere** > **Application Server** > **Administrator's Console**).
3. On the menu at the left side of the console, right-click on **Enterprise Applications** and select **Install Enterprise Application**.
4. On the screen displayed, do the following:
   a. Select **Install stand-alone module**.
   b. In the **Path** field, browse to the path to the portal<version #>.war file. The default is <CD Rom Drive>:\portal<version #>.war.

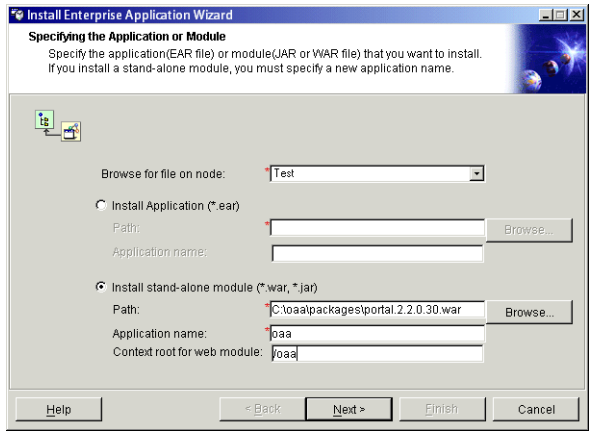
For <version #>, Select the most recent version available (4.0.0.44 or greater).

c In the **Application Name** field, type `oaa`.

d In the **Context Root** field, type the name of Get-Services virtual Web server directory you wish to use. Example: `/oaa`.

Important: You must create a Web server virtual directory matching the context root you enter here.

The following screen shows the completed form.

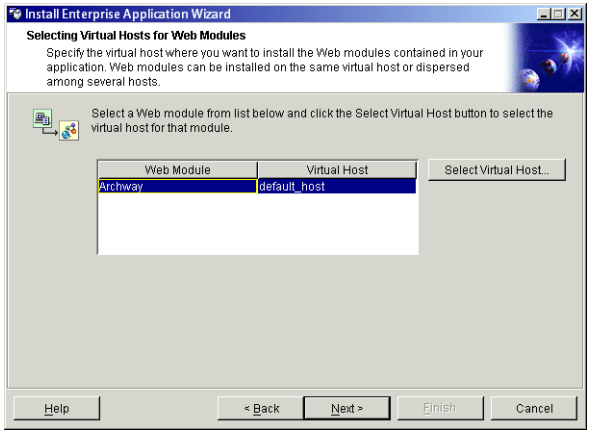


5 Click **Next**.

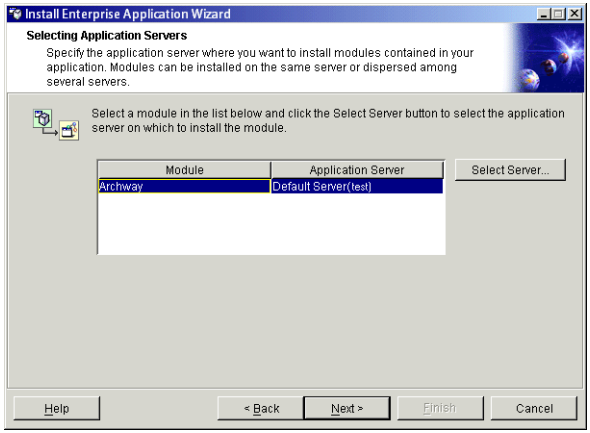
6 Click **Next** on the following dialog boxes. These screens will not be used.

- Mapping Users to Roles
- Mapping EJB Run As Roles to Users
- Binding Enterprise Beans to JNDI Names
- Mapping EJB References to Enterprise Beans
- Mapping Resource References to Resources
- Specifying the Default Datasource
- Specifying Data Sources for Individual CMP Beans

7 In the Selecting Virtual Hosts for Web Modules, select the WebSphere server instance you want to use, and then click **Next**.



8 In the Selecting Application Servers dialog box, select the WebSphere server instance you want to use, and then click **Next**.



9 On the dialog box displayed, click **Finish**.

Setting the Java heap size

You can configure how much memory is available for your application server instances. The following instructions assume you are only using one WebSphere instance. You will need to adjust the heap size accordingly if you are load balancing across several WebSphere instances.

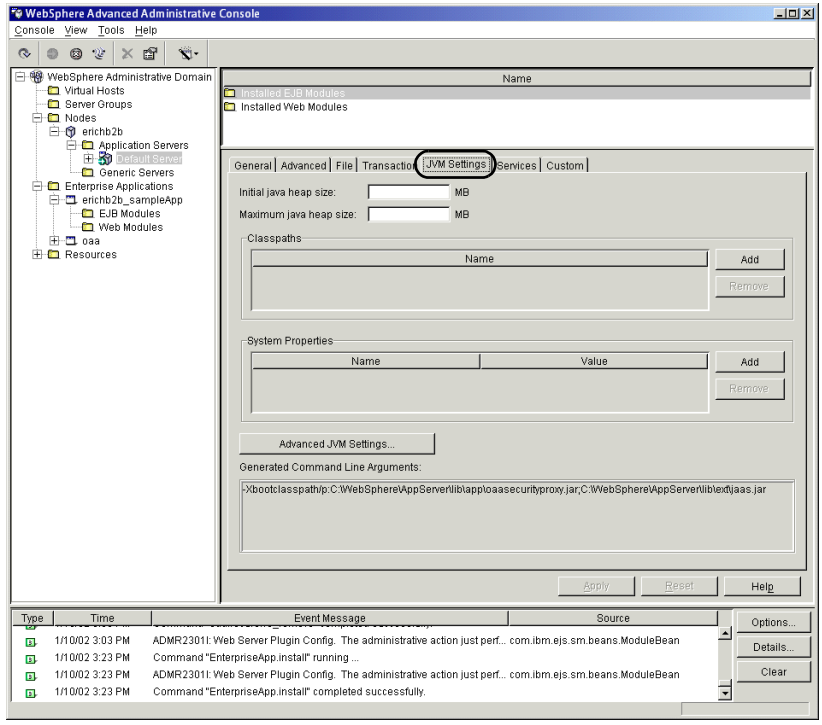
To set the Java heap size:

1 Verify that the WebSphere Admin Server has been started.

2 Open the WebSphere Advanced Administrator's Console (**Start** > **Programs** > **IBM WebSphere** > **Application Server** > **Administrator's Console**).

3 Click **Nodes** > <*System Name*> > **Application Servers** > <*Application server name*>.

The server settings page opens.



4 Click the JVM Settings tab.

5 Set the following JVM settings:

a **Initial java heap size**. Type `60`.

b **Maximum java heap size**. Type the value you want for heap memory. This setting should be at least 225 MB, but not more than 512 MB.

Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems.

Configuring a virtual directory for IBM HTTP Server

You must configure a virtual directory for Get-Services in your Web server. The following instructions assume that you are using WebSphere's built-in Web server – IBM HTTP Server. See your Web server documentation to determine how to create a virtual directory if you are using another Web server.

To configure a virtual directory for IBM HTTP Server:

1. Stop the IBM HTTP Server.
2. Open the file **httpd.conf** in any text editor. By default this file is located at:

   C:\IBM HTTP Server\conf
3. Add the following line to the end of the file:

   ```
   Alias /oaa/ "C:/WebSphere/AppServer/installedApps/oaa.ear/portal.
   <version>.war/"
   ```

   For <*version*>, enter the version number of the WAR file you installed.

   Important: The name you define for the virtual directory here must match the context root you defined in WebSphere.
4. Save the file.
5. Start the IBM HTTP Server.

Installing WebSphere Portal Server

You can configure Get-Services to display in a WebSphere Portal Server in one of two configurations:

- All Get-Services and WebSphere components running on a single system. See *Recommended WebSphere Portal Server configuration* on page 39.
- Get-Services components running on one system and WebSphere components running on another. See *Alternate WebSphere Portal Server configuration* on page 41.

Important: In either configuration, you must first install WebSphere Portal Server. See your WebSphere Portal Server documentation for details.

Recommended WebSphere Portal Server configuration

Use the following steps to configure Get-Services for the recommended WebSphere Portal Server configuration:

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 42.

Step 2 Generate a Get-Services WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Services WAR file* on page 43.

Step 3 Login to the Get-Services server and stop the WebSphere application server.

Step 4 Modify the archway.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the archway.xml file* on page 44.

Step 5 Modify the web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 44.

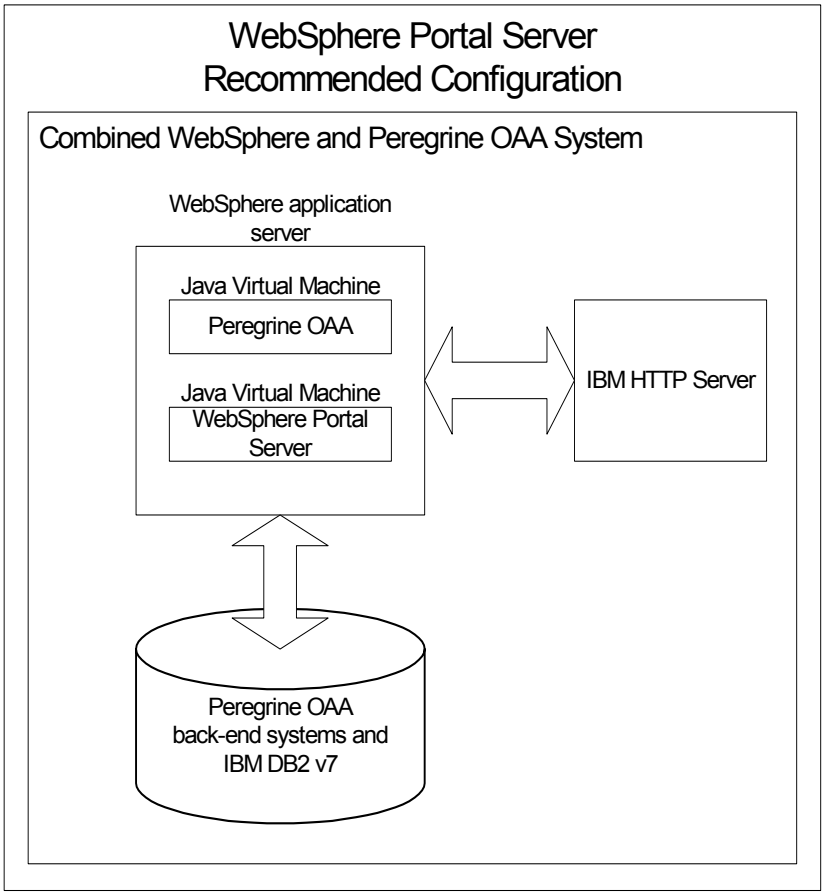
Step 6 Start the WebSphere application server.

Step 7 Deploy the Get-Services WAR file to WebSphere Portal Server. See *Deploying the Get-Services WAR file to WebSphere Portal Server* on page 45.

Step 8 Create places and pages in WebSphere Portal Server to display Get-Services portlets. See *Configuring WebSphere Portal Server places and pages* on page 46.

Step 9 Enable edit rights for Get-Services portlets. See *Enabling edit rights for Get-Services portlets* on page 46.

When complete, your installation will have the following configuration:



Alternate WebSphere Portal Server configuration

Use the following steps to configure Get-Services for the alternate WebSphere Portal Server configuration:

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 42.

Step 2 Generate a Get-Services WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Services WAR file* on page 43.

Step 3 Login to the Get-Services server and stop the WebSphere application server.

Step 4 Modify archway.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the archway.xml file* on page 44.

Step 5 Modify web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 44.

Step 6 Modify setDomain.js to call the SetDomain function. See *Modifying the setDomain.js file* on page 45.

Step 7 Start the WebSphere application server.

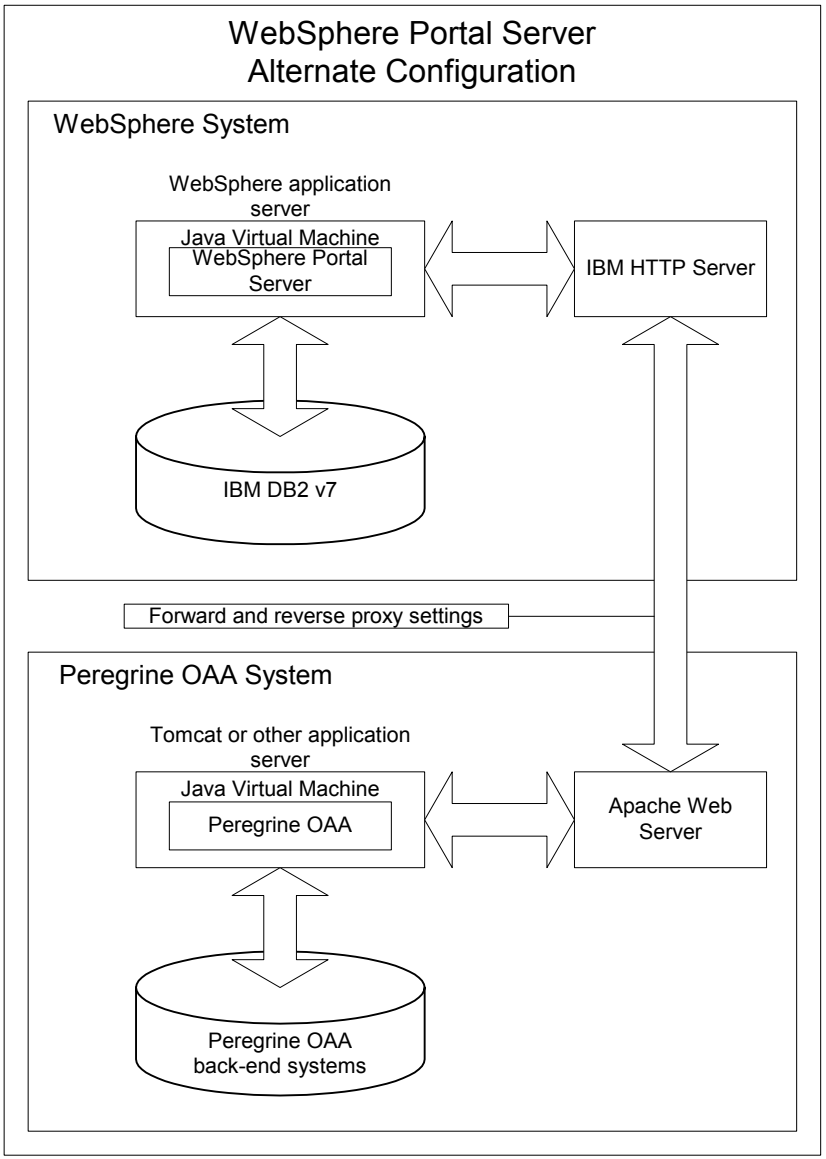
Step 8 Deploy the Get-Services WAR file to WebSphere Portal Server. See *Deploying the Get-Services WAR file to WebSphere Portal Server* on page 45.

Step 9 Create places and pages in WebSphere Portal Server to display Get-Services portlets. See *Configuring WebSphere Portal Server places and pages* on page 46.

Step 10 Enable edit rights for Get-Services portlets. See *Enabling edit rights for Get-Services portlets* on page 46.

Step 11 Modify IBM HTTP Server's httpd.conf file to add forward and reverse proxy URLs. See *Modifying httpd.conf for IBM HTTP Server* on page 47.

When complete, your installation will have the following configuration:



WebSphere Portal Server installation requirements

The recommended configuration of the WebSphere Portal Server requires the following items to be installed on the same server:

- WebSphere application server 4.0.2

- IBM HTTP Server 1.3.19
- IBM DB2 v7 database server
- WebSphere Portal Server
- A custom installation of Get-Services with WebSphere selected as the application server

The alternate configuration of the WebSphere Portal Server requires the following items be installed on a minimum of two servers:

- Server 1
  - WebSphere application server 4.0.2
  - IBM HTTP Server 1.3.19
  - IBM DB2 v7 database server
  - WebSphere Portal Server
- Server 2
  - Get-Services compatible application server
  - Web server
  - Back-end database for Get-Services
  - An installation of Get-Services

Generating a Get-Services WAR file

In order to display Get-Services in WebSphere Portal Server, you must first export the Get-Services portal components as a WAR file. You can then import this WAR file into WebSphere Portal Server, and choose the portal components you want to display as WebSphere Portal Server portlets.

To generate a Get-Services WAR file:

1. Login to the Get-Services administration page (admin.jsp).
2. Click **IBM WebSphere Portal Integration**.
3. Enter the following configuration information:
   a. **Source Path**. Enter the full path to the WebSphere.war in the Get-Services package folder. By default this folder is:

      <WebSphere>/oaa/packages

   b. **Destination Path**. Enter the full path and file name you want to use for the generated Get-Services WAR file.

c **Base URL**. Enter the full URL to the Get-Services deployment directory. By default this URL is:

http://<server>:<port>/oaa/servlet/basicauth

4 Click **Generate WAR file**.

Get-Services generates a new WAR file with the name and path specified in the Destination Path of step 3.

Modifying the archway.xml file

In order to login via WebSphere Portal Server, you configure Get-Services to use an alternate HTTP authentication method.

To modify the archway.xml file:

1 Using a text editor, open the archway.xml file located at:

<application server>\webapps\oaa\WEB-INF\default.

2 Edit the line containing:

```
<httpauthclass ...>HttpBasicAuthenticationManager</httpauthclass>
```

3 Change the value `HttpBasicAuthenticationManager` to `HttpAlternateAuthenticationManager`.

4 Save the file.

Modifying the web.xml file

You will need to enable the AuthController servlet to establish a proxy for HTTP basic authentication.

To modify the web.xml file:

1 Using a text editor, open the web.xml file located at:

<application server>\webapps\oaa\WEB-INF.

2 Add the following lines at the end of the last <servlet> definition:

```
<servlet>
  <servlet-name>AuthController</servlet-name>
  <display-name>AuthController</display-name>
  <description>A controller (decorator) servlet that can be used to
enable configurable auth protection of any resource.</description>

  <servlet-class>com.peregrine.oaa.archway.AuthControllerServlet
  </servlet-class>
  <load-on-startup>2</load-on-startup>
</servlet>
```

```
<servlet-mapping>
  <servlet-name>AuthController</servlet-name>
  <url-pattern>/servlet/basicauth/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>AuthController</servlet-name>
  <url-pattern>/servlet/auth/*</url-pattern>
</servlet-mapping>
```

3 Save the file.

Modifying the setDomain.js file

To use the alternate configuration of WebSphere Portal Server, you must enable the setDomain function.

Note: If you are setting up WebSphere Portal Server in the recommended configuration, you may skip these instructions.

To modify the setDomain.js file:

1 Login to the Get-Services server.

2 Stop your application server.

3 Using a text editor, open the setDomain.js file located at:

<application server>\webapps\oaa\js.

4 Add the following line to the end of the file:

```
setDomain();
```

5 Save the file.

Deploying the Get-Services WAR file to WebSphere Portal Server

After you deploy the Get-Services WAR file to WebSphere Portal Server, you can then configure the portlets you want to display, the display settings, and the access rights to each portlet.

See your WebSphere Portal Server documentation for detailed instructions.

To deploy the Get-Services WAR file:

1 Login to the WebSphere Portal as wpsadmin or another user with administrative rights.

2 Select **Portal Administration** from the Places menu.

3 Click **Portlets** > **Install Portlets**.

4 Click **Browse** and navigate to the Destination path you entered when you created the Get-Services WAR file.

5 Click Next to load the Get-Services WAR file.

WebSphere Postal Server displays a list of portlets to be installed.

6 Click Install.

WebSphere Portal Server installs the portlets and displays the message "Portlets successfully installed."

Configuring WebSphere Portal Server places and pages

You can deploy Get-Services portlets in any place or page that meet the following requirements.

Places

Your WebSphere Portal Server places must have the following characteristics:

- Supported markups must include HTML

Pages

Your WebSphere Portal Server pages must have the following characteristics:

- Supported markups must include HTML
- The page must be set to "allow all portlets that a user can access"
- All Get-Services portlets that you display in a page must grant "all authenticated users" the minimum edit permission.

Enabling edit rights for Get-Services portlets

WebSphere Portal Server users will need edit rights to the Get-Services portlets in order to add and customize them to their portal page.

To enable edit rights for Get-Services portlets:

1 Login to the WebSphere Portal as wpsadmin or another user with administrative rights.

2 Select **Portal Administration** from the Places menu.

3 Click **Security** > **Access Control List**.

4 Select the **Special groups** option and select **All authenticated users** from the select box.

5 From the Select the objects for the permissions select box, select **portlet applications**.

6 Select the Search on option, and then enter Peregrine in the Name contains field.

7 Click **Go**.

WebSphere Portal Server displays a list of portlets with Peregrine in the name.

8 In the Edit column, click **Select All** at the bottom of the table.

9 Click **Save**.

Users can now view and customize Get-Services portlets from the WebSphere Portal Server interface.

Modifying httpd.conf for IBM HTTP Server

In order to use the alternate configuration of WebSphere Portal Server, you will need to modify the httpd.conf file used by the IBM HTTP Server to add the forward and reverse proxy URLs to your remote instance of Get-Services.

Note: If you are setting up WebSphere Portal Server in the recommended configuration, you may skip these instructions.

To modify httpd.conf for IBM HTTP Server:

1 Login to the Get-Services server.

2 Stop your IBM HTTP Server.

3 Using a text editor, open the httpd.conf file located at:

C:\IBM HTTP Server\conf

4 Add the following lines to the end of the file:

```
ProxyPass /<oaa root>/ http://<server>:<port>/
<oaa root>/servlet/basicauth/
ProxyPassReverse /<oaa root>/ http://<server>:<port>/
<oaa root>/servlet/basicauth/
```

For *<oaa root>*, enter the name of the oaa virtual directory used by IBM HTTP Server. By default, this virtual directory is oaa.

For *<server>*:*<port>*, enter the server name and communications port number where Get-Services is installed.

5 Save the file.

Configuring WebSphere Translation Server for Get-Services

You can configure Get-Services to use a WebSphere Translation Server to provide real-time translations of on-screen data.

To configure WebSphere Translation Server for Get-Services:

Step 1 Copy the file wts.jar to the Get-Services deployment folder. See *Copying wts.jar to the Get-Services deployment folder* on page 48.

Step 2 Configure Get-Services to use the WebSphere Translation Server. See *Configuring Get-Services to use the WebSphere Translation Server* on page 48.

Copying wts.jar to the Get-Services deployment folder

The following instructions describe where to find and copy the file wts.jar.

To copy wts.jar to the Get-Services deployment folder:

1. Stop your application server.
2. Browse to the location of your WebSphere Translation Server installation.
3. Copy the file wts.jar from this folder.
4. Paste the file wts.jar into the Get-Services deployment folder located at:

   <Application server install>\WEB-INF\lib
5. Restart your application server.

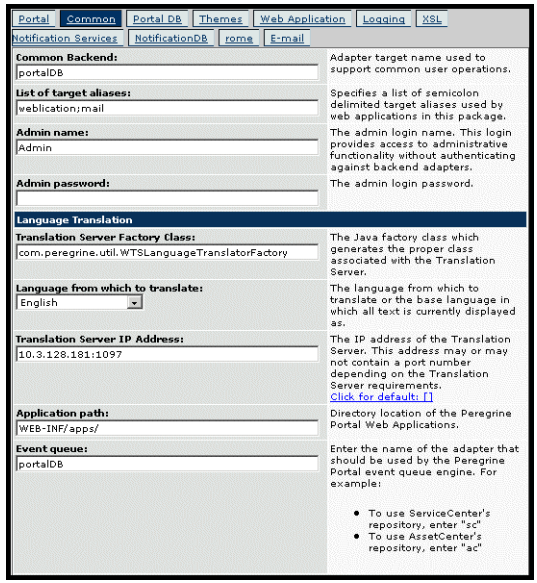
Configuring Get-Services to use the WebSphere Translation Server

The following instructions describe how to configure Get-Services to use the WebSphere Translation Server.

To configure Get-Services to use the WebSphere Translation Server:

1. Login to the Get-Services admin page (admin.jsp).
2. Click **Settings** > **Common** tab.

The Admin Settings page opens.



3 Enter the following configuration settings:

   a **Translation Server Factory Class**: Enter the Java factory class for the Translation server. The default Java factory class is:

   com.peregrine.util.WTSLanguageTranslatorFactory

   b **Language from which to translate**: Enter the source language that you want translated. The default value is English.

   c **Translation Server IP Address**: Enter the IP address and communications port to the Translation Server. For example: 10.3.128.181:1097.

   d Application path: Enter the relative path to the application serverapplications directory. The default value is:

   WEB-INF/apps/

   e **Event queue**: Enter the adapter name to you want to use for the event queue engine. The default value is: PortalDB

4 Click **Save**.

   The Control Panel opens.

5 Click **Reset Server**.

Translating on-screen data with a Translation Server

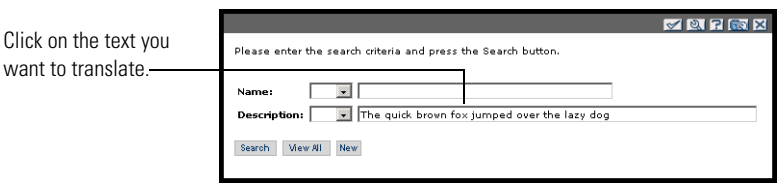
If you plan to store Get-Services data in a mixture of languages, you can configure Get-Services to send data to a Translation Server for real time translation. This interface will only translate data retrieved from the back-end database or manually typed into form inputs. If you need a translated user interface, you can purchase a Get-Services language pack directly from Peregrine Systems.

To translate on-screen data with a Translation Server:

1 Enable the translation server from the **Administration** > **Settings** page as described in *Configuring Get-Services to use the WebSphere Translation Server* on page 48.

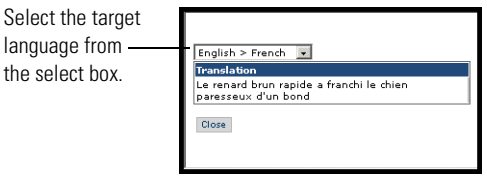
The translate button appears in the upper right tool bar.

2 Click on the source data or form input you want to translate.



3 Click the translate button.

The Translation window opens.



4 Select the target language to which you want to translate from the drop down select box.

The translation of your selection displays in the Translation box.

WebLogic 6.1 SP3 or SP4

The following procedures configure WebLogic to run Get-Services on Windows.

To configure WebLogic 6.3 SP3 or SP4:

Step 1 Stop both WebLogic and your Web server

Step 2 Edit the **startWebLogic.cmd** file to set the system password, memory settings, and start mode. See *Editing startWebLogic.cmd* on page 51.

Step 3 Edit the **Server.Policy** file to set the debug to true. See *Editing Server.Policy* on page 52.

Step 4 Run the Get-Services installer.

Step 5 Configure IIS to use **iisforward.dll** as an ISAPI filter and create an extension. See *Configuring the issforward.dll as an ISAPI filter and an extension* on page 52.

Step 6 Configure IIS to use iisproxy.dll as an ISAPI filter and create an extension. See *Configuring the iisproxy.dll as an ISAPI filter and an extension* on page 53.

Step 7 Create a virtual directory for Get-Services in your Web server. See *Creating a virtual directory for Get-Services* on page 54.

Step 8 Restart WebLogic and your Web server.

Editing startWebLogic.cmd

To edit startWebLogic.cmd:

1. Open the file **startWebLogic.cmd** file in any text editor. By default the file is located at:

   c:\bea\wlserver6.1\config<*mydomain*>****

2. Scroll to the following section of the script:

```
echo ************************************************
echo * To start WebLogic Server, use the password      *
echo * assigned to the system user. The system         *
echo * username and password must also be used to       *
echo * access the WebLogic Server console from a web   *
echo * browser.                                        *
echo ************************************************
@rem Set WLS_PW equal to your system password for no password prompt
server startup.
set WLS_PW=password
```

3 In the last line, change the word "password" to your WebLogic system password.

4 Search for the -mx parameter setting in the file. Change this setting to at least 225 MB, but not more than 512 MB.

Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems.

5 Set the STARTMODE variable to STARTMODE=false.

The first time you start WebLogic after the installation, you will need to start it in development mode for it to find the Web applications that have been deployed.

6 Add the following line before the end of the "goto finish" entry:

```
"-Djava.security.auth.login.config==<Weblogic>\lib\server.policy"
weblogic.server
```

For <*Weblogic*>, enter the installation path for Weblogic. By default this is:

c:\bea\wlserver6.1

7 Save the file.

Editing Server.Policy

To edit Server.Policy:

1 Open the file Server.Policy file in any text editor. By default the file is located at:

c:\bea\wlserver6.1\lib\

2 Add the following lines to the end of the file:

```
ServerLoginModule
{
  weblogic.security.internal.ServerLoginModule required debug=true;
};
```

3 Save the file.

Configuring the issforward.dll as an ISAPI filter and an extension

To establish a connection between WebLogic and IIS, you will need to install the file iisforward.dll as an ISAPI filter.

To install issforward.dll as an ISAPI filter and an extension:

1. Open Internet Services management console.
2. Right-click the **Default Web Site** node and then click **Properties**.
3. Click **Edit** from the Master Properties pane.
4. Click the **ISAPI Filters** tab.
5. Click **Add**.
6. Enter the following information:
   a. **Filter Name**: iisforward.
   b. **Executable**: issforward.dll. By default this file is located at:
      c:\bea\wlserver6.1\bin\issforward.dll
7. Click **OK**.
8. Click the **Home Directory** tab.
9. Click **Configuration**.
   The Application Configuration page opens on the App Mappings tab.
10. Click **Add**.
11. Enter the following information:
    a. Executable: issforward.dll. By default this file is located at:
       c:\bea\wlserver6.1\bin\issforward.dll
    b. Extension: .wlforward.
12. Close the Internet Services management console.

Configuring the iisproxy.dll as an ISAPI filter and an extension

To establish a connection between WebLogic and IIS, you will need to install the file iisproxy.dll as an ISAPI filter.

To install iisproxy.dll as an ISAPI filter and an extension:

1. Open Internet Services management console.
2. Right-click the **Default Web Site** node and then click **Properties**.
3. Click the **ISAPI Filters** tab.
4. Click **Add**.
5. Enter the following information:
   a. **Filter Name**: iisproxy.
   b. **Executable**: iisproxy.dll. By default this file is located at:

c:\bea\wlserver6.1\bin\iisproxy.dll

6 Click OK.

7 Click the Home Directory tab.

8 Click Configuration.

The Application Configuration page opens on the App Mappings tab.

9 Click Add.

10 Enter the following information:

a Executable: iisproxy.dll. By default this file is located at:

c:\bea\wlserver6.1\bin\iisproxy.dll

b Extension: .jsp.

11 Close the Internet Services management console.

Creating a virtual directory for Get-Services

To run Get-Services, you need to create a virtual directory in your Web server that maps to your WebLogic deployment folder. The typical installation creates a virtual directory called oaa, but you may specify a different virtual directory name.

Requirements for Get-Services virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Directory access rights	anonymous
Map to physical path	<WebLogic>\applications\oaa
Set Execute Permissions to	Scripts and Executables
Remove “allow anonymous access” to	default.asp, login.asp, e_login_main_start.asp
Set security access to only allow “System” and “Authenticated Users” to	default.asp, login.asp, e_login_main_start.asp

For <*oaa*>, enter the name of the virtual directory you want to use for Get-Services. Whatever name you enter here you will need to replicate in your application server configuration.

For <*WebLogic*>, enter the path to your WebLogic installation. By default the path is:

c:\bea\wlserver6.1\config\<*mydomain*>

JRun 3.1

The following procedures configure JRun to run Get-Services on Windows.

To configure JRun 3.1:

Step 1 Install a Java run-time environment. The Get-Services installer includes the Java 2 SDK Standard Edition v1.3.1_05. However, you can also use JRE 1.3.1 if you already have it installed. See *Custom installation components* on page 70.

Step 2 Install JRun from the Allaire Web site to the root of your hard drive (for example, C:\).

Step 3 Apply the latest JRun update. See *Applying the latest JRun update* on page 55.

Step 4 Deploy the Portal WAR file to JRun to create the necessary folder structure for Get-Services. See *Deploying the Portal WAR file to JRun* on page 56.

Step 5 Run the Get-Services installer.

Step 6 Move js.jar to the Java development kit ext folder. See *Moving js.jar to the Java development kit* on page 58.

Step 7 Run the JRun Connector Wizard to establish a connection between JRun and your Web server. See *Running the JRun Connector Wizard* on page 58.

Step 8 Configure your JRun Java settings. See *Configuring Java settings* on page 59.

Step 9 Create a virtual directory for Get-Services in your Web server. See *Creating a virtual directory for Get-Services* on page 61.

Step 10 Restart JRun and your Web server.

Applying the latest JRun update

Before you install Get-Services, you must apply the latest Jrun 3.1 update.

To install the latest JRun update:

1 Browse to the following URL:

http://www.macromedia.com/support/jrun/updates/3/updates_31.html

2 Click the link for the **JRun edition (Enterprise, Advanced, or Professional) and operating system of your server.**

3 Follow the installation instructions provided.

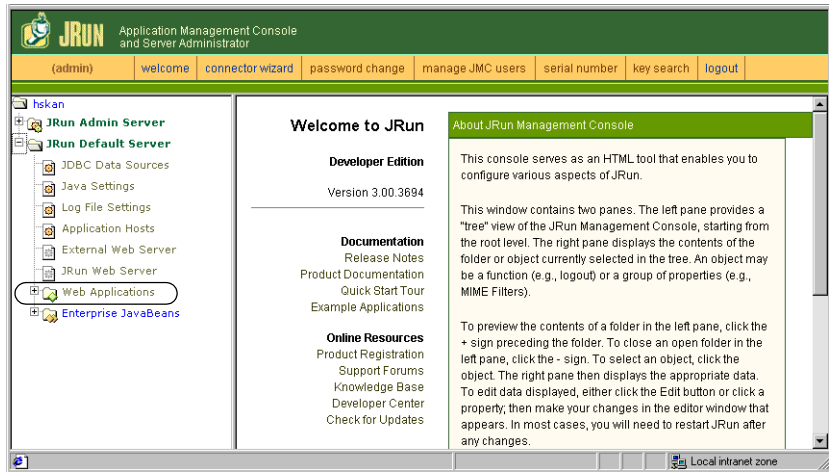
Deploying the Portal WAR file to JRun

The Portal WAR file creates the folder structure necessary to deploy Get-Services in your application server. After you have deployed this file to JRun you will be ready to run the Get-Services installer.

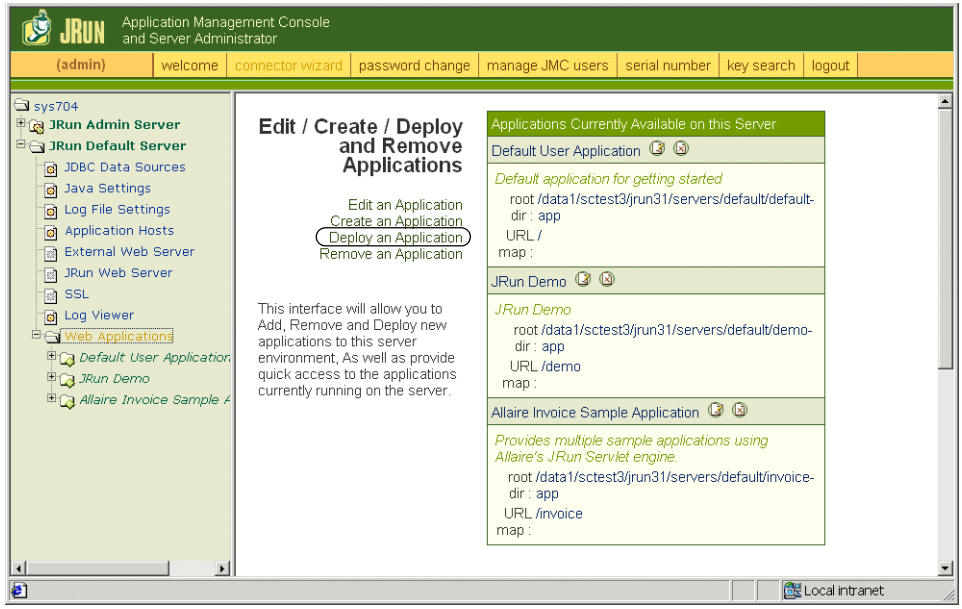
To deploy the Get-Services Portal WAR file to JRun:

1 Open the JRun Management Console and log in.

2 Select JRun Default Server > Web Applications.



The Edit / Create / Deploy and Remove Applications page opens.



3 Click the **Deploy an Application** link.

4 In the page that opens, fill out the fields as follows:

- Servlet War File or Directory:

  Browse to **<CD Rom Drive>:\oaa\packages\portal<version #>.war**.

For <version>, select the most current version.

Select this file, and then click **Accept**.

- JRun Server Name:

  Select **JRun Default Server**.

- Application Name:

  Type oaa.

- Application URL:

  Type /oaa.

- Application Deploy Directory:

  JRun generates this directory. Make a note of this path. You will need this information later in the procedure. Example:

  c:\JRun\servers\default\oaa

5 Click **Deploy**.

A message that OAA has been successfully deployed appears.

Moving js.jar to the Java development kit

JRun requires an updated version of js.jar in the Java development kit.

To move the js.jar to the Java development kit:

1 Stop any running JRun services.

2 Locate js.jar. By default this file is installed at:

<JRun installation>\servers\default\oaa\WEB-INF\lib

3 Cut and paste the file to your Java Development Kit ext folder. For example:

C:\Program Files\Peregrine\Common\jdk1.3.1_05\jre\lib\ext

4 Restart JRun.

Running the JRun Connector Wizard

The JRun Connector Wizard establishes a connection between JRun and your Web server.

To run the JRun Connector Wizard:

1 Login to the JRun Management Console.

2 Click Connector Wizard.

3 Select the JRun Default Server as the JRun Server Name.

4 Select your Web server from the drop down list box.

5 If your Web server uses a different IP address than your JRun server, enter the IP address of your JRun server in JRun Server IP Address.

6 Confirm that the JRun Server Connector Port is not in conflict with another communications port used on this server.

7 Enter the path to the Scripts Directory. For IIS 5.0 this value is:

C:\Inetpub\Scripts

8 Click Done.

Configuring Java settings

After you have installed Get-Services, you must configure the Java settings that JRun will use to run the Web application.

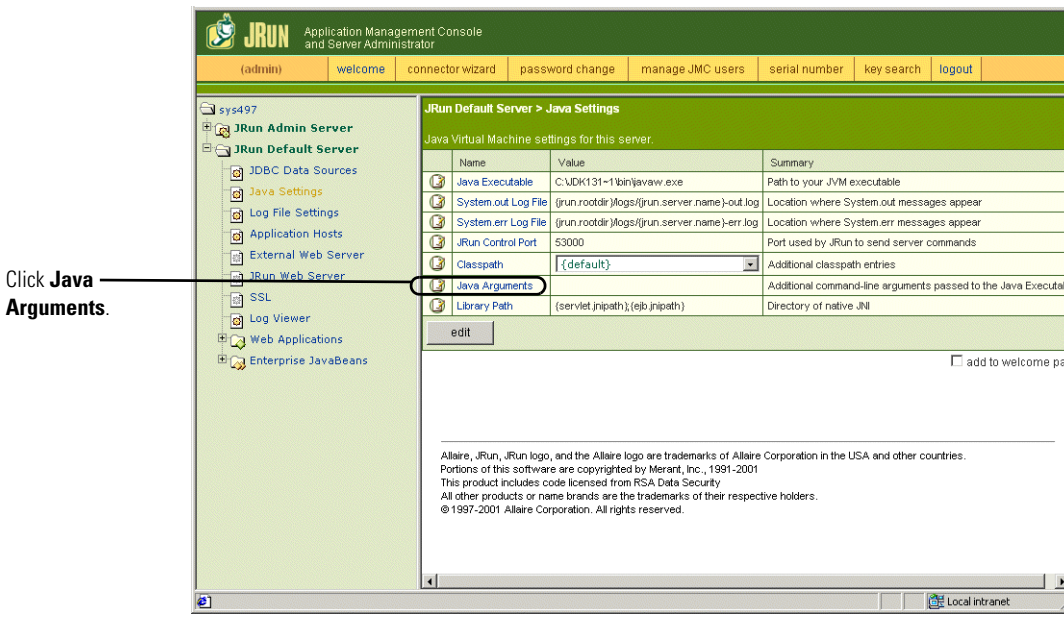
To configure Java settings:

1 Login to the JRun Management Console.

2 Click JRun Default Server > Java Settings.

The Java Settings page opens.

3 Click **Java Arguments**.



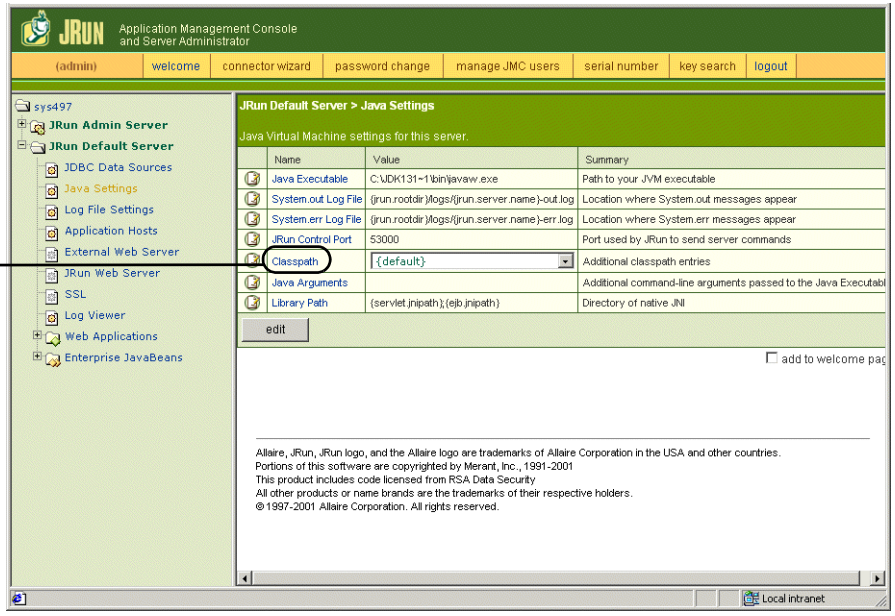
The Edit Window opens.

4 Enter an -Xmx value to define the maximum amount of heap memory allocated for your system. It is recommended that you set this value to at least 225 MB, but not more than 512 MB.

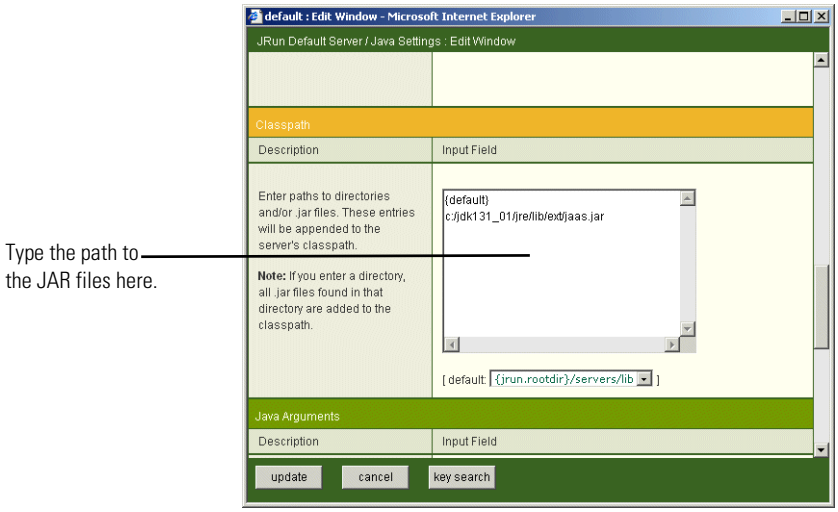
Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems. Applications using Persistence may require a higher setting.

5 On the Java Settings page, click **Classpath**.

Click **ClassPath**.



The Edit Window opens.



6 Enter the following classpaths:

- Java Development Kit ext folder. For example:

  C:\Program Files\Peregrine\Common\jdk1.3.1_05\jre\lib\ext

7 On the Java Settings page, click **Java Executable**.

8 Verify that path to your Java Development kit matches the path listed in the Classpath setting. For example:

  C:\Program Files\Peregrine\Common\jdk1.3.1_05\bin\javaw.exe

9 Click **update**.

10 Log out of the JRun Management Console.

Creating a virtual directory for Get-Services

To run Get-Services, you need to create a virtual directory in your Web server that maps to your JRun deployment folder. The typical installation creates a virtual directory called **oaa**, but you may specify a different virtual directory name.

Requirements for Get-Services virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Directory access rights	anonymous
Map to physical path	<JRun installation>\oaa
Set Execute Permissions to	Scripts and Executables
Remove "allow anonymous access" to	default.asp, login.asp, e_login_main_start.asp
Set security access to only allow "System" and "Authenticated Users" to	default.asp, login.asp, e_login_main_start.asp

For <*oaa*>, enter the name of the virtual directory you want to use for Get-Services. Whatever name you enter here you will need to replicate in your application server configuration.

For <*JRun*>, enter the path to your JRun installation. The recommended installation path is:

C:\JRun\servers\default

Typical installation option

A typical installation of Get-Services installs the most commonly used components of the product and saves application files and data in default destination directories. Most users choose Typical installation.

Typical installation components

Following is a brief description of the components that are automatically installed with a Typical installation of Get-Services:

Applications and File Locations

Get-Services Component	Default Installation Directory
Apache Web Server	C:\Program Files\Peregrine\Common\Apache2
Tomcat Application Server	C:\Program Files\Peregrine\Common\Tomcat4
Java Development Kit	C:\Program Files\Peregrine\Common\jdk1.3.1_05
OAA Platform and Get-Services	C:\Program Files\Peregrine\oaa

Services

The installation program will also create and start the following services on your Windows server:

- Apache Web Service
- Peregrine Tomcat Service

Important: If you are already running another web or application server on the target Windows system, you will need to stop the service(s) for the application(s) before beginning the Get-Services setup program. If you do not stop these services before beginning the install process, the setup program will complete successfully, however, it may not create the Apache and Peregrine Tomcat services needed to run Get-Services.

Communications ports

Get-Services uses the following communications ports in a typical installation. After installation, you can configure Get-Services to use one or more of the alternate communications ports if your local network already uses these communications ports.

Default port	Component used by	Alternate port
80	Apache Web Server	8081
8005	Tomcat application server administration	8015
8009	Tomcat application server worker file	8019
8011	Tomcat application server worker file for load balancing (optional).	8021
8013	Tomcat application server worker file for load balancing (optional).	8023
8015	Tomcat application server worker file for load balancing (optional).	8025

Note: To change settings for these components or to use or install different components, use the Custom installation option for Get-Services.

Typical installation procedures

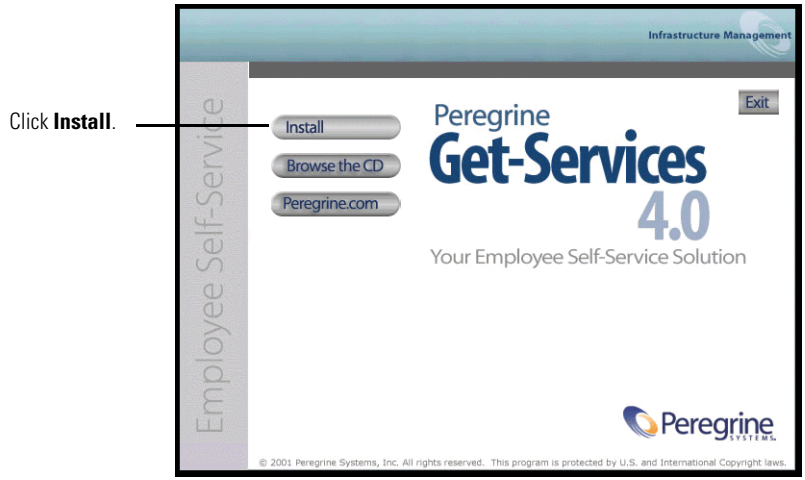
This section explains how to install Get-Services with a Tomcat application server and an Apache web server on a Windows operating system.

To perform a typical installation of Get-Services on Windows:

1 Insert the Get-Services installation CD into your computer's CD ROM drive. Your computer should automatically launch the installation program.

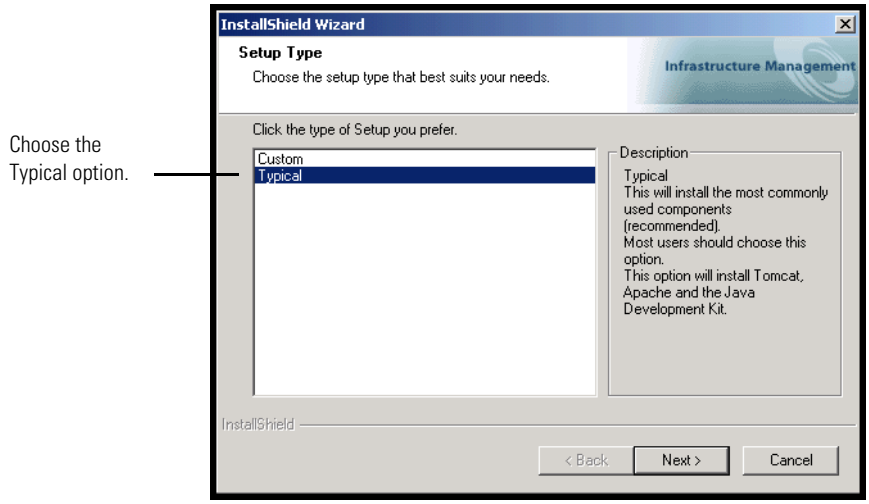
If the installation program does not automatically start, using the Windows **Start** > **Run** command, browse to the CD ROM drive and open **Setup.exe**.

2 In the main installation program screen, click **Install**.



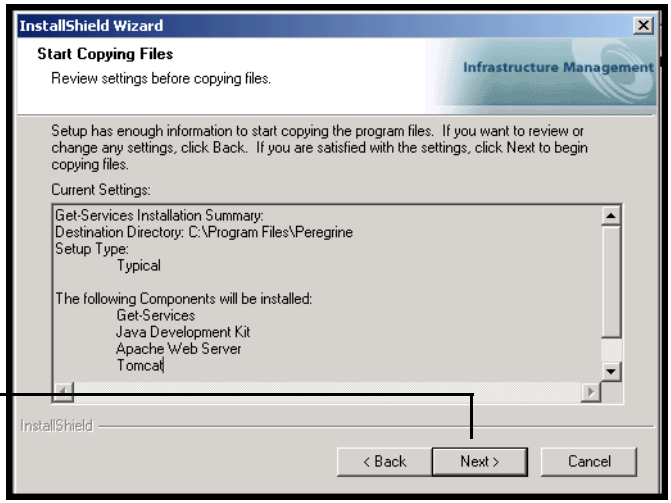
Status messages indicate that the Setup program is preparing the InstallShield Wizard.

3 In the Setup Type screen, select **Typical**, and then click **Next**.



The installer displays status messages to validate the location of the Host system name.

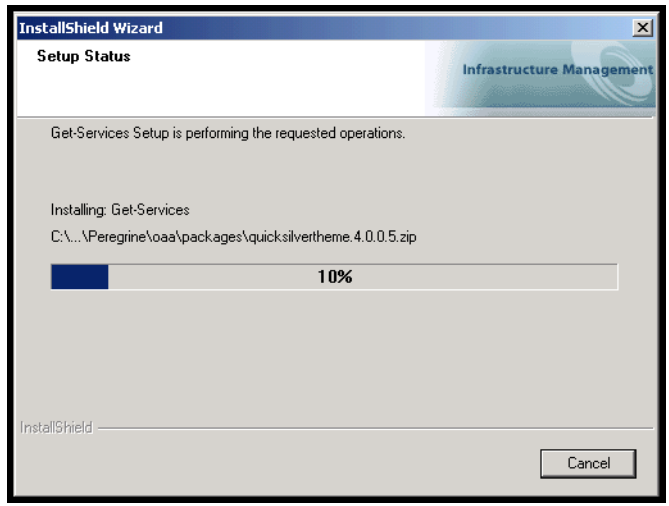
4 In the Start Copying Files screen, click **Next**.



Click **Next**.

If you are installing the Change Managment module, the component list includes Get-Services-Change.

The Setup Status screen that appears shows that Get-Services is installing the files to your system.

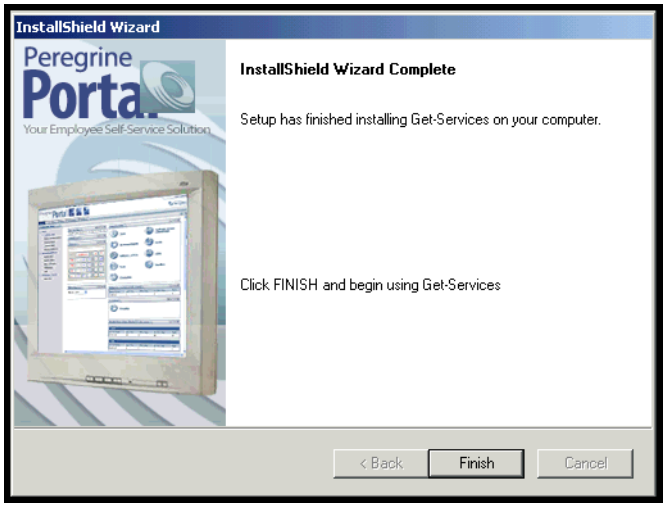


A status message indicates that the Get-Services packages are being deployed to your server. This phase can take several minutes.

While this status message is displayed, the setup program is performing the following:

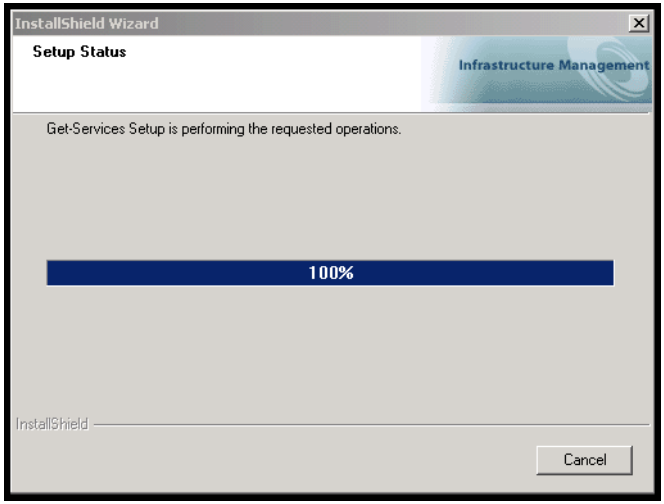
- Copying the documents
- Creating and starting the “Apache” and “PeregrineTomcat” services

5 In the InstallShield Wizard Complete screen, click **Finish**.



If you are installing the Change Management module, the text in the dialog box reads, "Setup has finished installing Get-Services-Change on your computer. Click FINISH and begin using Get-Services-Change."

The setup program completes the final steps of the installation.



If you are installing the Change Management module, the text in the dialog box reads, “Get-Services-Change Setup is performing the requested operations.”

6 After the InstallShield Wizard screen closes, access Windows Services to verify that the “Apache” and “Peregrine Tomcat” services were created. Status for both services should display “Started”.

If one or both of these services did not startup, refer to *Troubleshooting* to troubleshoot possible install issues.

This completes the procedures required for a Typical installation of Get-Services on a Windows operating system server.

Custom installation option

The following section describes how to perform a custom installation of Get-Services on a Windows operating system server, including overview steps for a Development and Production environment.

Custom installation components

Following is a brief description of the components that are available for a custom installation of Get-Services:

Application options

Get-Services Component	Options
Web Server	■ Apache 2.0.43 ■ IBM HTTP Server 1.3.19 ■ Microsoft IIS 5.0
Application Server	■ Tomcat 4.1.12 ■ WebSphere 4.02 ■ WebLogic 6.1 SP3 ■ JRun 3.1
Java Development Kit	Java 2 SDK
OAA Platform and Get-Services	Change Management

Communications ports

The communications ports used by a custom installation of Get-Services depend upon the application components that you select. Refer to your Web and application server documentation to determine what communications port they require. After installation, you can configure Get-Services to use alternate communications ports if your local network already uses particular communications ports.

Get-Services on servers running Oracle 9.2.0.1

If you are running Get-Services on a server using Oracle 9.2.0.1 you may experience a port conflict over communications ports 8009 and 8080. Consult your Web and application server documentation to see if they use either of these two ports.

If you are using Tomcat as your application server, then by default, there will be a port conflict over port 8009. It is recommended that you change Tomcat to use a different communications port on servers running Oracle 9.2.0.1.

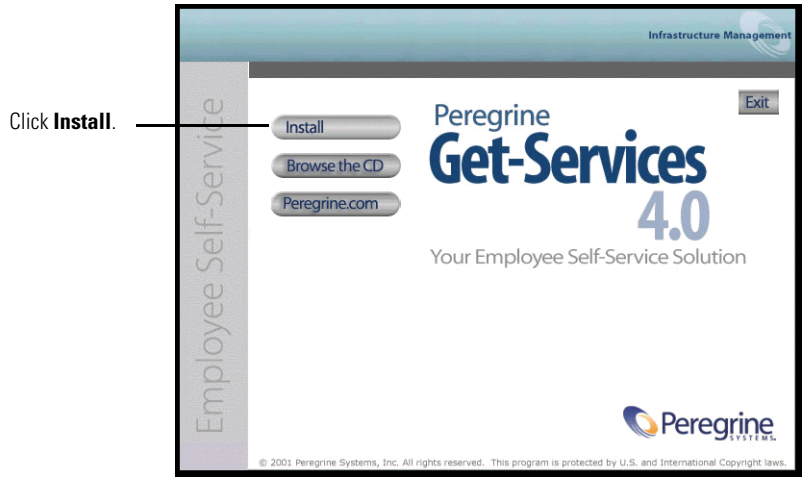
Custom installation procedures

To perform a custom installation of Get-Services on Windows:

1 Insert the Get-Services installation CD into your computer's CD ROM drive. Your computer should automatically launch the installation program.

If the installation program does not automatically start, using the Windows **Start** > **Run** command, browse to the CD ROM drive and open **Setup.exe**.

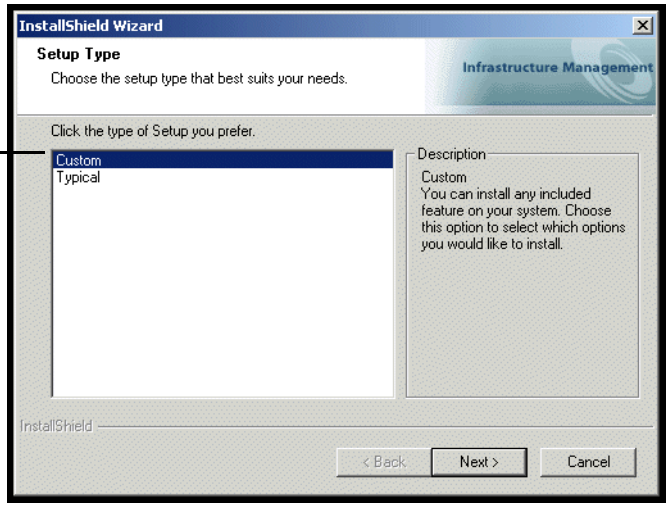
2 In the main installation program screen, click **Install**.



Status messages indicate that the Setup program is preparing the InstallShield Wizard.

3 In the Setup Type screen, select **Custom**, and then click **Next**.

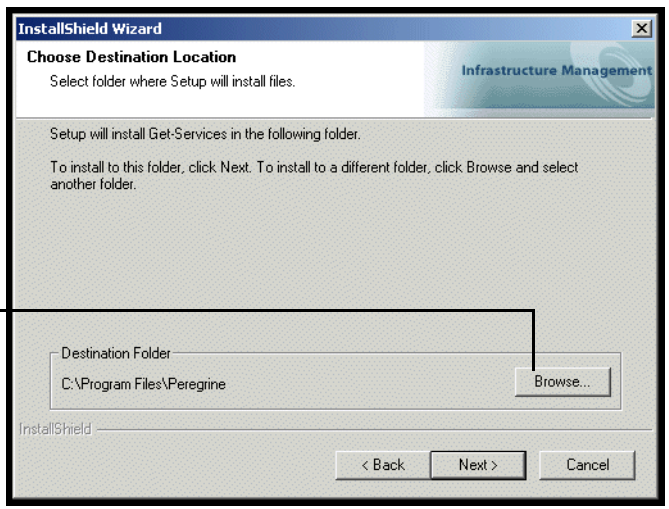
Select **Custom**



4 In the Choose Destination Location screen, browse to the location where you want Get-Services installed. Click **Next** to continue.

The default location is C:\Program Files\Peregrine.

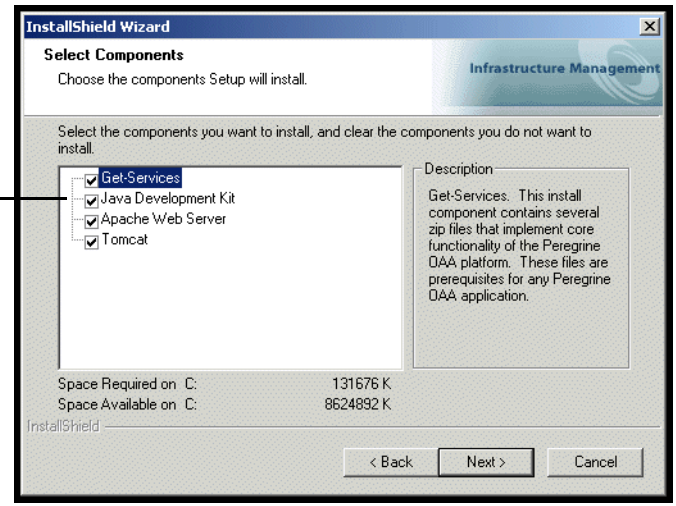
Select **Browse** to specify a different location.



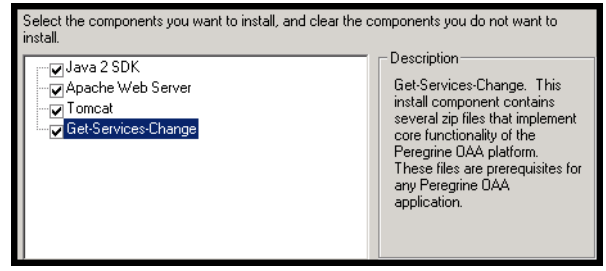
If you are installing the Change Management module, the text in the dialog box reads, "Setup will install Get-Services-Change in the following folder."

5 In the Select Components screen, choose the components that you want installed on this machine and click **Next**.

Select components to install



If you are installing the Change Management module, the component list includes Get-Services-Change.



For a development environment, either select all components or clear the components that you will install manually or for which you have alternate software. For example, clear the Tomcat option to install an alternate application server.

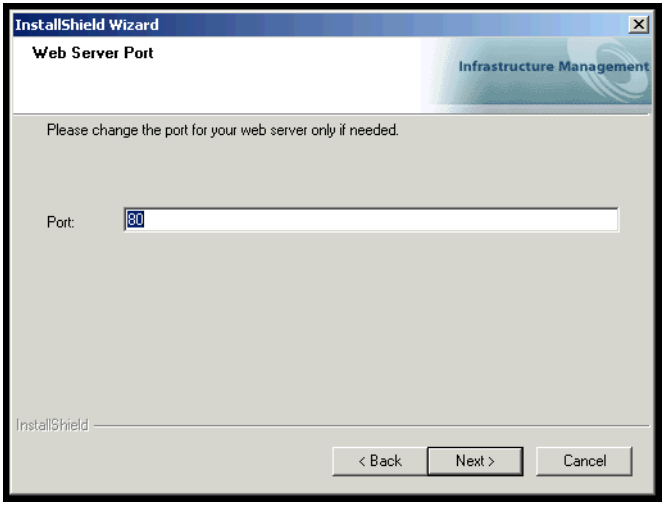
For a production environment, select the components you want to run from this machine.

- Get-Services. Installs the programs files necessary for Get-Services. These files need to be on the same machine as the primary application server.

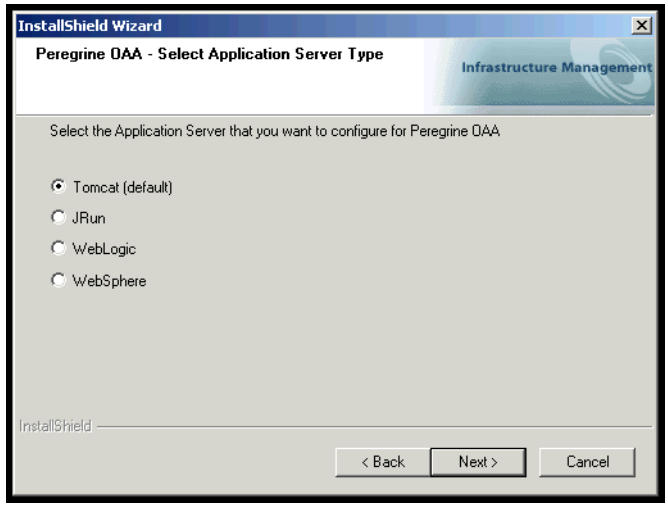
- Java Development Kit. Installs the Sun Microsystems Java Development Kit (Java 2 SDK). This component needs to be installed on every machine running an application server.
- Apache Web Server. Installs the Apache Web server. Get-Services requires a Web server in order to serve pages. See *Custom installation components* on page 70 for a list of available Web servers. If you also select the Tomcat application server, then the installer will automatically configure a connection between Apache and Tomcat.
- Tomcat. Installs the Tomcat application server. Get-Services requires at least one application server to process Java applications. You may also install multiple instances of your application server for load balancing. See *Custom installation components* on page 70 for a list of available application servers. If you also select the Apache Web server, then the installer will automatically configure a connection between Apache and Tomcat.

Status messages indicate the validation and location of the Host system name.

6 In the Web Server Port screen, select the communication port you want to use for Web server. The default communications port number for Get-Services is port 80. To continue, click **Next**.



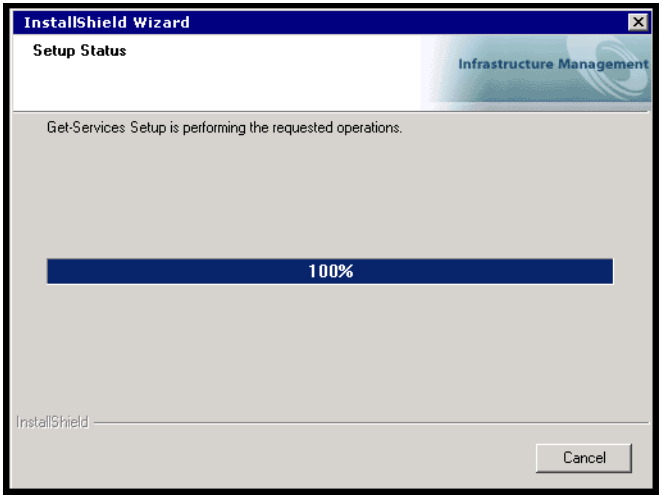
7 In the Application Server Type screen, select the application server that you would like to use with Get-Services. The default application server is Tomcat. To continue, click **Next**.



8 In the Start Copying Files screen, verify the Custom installation components. To review or change settings, click **Back**. To continue with the installation, click **Next**.

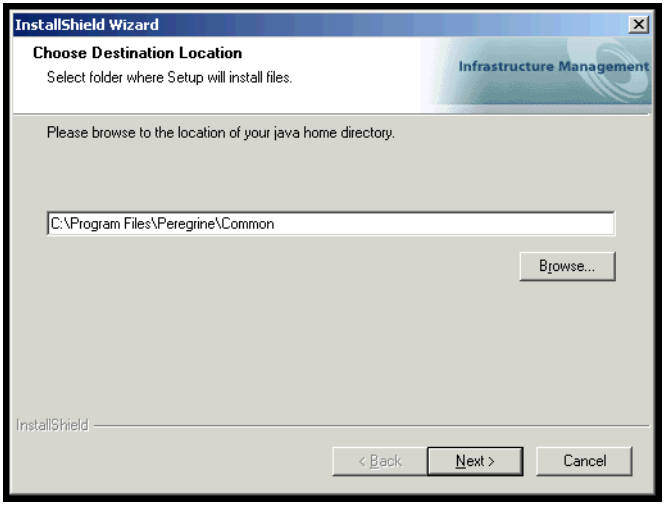
If you are installing Get-Services-Change, the dialog box will reflect the Get-Services-Change files

The Setup Status screen is displayed while the installation program performs the requested operations.

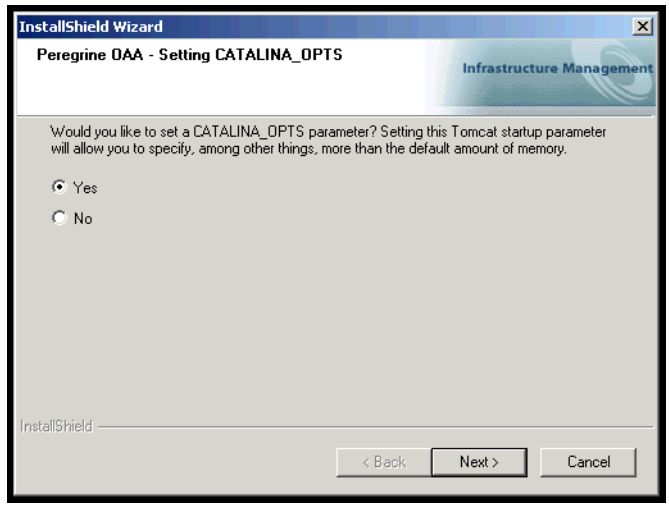


If you are installing Get-Services-Change, the dialog box will reflect Get-Services-Change.

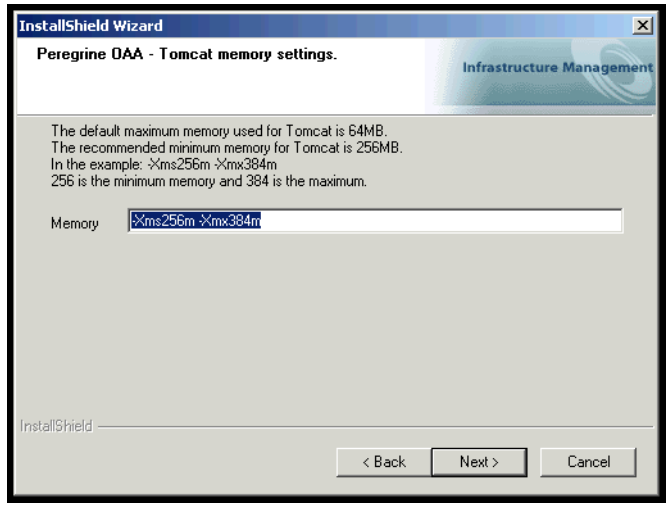
9 In the Choose Destination Location screen, browse to the location where you would like the Java application files installed. To continue, click **Next**.



10 In the Setting CATALINA_OPTS screen, select whether or not you would like to configure this parameter for Tomcat memory settings.



- To configure this parameter, select **Yes,** then click **Next.** The installation program will display a Tomcat Memory Settings screen. Modify the minimum and maximum memory range used for Tomcat as necessary, then click **Next** to continue with the installation.

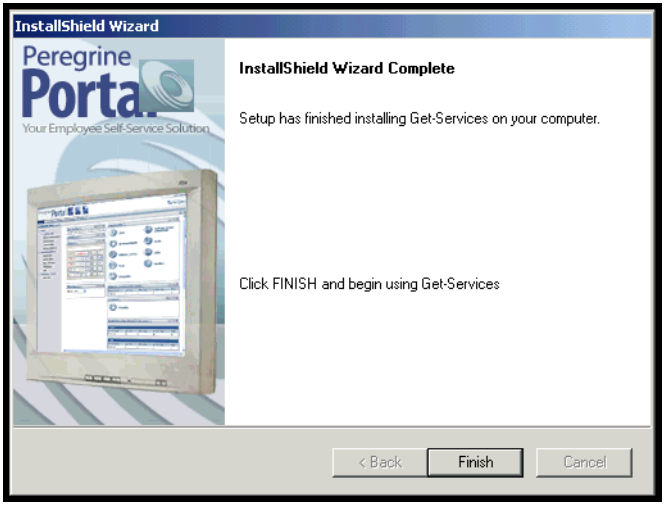


- To bypass configuration of this parameter, select **No**, then click **Next**.

A status message indicates that the Get-Services packages are being deployed to your server.

This phase can take several minutes.

11 In the InstallShield Wizard Complete screen, click **Finish** to end the Get-Services installation program.



If you are installing Get-Services-Change, the dialog box reads: Setup has finished installing Get-Services-Change on your computer. Click FINISH and begin using Get-Services-Change.

12 After the InstallShield Wizard screen closes, access Windows Services to verify startup of your web and application server services.

If one or both of these services did not startup, refer to *Troubleshooting* to troubleshoot possible installation issues.

This completes the procedures required for a Custom installation of Get-Services on a Windows operating system server.

Uninstalling Get-Services

Follow these procedures to uninstall Get-Services from your Windows system:

1. Access the Windows Add/Remove Programs utility.
2. Select **Peregrine Portal 4.0** and click **Change/Remove**.

   A status message indicates that the setup program is preparing the InstallShield wizard to guide you through the process.
3. The Close Programs screen opens if any Get-Services services or applications are running. Click **Next** to continue.
4. The verification message box opens, "Are you sure you want to stop shared applications and/or services?" Click **Yes** to continue.

   Status messages indicate the termination of the services for Apache and Tomcat.
5. The Confirm Uninstall dialog box opens, "Warning, you are about to uninstall all Peregrine Portal 4.0 applications and data." Click **Yes** to remove Get-Services.

   Important: Back up any data you want to save before continuing.

6. The Shared Files screen opens if there are any shared files to be removed during setup.

   If WebSphere is installed on this computer, setup will prompt you confirm the removal of six JAR files. Click **No** or **No to All** to retain these JAR files.

   Warning: Do not remove the shared JAR files as the WebSphere Advanced Administrative Console requires these files to function.

   If there are no shared filed to remove, then a status message indicates that the uninstall program is removing files from your computer.
7. The Maintenance Complete screen opens. Click **Finish** to complete the uninstall of Get-Services.

Testing your installation

Use the following steps to confirm that you have properly installed Get-Services on Windows.

To test your Get-Services installation:

1. Verify that your application and Web servers are started.
2. Open a Web browser and type the following in the Address field:

   http://<server name>:<port>/oaa/admin.jsp

   For <*server name*>, enter the server name where the Get-Services Web server resides.

   For <*port*>, enter one of the following communications port numbers:

Application Server used	Port Number
WebSphere	9080
WebLogic	7001
JRun	80, can be omitted from URL
Tomcat	80, can be omitted from URL

If everything is configured properly, the Administrator login page opens.

If the Get-Services administration login page does not open, see *Troubleshooting* for more information.

CHAPTER 3

Installing on AIX, Linux, or Solaris

This chapter covers the following topics:

Choosing an installation environment

You can install Get-Services in one of two installation environments:

- Development environment
- Production environment

The Get-Services development environment is intended for you to evaluate product features and customize your installation prior to deployment in a production environment. In a development environment, you install all software required for Get-Services on one computer system.

You have two choices of development environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Services deployed on Tomcat 4.1.12 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Services

The Get-Services production environment is intended to maximize server performance and scalability, and to deploy any customizations you have made. In a production environment, you install the various components of Get-Services on different servers to maximize performance.

You have two choices of production environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Services deployed on multiple instances of Tomcat 4.1.12 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Services

Development Environment

The following procedures describe how to install Get-Services in a development environment.

To install Get-Services in a typical development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Run the Get-Services installer and select the Typical installation option. See *Typical Installation Option* on page 120.

Step 4 Configure the back-end databases and create Get-Services users.

To install Get-Services in a custom development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Install alternate application and Web servers.

Step 4 Configure the alternate application server for Get-Services. See *Configuring alternate application servers* on page 89.

Step 5 Run the Get-Services installer and select the Custom installation option. See *Custom Installation Option* on page 128.

Step 6 Configure the back-end databases and create Get-Services users.

Production Environment

The following procedures describe how to install Get-Services in a production environment.

To install Get-Services in a typical production environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services on a separate server.

Step 3 Run the Get-Services installer and select the Typical installation option. See *Typical Installation Option* on page 120.

Step 4 Configure multiple instances of Tomcat for load balancing on the Apache Web server.

Step 5 Configure the back-end databases and create Get-Services users.

To install Get-Services in a custom development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Install the back-end database required for Get-Services.

Step 3 Install the alternate application server and Web server on separate servers.

Step 4 Configure the alternate application server for Get-Services. See *Configuring alternate application servers* on page 89.

Step 5 Run the Get-Services installer and select the Custom installation option. See *Custom Installation Option* on page 128.

Step 6 Configure the Web servers and application servers for load balancing.

Step 7 Configure the back-end databases and create Get-Services users.

Migrating Get-Services from previous versions

To migrate older versions of Get-It or Get-Services to Get-Services 4.0 requires both a manual data migration process and the recreation of any interface customizations you have made. The following steps describe the migration process.

To migrate previous versions to Get-Services 4.0:

Step 1 Review the customizations of previous version and determine which customizations need to be recreated in Get-Services 4.0. See *Recreating customizations in Get-Services 4.0* on page 85.

Step 2 Install Get-Services 4.0 on a new system. See *Choosing an installation environment* on page 82.

Step 3 Apply any required configuration changes to the back-end database you want to migrate to Get-Services 4.0. See *Configuring an existing back-end database for Get-Services 4.0* on page 88.

Recreating customizations in Get-Services 4.0

You cannot directly migrate customizations implemented in previous versions to Get-Services 4.0. Instead, you must recreate your changes using the new features and methods available in Get-Services 4.0.

The following sections describe how to recreate your customizations from previous versions.

No customizations

If you have made no customizations to Get-Services, you can simply install Get-Services 4.0 on a new system and migrate your data from your existing back-end database.

Customized JSP files

In previous versions, customers had to directly modify JSP files in order to add or remove certain functionality. The following table describes how to recreate some of the more common JSP file modifications.

JSP file modification	New method to use
Remove the user self-registration option from login page	Enable or disable the user registration option from the Administration Settings page
Remove the change password option from the login page	Enable or disable the change password option from the Administration Settings page

Personalized pages

Get-Services 4.0 offers many more pages that you can personalize directly from the Web interface. If you personalized pages in a previous version, you must recreate your personalized pages in Get-Services 4.0 using DocExplorer.

You can use personalization to:

- Add or remove fields from a page
- Save a personalized search results or details on your portal page

Customized skins, stylesheets, and themes

Get-Services 4.0 has combined all interface images and stylesheets into themes. Users can no longer select separate skins and stylesheets. The new themes consist of skins (which themselves are composed of image files, frame definitions, and layer files), cascading stylesheet definitions, and XSL templates.

Although you may copy over older custom themes to Get-Services 4.0, you may experience rendering errors due to the new images, CSS definitions, frame definitions, and layers. It is recommended that you recreate any custom themes using the Get-Services 4.0 version of the classic theme as your template.

Alternate login pages and authentication methods

If you used a custom login page or an alternate authentication method in a previous version, you can re-use or recreate these customizations using the updated instruction. You can find information about alternate security methods in the *Get-Services Administration Guide*.

Customizations made with a previous tailoring kit

Many customizations that required a tailoring kit in previous versions can now be done directly from the Get-Services Web interface. The following table describes how to recreate some of the more common tailoring kit changes.

Tailoring kit modification	New method to use
Added or removed fields form a form	Add or remove fields from Personalization
Added a new language or locale to the Get-Services interface	Create and edit language strings files directly. You may also purchase officially supported language packs from Peregrine Systems
Made changes to the common, portal, or Peregrine Studio packages	These packages are no longer available for tailoring, however most common interface settings can now be customized from the Administration Settings page.
Made changes to schemas or ECMA server-side scripts	Review new functionality and determine if you still need the customized scripts and schemas. If you do need the customizations, you will to recreate them in the current version of the Get-Services tailoring kit.

Configuring an existing back-end database for Get-Services 4.0

The following table lists the options available for data migration.

Get-Services 2.3 to Get-Services 4.0

Back-end version	Migration required
ServiceCenter 3.0	Upgrade to ServiceCenter 4.x or 5.0.x
ServiceCenter 4.x	Apply Get-Services 4.0 unload files to existing ServiceCenter 4.x
Service 5.0.x	Apply Get-Services 4.0 unload files to existing ServiceCenter 5.0.x

Configuring alternate application servers

You must install a Java-enabled application server to support your Peregrine Web applications. Peregrine OAA supports the following alternate application servers:

- *Tomcat 4.1.12*
- *WebSphere 4.0.2*
- *WebLogic 6.1 SP3 or SP4*
- *JRun 3.1*

The Get-Services typical installation option installs Tomcat 4.1.12 and connects it to an Apache 2.0 web server. You can also install Tomcat 4.1.12 using the custom installation option.

Important: If you want to use an application server other than Tomcat 4.1.12, then you must configure your application and Web servers *prior* to running the Get-Services installer.

See the following sections for instructions configuring alternate application servers for Get-Services.

Tomcat 4.1.12

If you use select both the Apache Web server and the Tomcat application server for your installation of Get-Services, then the Get-Services installer will automatically configure a connection between Apache and Tomcat. In addition, if you select a typical installation, then the installer will automatically pre-compile the Get-Services JSP files. If you are running a custom installation, you can manually pre-compile the Get-Services JSP files when you are ready to deploy into a production environment.

Precompiling JSP files for a Tomcat production environment

If you plan to use Tomcat 4.1.12 in a production environment or on a system with multiple processors, you will need to pre-compile the JSP files deployed for Get-Services. This extra configuration step is the result of a known issue with Tomcat. See the following URL for more details about this issue:

http://nagoya.apache.org/bugzilla/show_bug.cgi?id=14077

To pre-compile JSP files for Tomcat 4.1.12:

1 Stop the Tomcat application server.

2 Open a command prompt.

3 Change directories to usr/local/peregrine/samples.

4 Copy the following files to the locations specified:

Copy this file	to this location
excludes.txt	<Tomcat>/webapps
builds.xml	<Tomcat>/webapps
precompile.sh	<Tomcat>/bin

For <*Tomcat*>, enter the path to your Tomcat installation. For example: /usr/local/peregrine/common/Tomcat4

5 Enter one of the following commands based upon your operating system:

Operating system	Command required
UNIX ksh	export JASPER_HOME=<*Tomcat directory*>
UNIX csh	setenv JASPER_HOME <*Tomcat directory*>

For <*Tomcat directory*> enter the absolute path to your Tomcat installation.

6 Change directories to the Tomcat bin folder.

Operating system	Command required
UNIX, Solaris, AIX	cd $JASPER_HOME/bin/samples/

7 Run the precompile batch file.

Operating system	Command required
UNIX, Solaris, AIX	./precompile.sh <*Web app name*> <*Tomcat instance name*>

For <*Web app name*> enter the name of the Get-Services deployment folder. You may omit this name if you are using the default folder named oaa.

For <*Tomcat instance name*> enter the name of the Tomcat instance you have installed. You may omit this name if you are using the default instance named Standalone.

The batch file displays the progress of the conversion. When it is complete, the command prompt returns.

8 Start the Tomcat application server.

WebSphere 4.0.2

Use the following procedures to configure WebSphere to run Get-Services on AIX, Linux, and Solaris.

To configure WebSphere 4.02:

Step 1 Install WebSphere 4.02. Your version of WebSphere 4.0.2 includes the IBM HTTP Server.

Step 2 Deploy the Portal WAR file to WebSphere to create the necessary folder structure for Get-Services. See *Deploying the Portal WAR file to WebSphere* on page 92.

Step 3 Set the JVM Java heap size for each WebSphere instance running Get-Services. See *Setting the Java heap size* on page 94.

Step 4 Create the virtual directory you want to use for Get-Services in your Web server. See *Configuring a virtual directory for IBM HTTP Server* on page 96.

Step 5 Run the Get-Services installer.

If you plan on setting up a WebSphere Portal Server or a WebSphere Translation Server, see *Installing WebSphere Portal Server* on page 97 or *Configuring WebSphere Translation Server for Get-Services* on page 106.

Deploying the Portal WAR file to WebSphere

The Portal WAR file creates the folder structure necessary to deploy Get-Services in your application server. After you have deployed this file to WebSphere you will be ready to run the Get-Services installer.

To deploy the Portal WAR file to WebSphere:

1. Verify that the WebSphere Admin Server has been started.
2. Open the WebSphere Advanced Administrator's Console (/WebSphere/AppServer/bin/adminclient.sh).
3. On the menu at the left side of the console, right-click on **Enterprise Applications** and select **Install Enterprise Application**.
4. On the screen displayed, do the following:
   a. Select **Install stand-alone module**.
   b. In the **Path** field, browse to the path to the portal<version #>.war file. The default is <CD Rom Drive>/portal<version #>.war.

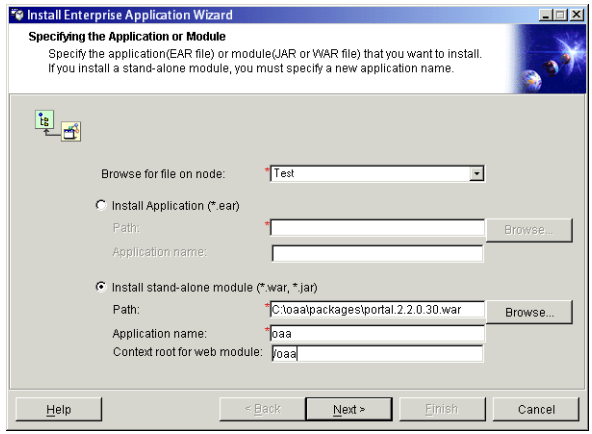
For <version #>, Select the most recent version available (4.0.0.44 or greater).

c In the **Application Name** field, type `oaa`.

d In the **Context Root** field, type the name of Get-Services virtual Web server directory you wish to use. Example: `/oaa`.

Important: You must create a Web server virtual directory matching the context root you enter here.

The following screen shows the completed form.

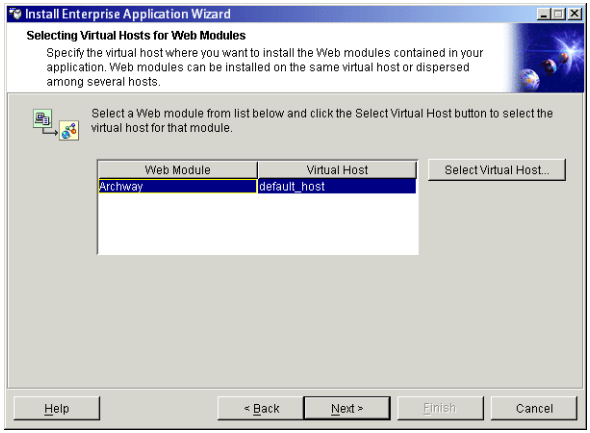


5 Click **Next**.

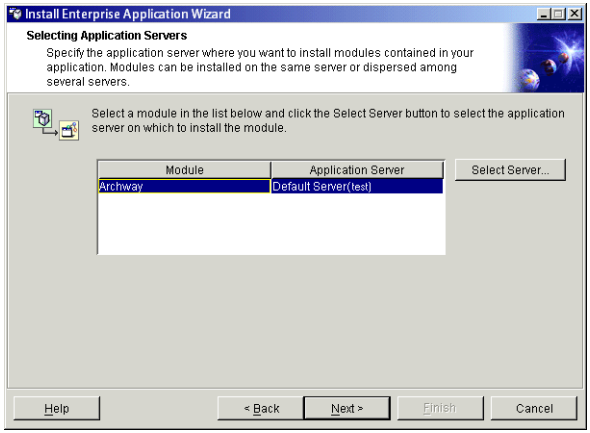
6 Click **Next** on the following dialog boxes. These screens will not be used.

- Mapping Users to Roles
- Mapping EJB Run As Roles to Users
- Binding Enterprise Beans to JNDI Names
- Mapping EJB References to Enterprise Beans
- Mapping Resource References to Resources
- Specifying the Default Datasource
- Specifying Data Sources for Individual CMP Beans

7 In the Selecting Virtual Hosts for Web Modules, select the WebSphere server instance you want to use, and then click **Next**.



8 In the Selecting Application Servers dialog box, select the WebSphere server instance you want to use, and then click **Next**.



9 On the dialog box displayed, click **Finish**.

Setting the Java heap size

You can configure how much memory is available for your application server instances. The following instructions assume you are only using one WebSphere instance. You will need to adjust the heap size accordingly if you are load balancing across several WebSphere instances.

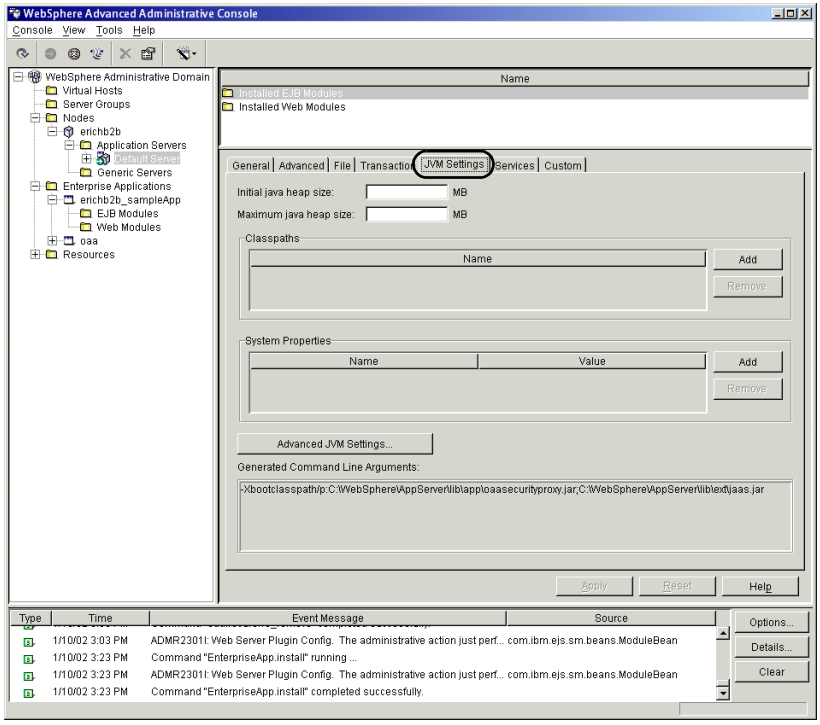
To set the Java heap size:

1 Verify that the WebSphere Admin Server has been started.

2 Open the WebSphere Advanced Administrator's Console (**Start** > **Programs** > **IBM WebSphere** > **Application Server** > **Administrator's Console**).

3 Click **Nodes** > <*System Name*> > **Application Servers** > <*Application server name*>.

The server settings page opens.



4 Click the JVM Settings tab.

5 Set the following JVM settings:

a **Initial java heap size**. Type `60`.

b **Maximum java heap size**. Type the value you want for heap memory. This setting should be at least 225 MB, but not more than 512 MB.

Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overallperformance. A setting of 256 MB should be sufficient for most systems.

Configuring a virtual directory for IBM HTTP Server

You must configure a virtual directory for Get-Services in your Web server. The following instructions assume that you are using WebSphere's built-in Web server – IBM HTTP Server. See your Web server documentation to determine how to create a virtual directory if you are using another Web server.

To configure IBM HTTP Server for Get-Services:

1. Stop the IBM HTTP Server.
2. Open the file **httpd.conf** in any text editor. By default this file is located at:

   <root>/usr/HTTPServer/conf
3. Add the following line to the end of the file:

   ```
   Alias /oaa/ "<root>/WebSphere/AppServer/installedApps/oaa.ear/portal.
   <version>.war/"
   ```

   For <*root*>, enter the root directory of the system.

   For <*version*>, enter the version number of the WAR file you installed.

   Important: The name you define for the virtual directory here must match the context root you defined in WebSphere.
4. Save the file.
5. Start the IBM HTTP Server.

Installing WebSphere Portal Server

You can configure Get-Services to display in a WebSphere Portal Server in one of two configurations:

- All Get-Services and WebSphere components running on a single system. See *Recommended WebSphere Portal Server configuration* on page 97.
- Get-Services components running on one system and WebSphere components running on another. See *Alternate WebSphere Portal Server configuration* on page 99.

Important: In either configuration, you must first install WebSphere Portal Server. See your WebSphere Portal Server documentation for details.

Recommended WebSphere Portal Server configuration

Use the following steps to configure Get-Services for the recommended WebSphere Portal Server configuration:

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 100.

Step 2 Generate a Get-Services WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Services WAR file* on page 101.

Step 3 Login to the Get-Services server and stop the WebSphere application server.

Step 4 Modify the archway.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the archway.xml file* on page 102.

Step 5 Modify the web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 102.

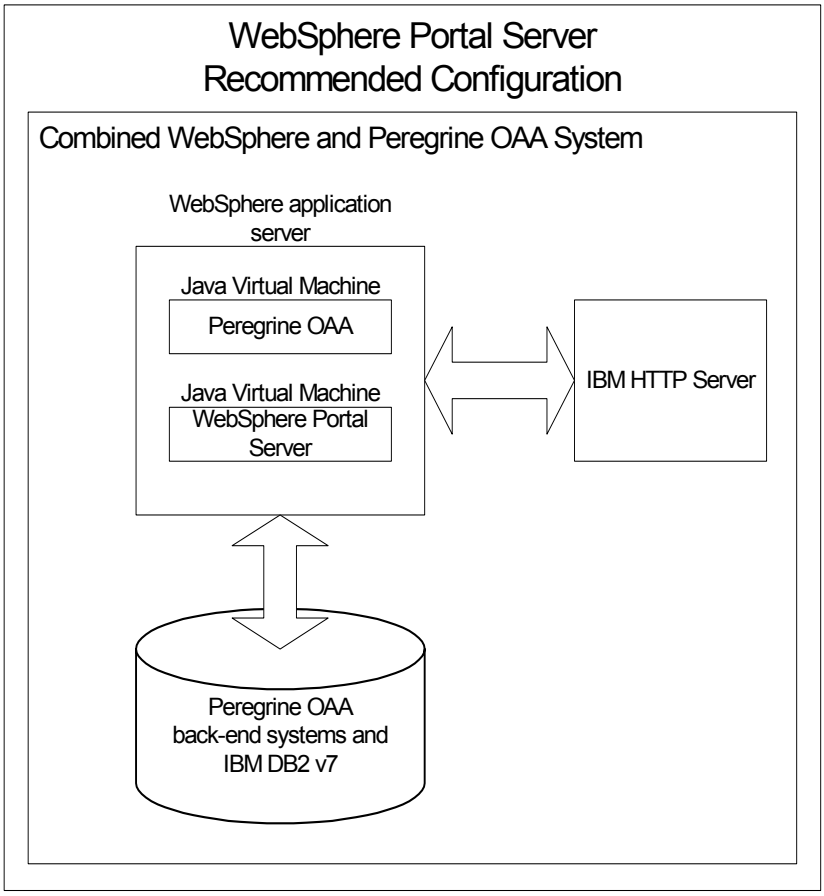
Step 6 Start the WebSphere application server.

Step 7 Deploy the Get-Services WAR file to WebSphere Portal Server. See *Deploying the Get-Services WAR file to WebSphere Portal Server* on page 103.

Step 8 Create places and pages in WebSphere Portal Server to display Get-Services portlets. See *Configuring WebSphere Portal Server places and pages* on page 104.

Step 9 Enable edit rights for Get-Services portlets. See *Enabling edit rights for Get-Services portlets* on page 104.

When complete, your installation will have the following configuration:



Alternate WebSphere Portal Server configuration

Use the following steps to configure Get-Services for the alternate WebSphere Portal Server configuration:

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 100.

Step 2 Generate a Get-Services WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Services WAR file* on page 101.

Step 3 Login to the Get-Services server and stop the WebSphere application server.

Step 4 Modify archway.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the archway.xml file* on page 102.

Step 5 Modify web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 102.

Step 6 Modify setDomain.js to call the SetDomain function. See *Modifying the setDomain.js file* on page 103.

Step 7 Start the WebSphere application server.

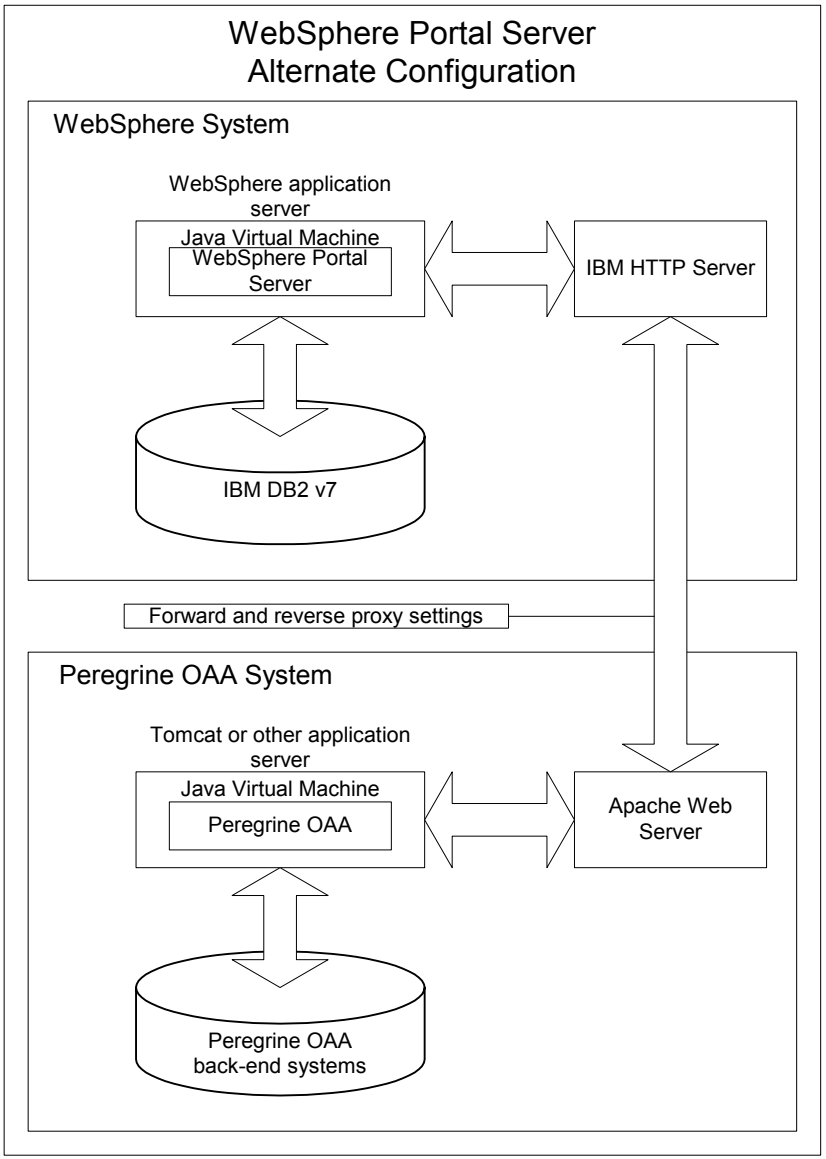
Step 8 Deploy the Get-Services WAR file to WebSphere Portal Server. See *Deploying the Get-Services WAR file to WebSphere Portal Server* on page 103.

Step 9 Create places and pages in WebSphere Portal Server to display Get-Services portlets. See *Configuring WebSphere Portal Server places and pages* on page 104.

Step 10 Enable edit rights for Get-Services portlets. See *Enabling edit rights for Get-Services portlets* on page 104.

Step 11 Modify IBM HTTP Server's httpd.conf file to add forward and reverse proxy URLs. See *Modifying httpd.conf for IBM HTTP Server* on page 105.

When complete, your installation will have the following configuration:



WebSphere Portal Server installation requirements

The recommended configuration of the WebSphere Portal Server requires the following items to be installed on the same server:

- WebSphere application server 4.0.2

- IBM HTTP Server 1.3.19
- IBM DB2 v7 database server
- WebSphere Portal Server
- A custom installation of Get-Services with WebSphere selected as the application server

The alternate configuration of the WebSphere Portal Server requires the following items be installed on a minimum of two servers:

- Server 1
  - WebSphere application server 4.0.2
  - IBM HTTP Server 1.3.19
  - IBM DB2 v7 database server
  - WebSphere Portal Server
- Server 2
  - Get-Services compatible application server
  - Web server
  - Back-end database for Get-Services
  - An installation of Get-Services

Generating a Get-Services WAR file

In order to display Get-Services in WebSphere Portal Server, you must first export the Get-Services portal components as a WAR file. You can then import this WAR file into WebSphere Portal Server, and choose the portal components you want to display as WebSphere Portal Server portlets.

To generate a Get-Services WAR file:

1. Login to the Get-Services administration page (admin.jsp).
2. Click **IBM WebSphere Portal Integration**.
3. Enter the following configuration information:
   a. **Source Path**. Enter the full path to the WebSphere.war in the Get-Services package folder. By default this folder is:

      <WebSphere>/oaa/packages

   b. **Destination Path**. Enter the full path and file name you want to use for the generated Get-Services WAR file.

c **Base URL**. Enter the full URL to the Get-Services deployment directory. By default this URL is:

http://<server>:<port>/oaa/servlet/basicauth

4 Click **Generate WAR file**.

Get-Services generates a new WAR file with the name and path specified in the Destination Path of step 3.

Modifying the archway.xml file

In order to login via WebSphere Portal Server, you configure Get-Services to use an alternate HTTP authentication method.

To modify the archway.xml file:

1 Using a text editor, open the archway.xml file located at:

<application server>/webapps/oaa/WEB-INF/default.

2 Edit the line containing:

```
<httpauthclass ...>HttpBasicAuthenticationManager</httpauthclass>
```

3 Change the value `HttpBasicAuthenticationManager` to `HttpAlternateAuthenticationManager`.

4 Save the file.

Modifying the web.xml file

You will need to enable the AuthController servlet to establish a proxy for HTTP basic authentication.

To modify the web.xml file:

1 Using a text editor, open the web.xml file located at:

<application server>/webapps/oaa/WEB-INF.

2 Search for the line containing:

```
<!-- Uncomment to add support for http basic authentication proxy
```

3 Move the ending comment tag --> from the end of the servlet definition to the comment at the beginning of the servlet definition.

The new servlet definition should appear as follows:

```
<!-- Uncomment to add support for http basic authentication proxy-->
  <servlet>
    <servlet-name>AuthController</servlet-name>
    <display-name>AuthController</display-name>
```

```
    <description>A controller (decorator) servlet that can be used to
enable configurable auth protection of any resource.</description>

<servlet-class>com.peregrine.oaa.archway.AuthControllerServlet</servl
et-class>
    <load-on-startup>2</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>AuthController</servlet-name>
    <url-pattern>/servlet/basicauth/*</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>AuthController</servlet-name>
    <url-pattern>/servlet/auth/*</url-pattern>
  </servlet-mapping>
```

4 Save the file.

Modifying the setDomain.js file

To use the alternate configuration of WebSphere Portal Server, you must enable the setDomain function.

Note: If you are setting up WebSphere Portal Server in the recommended configuration, you may skip these instructions.

To modify the setDomain.js file:

1 Login to the Get-Services server.

2 Stop your application server.

3 Using a text editor, open the **setDomain.js** file located at:

<application server>/webapps/oaa/js.

4 Add the following line to the end of the file:

```
setDomain();
```

5 Save the file.

Deploying the Get-Services WAR file to WebSphere Portal Server

After you deploy the Get-Services WAR file to WebSphere Portal Server, you can then configure the portlets you want to display, the display settings, and the access rights to each portlet.

See your WebSphere Portal Server documentation for detailed instructions.

To deploy the Get-Services WAR file:

1. Login to the WebSphere Portal as wpsadmin or another user with administrative rights.
2. Select **Portal Administration** from the Places menu.
3. Click **Portlets** > **Install Portlets**.
4. Click **Browse** and navigate to the Destination path you entered when you created the Get-Services WAR file.
5. Click Next to load the Get-Services WAR file.

   WebSphere Postal Server displays a list of portlets to be installed.
6. Click Install.

   WebSphere Portal Server installs the portlets and displays the message "Portlets successfully installed."

Configuring WebSphere Portal Server places and pages

You can deploy Get-Services portlets in any place or page that meet the following requirements.

Places

Your WebSphere Portal Server places must have the following characteristics:

- Supported markups must include HTML

Pages

Your WebSphere Portal Server pages must have the following characteristics:

- Supported markups must include HTML
- The page must be set to "allow all portlets that a user can access"
- All Get-Services portlets that you display in a page must grant "all authenticated users" the minimum edit permission.

Enabling edit rights for Get-Services portlets

WebSphere Portal Server users will need edit rights to the Get-Services portlets in order to add and customize them to their portal page.

To enable edit rights for Get-Services portlets:

1. Login to the WebSphere Portal as wpsadmin or another user with administrative rights.
2. Select **Portal Administration** from the Places menu.

3 Click **Security** > **Access Control List**.

4 Select the **Special groups** option and select **All authenticated users** from the select box.

5 From the Select the objects for the permissions select box, select **portlet applications**.

6 Select the Search on option, and then enter `Peregrine` in the Name contains field.

7 Click **Go**.

WebSphere Portal Server displays a list of portlets with Peregrine in the name.

8 In the Edit column, click **Select All** at the bottom of the table.

9 Click **Save**.

Users can now view and customize Get-Services portlets from the WebSphere Portal Server interface.

Modifying httpd.conf for IBM HTTP Server

In order to use the alternate configuration of WebSphere Portal Server, you will need to modify the **httpd.conf** file used by the IBM HTTP Server to add the forward and reverse proxy URLs to your remote instance of Get-Services.

Note: If you are setting up WebSphere Portal Server in the recommended configuration, you may skip these instructions.

To modify httpd.conf for IBM HTTP Server:

1 Login to the Get-Services server.

2 Stop your IBM HTTP Server.

3 Using a text editor, open the **httpd.conf** file located at:

<root>/usr/HTTPServer/conf

4 Add the following lines to the end of the file:

```
ProxyPass /<oaa root>/ http://<server>:<port>/
<oaa root>/servlet/basicauth/
ProxyPassReverse /<oaa root>/ http://<server>:<port>/
<oaa root>/servlet/basicauth/
```

For <*oaa root*>, enter the name of the oaa virtual directory used by IBM HTTP Server. By default, this virtual directory is `oaa`.

For <*server*>:<*port*>, enter the server name and communications port number where Get-Services is installed.

5 Save the file.

Configuring WebSphere Translation Server for Get-Services

You can configure Get-Services to use a WebSphere Translation Server to provide real-time translations of on-screen data.

To configure WebSphere Translation Server for Get-Services:

Step 1 Copy the file **wts.jar** to the Get-Services deployment folder. See *Copying wts.jar to the Get-Services deployment folder* on page 106.

Step 2 Configure Get-Services to use the WebSphere Translation Server. See *Configuring WebSphere Translation Server for Get-Services* on page 106.

Copying wts.jar to the Get-Services deployment folder

The following instructions describe where to find and copy the file **wts.jar**.

To copy wts.jar to the Get-Services deployment folder:

1 Stop your application server.

2 Browse to the location of your WebSphere Translation Server installation.

3 Copy the file wts.jar from this folder.

4 Paste the file wts.jar into the Get-Services deployment folder located at:

<Application server install>/WEB-INF/lib

5 Restart your application server.

Configuring Get-Services to use the WebSphere Translation Server

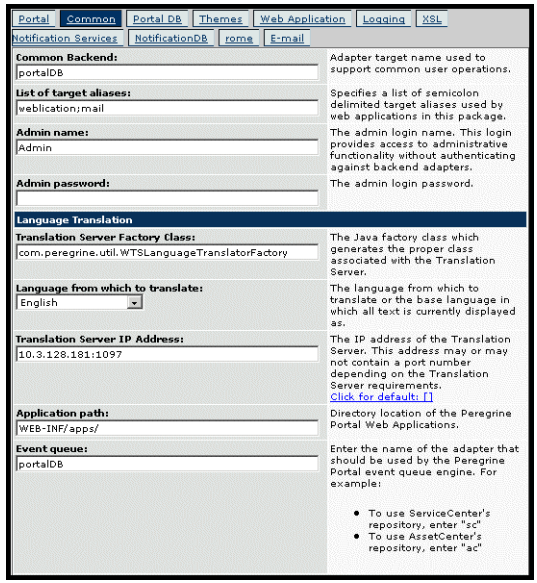
The following instructions describe how to configure Get-Services to use the WebSphere Translation Server.

To configure Get-Services to use the WebSphere Translation Server:

1 Login to the Get-Services admin page (**admin.jsp**).

2 Click **Settings** > **Common** tab.

The Admin Settings page opens.



3 Enter the following configuration settings:

   a **Translation Server Factory Class**: Enter the Java factory class for the Translation server. The default Java factory class is:

   com.peregrine.util.WTSLanguageTranslatorFactory

   b **Language from which to translate**: Enter the source language that you want translated. The default value is English.

   c **Translation Server IP Address**: Enter the IP address and communications port to the Translation Server. For example: 10.3.128.181:1097.

   d Application path: Enter the relative path to the application server applications directory. The default value is:

   WEB-INF/apps/

   e **Event queue**: Enter the adapter name to you want to use for the event queue engine. The default value is: PortalDB

4 Click **Save**.

   The Control Panel opens.

5 Click **Reset Server**.

Translating on-screen data with a Translation Server

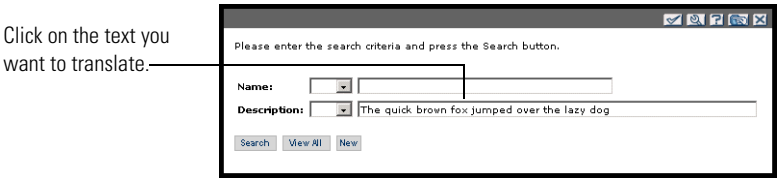
If you plan to store Get-Services data in a mixture of languages, you can configure Get-Services to send data to a Translation Server for real time translation. This interface will only translate data retrieved from the back-end database or manually typed into form inputs. If you need a translated user interface, you can purchase a Get-Services language pack directly from Peregrine Systems.

To translate on-screen data with a Translation Server:

1 Enable the translation server from the **Administration** > **Settings** page as described in *Configuring WebSphere Translation Server for Get-Services* on page 106.

The translate button appears in the upper right tool bar.

2 Click on the source data or form input you want to translate.



3 Click the translate button.

The Translation window opens.

4 Select the target language to which you want to translate from the drop down select box.

The translation of your selection displays in the Translation box.

WebLogic 6.1 SP3 or SP4

The following procedures configure WebLogic to run Get-Services on UNIX.

To configure WebLogic 6.1 SP3 or SP4:

Step 1 Stop both WebLogic and your Web server

Step 2 Edit the **startWebLogic.cmd** file to set the system password, memory settings, and start mode. See *Editing startWebLogic.cmd* on page 109.

Step 3 Run the Get-Services installer.

Step 4 Create a virtual directory for Get-Services in your Web server. See *Creating a virtual directory for Get-Services* on page 110.

Step 5 Restart WebLogic and your Web server.

Editing startWebLogic.cmd

To edit startWebLogic.cmd:

1. Open the file **startWebLogic.cmd** file in any text editor. By default the file is located at:

   /bea/wlserver6.1/config/<*mydomain*>**/**

2. Scroll to the following section of the script:

```
echo ************************************************
echo * To start WebLogic Server, use the password      *
echo * assigned to the system user. The system         *
echo * username and password must also be used to      *
echo * access the WebLogic Server console from a web   *
echo * browser.                                        *
echo ************************************************
@rem Set WLS_PW equal to your system password for no password prompt
server startup.
set WLS_PW=password
```

3. In the last line, change the word "**password**" to your WebLogic system password.
4. Search for the -mx parameter setting in the file. Change this setting to at least 225 MB, but not more than 512 MB.

Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems.

5 Set the STARTMODE variable to STARTMODE=false.

The first time you start WebLogic after the installation, you will need to start it in development mode for it to find the Web applications that have been deployed.

6 Save the file.

Creating a virtual directory for Get-Services

To run Get-Services, you need to create a virtual directory in your Web server that maps to your WebLogic deployment folder. The typical installation creates a virtual directory called oaa, but you may specify a different virtual directory name.

Requirements for Get-Services virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Directory access rights	anonymous
Map to physical path	<WebLogic>/applications/oaa
Remove "allow anonymous access" to	default.asp, login.asp, e_login_main_start.asp
Set security access to only allow "System" and "Authenticated Users" to	default.asp, login.asp, e_login_main_start.asp

For <*oaa*>, enter the name of the virtual directory you want to use for Get-Services. Whatever name you enter here you will need to replicate in your application server configuration.

For <*WebLogic*>, enter the path to your WebLogic installation. By default the path is:

/bea/wlserver6.1/config/<*mydomain*>

JRun 3.1

The following procedures configure JRun to run Get-Services on UNIX.

To configure JRun 3.1:

Step 1 Install a Java run-time environment. The Get-Services installer includes the Java 2 SDK Standard Edition v1.3.1_05. However, you can also use JRE 1.3.1 if you already have it installed. See *Custom Installation Components* on page 128.

Step 2 Install JRun from the Allaire Web site to the root of your hard drive.

Step 3 Apply the latest JRun update. See *Applying the latest JRun update* on page 111.

Step 4 Deploy the Portal WAR file to JRun to create the necessary folder structure for Get-Services. See *Deploying the Portal WAR file to JRun* on page 112.

Step 5 Run the Get-Services installer.

Step 6 Move **js.jar** to the Java development kit ext folder. See *Moving js.jar to the Java development kit* on page 115.

Step 7 Run the JRun Connector Wizard to establish a connection between JRun and your Web server. See *Running the JRun Connector Wizard* on page 115.

Step 8 Configure your JRun Java settings. See *Configuring Java settings* on page 115.

Step 9 Define any library path environmental variables to your back-end databases. See *Defining library path environment variables* on page 118.

Step 10 Create a virtual directory for Get-Services in your Web server. See *Creating a virtual directory for Get-Services* on page 119.

Step 11 Restart JRun and your Web server.

Applying the latest JRun update

Before you install Get-Services, you must apply the latest Jrun 3.1 update.

To install the latest JRun update:

1. Browse to the following URL:

   http://www.macromedia.com/support/jrun/updates/3/updates_31.html

2. Click the link for the **JRun edition (Enterprise, Advanced, or Professional) and operating system of your server.**

3 Follow the installation instructions provided.

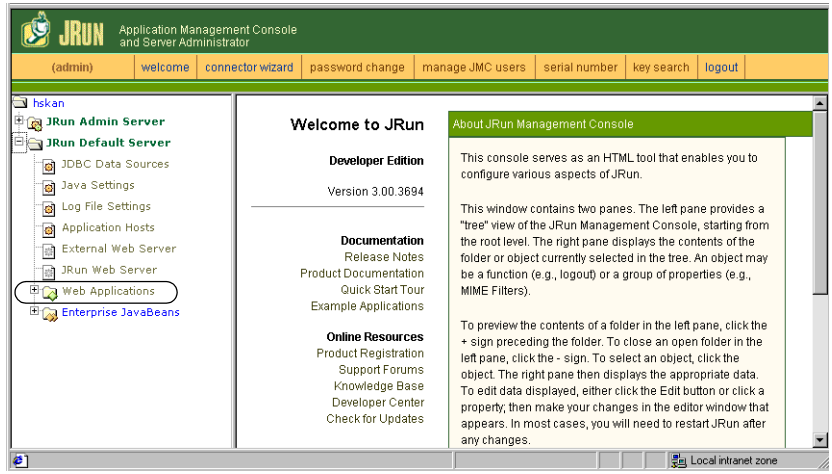
Deploying the Portal WAR file to JRun

The Portal WAR file creates the folder structure necessary to deploy Get-Services in your application server. After you have deployed this file to WebLogic you will be ready to run the Get-Services installer.

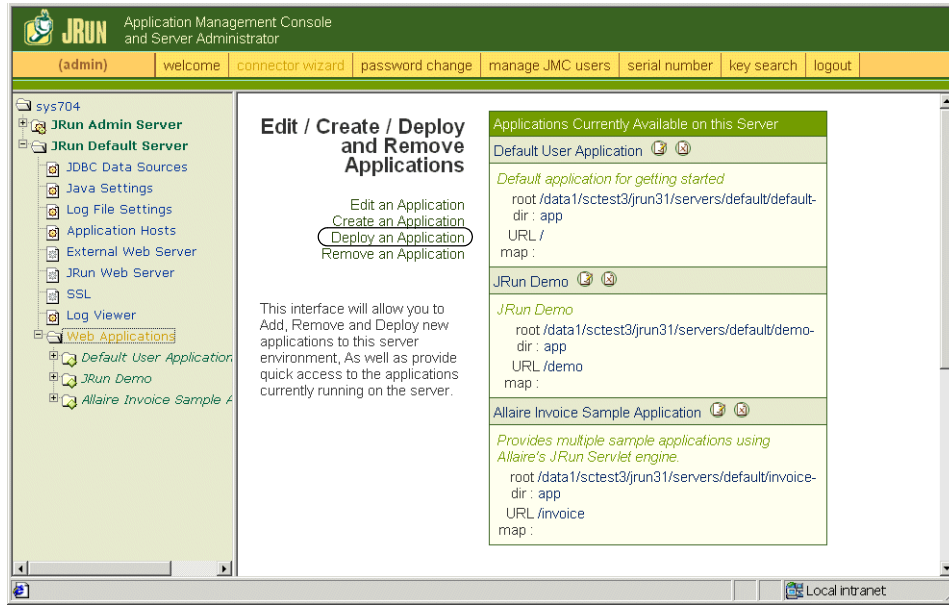
To deploy the Get-Services Portal WAR file to JRun:

1 Open the JRun Management Console and log in.

2 Select JRun Default Server > Web Applications.



The Edit / Create / Deploy and Remove Applications page opens.



3 Click the **Deploy an Application** link.

4 In the page that opens, fill out the fields as follows:

- Servlet War File or Directory:

  Browse to **<CD Rom Drive>/portal<version #>.war**.

For <version>, select the most current version.

Select this file, and then click **Accept**.

- JRun Server Name:

  Select **JRun Default Server**.

- Application Name:

  Type oaa.

- Application URL:

  Type /oaa.

- Application Deploy Directory:

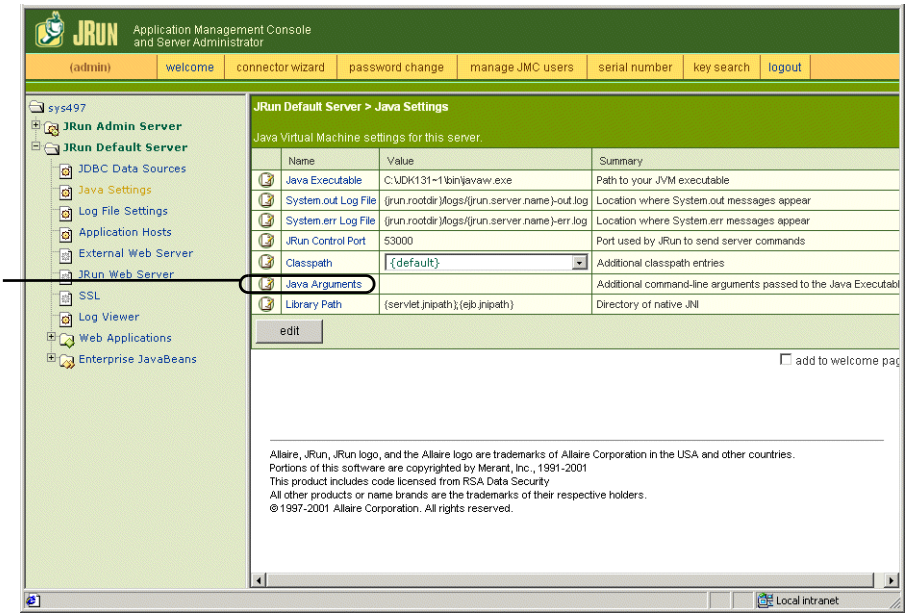
  JRun generates this directory. Make a note of this path. You will need this information later in the procedure.

5 Click **deploy**.

A message that OAA has been successfully deployed appears.

6 On the Java Settings page, click **Java Arguments**.

Click **Java Arguments**.



7 In the edit window that opens, use the format example to enter an -Xmx value. This defines the maximum amount of heap memory allocated for your system. It is recommended that you set this value to at least 225 MB, but not more than 512 MB.

Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems. Applications using Persistence may require a higher setting.

Moving js.jar to the Java development kit

JRun requires an updated version of js.jar in the Java development kit.

To move the js.jar to the Java development kit:

1. Stop JRun.
2. Locate js.jar. By default this file is installed at:

   <JRun>/servers/default/oaa/WEB-INF/lib
3. Cut and paste the file to the following path:

   /usr/local/peregrine/common/jdk1.3/jre/lib/ext
4. Restart JRun.

Running the JRun Connector Wizard

The JRun Connector Wizard establishes a connection between JRun and your Web server.

To run the JRun Connector Wizard:

1. Login to the JRun Management Console.
2. Click Connector Wizard.
3. Select the JRun Default Server as the JRun Server Name.
4. Select your Web server from the drop down list box.
5. If your Web server uses a different IP address than your JRun server, enter the IP address of your JRun server in JRun Server IP Address.
6. Confirm that the JRun Server Connector Port is not in conflict with another communications port used on this server.
7. Enter the path to the Scripts Directory.
8. Click Done.

Configuring Java settings

After you have installed Get-Services, you must configure the Java settings that JRun will use to run the Web application.

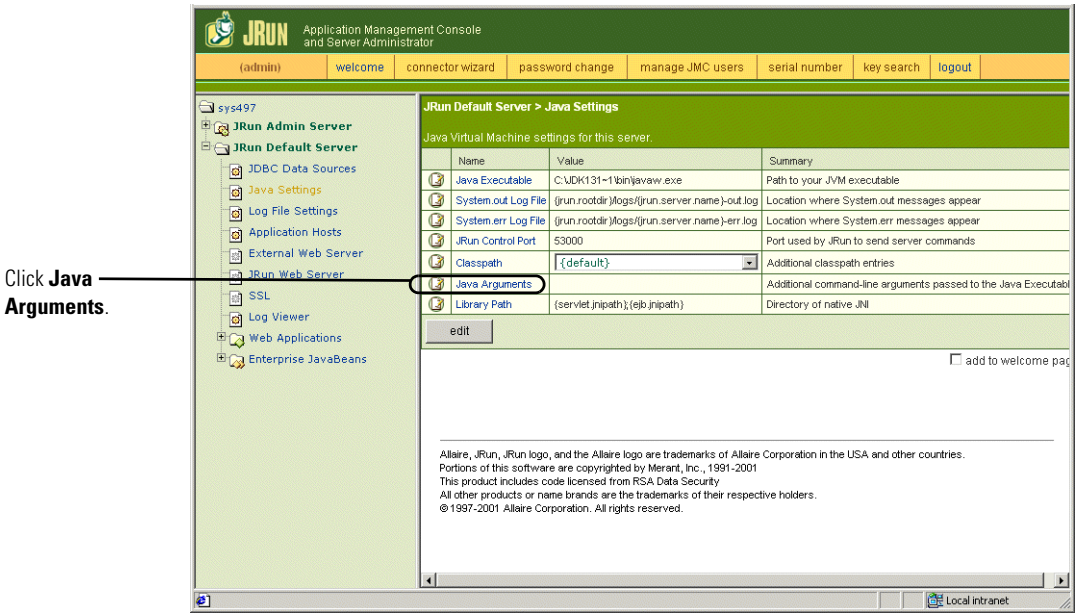
To configure Java settings:

1 Login to the JRun Management Console.

2 Click JRun Default Server > Java Settings.

The Java Settings page opens.

3 Click **Java Arguments**.

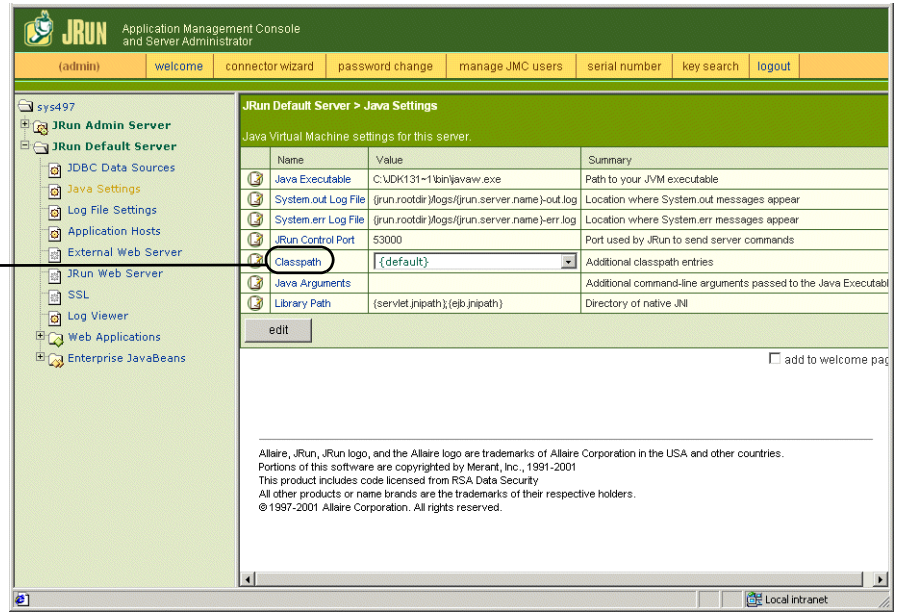


The Edit Window opens.

4 Enter an -Xmx value to define the maximum amount of heap memory allocated for your system. It is recommended that you set this value to at least 225 MB, but not more than 512 MB.

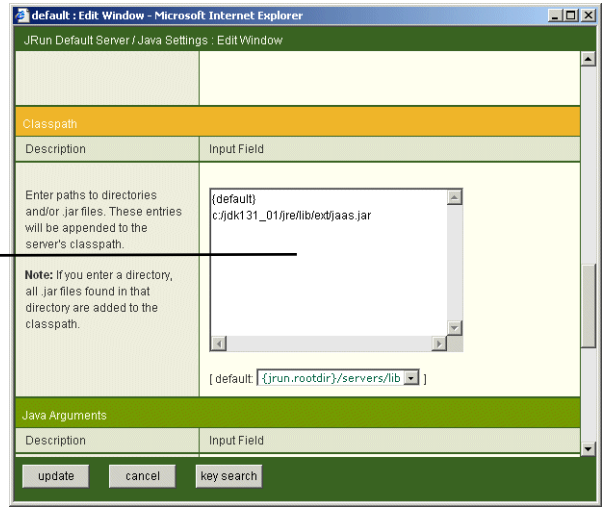
Note: Make sure that the setting for maximum heap size is less than the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance. A setting of 256 MB should be sufficient for most systems. Applications using Persistence may require a higher setting.

5 On the Java Settings page, click **Classpath**.



Click **ClassPath**.

The Edit Window opens.



Type the path to the JAR files here.

6 Enter the following classpaths:

- Java Development Kit ext folder. For example:

  /usr/peregrine/Common/jdk1.3.1_05/jre/lib/ext

7 On the Java Settings page, click **Java Executable**.

8 Verify that path to your Java Development kit matches the path listed in the Classpath setting. For example:

/usr/peregrine/Common/jdk1.3.1_05/bin/javaw.exe

9 Click **update**.

Defining library path environment variables

On UNIX-based systems, you must define the library paths to your back-end databases.

1 Open the JRun Management Console and log in.

2 On the menu at the left, select **JRun Default Server** > **Java Settings**.

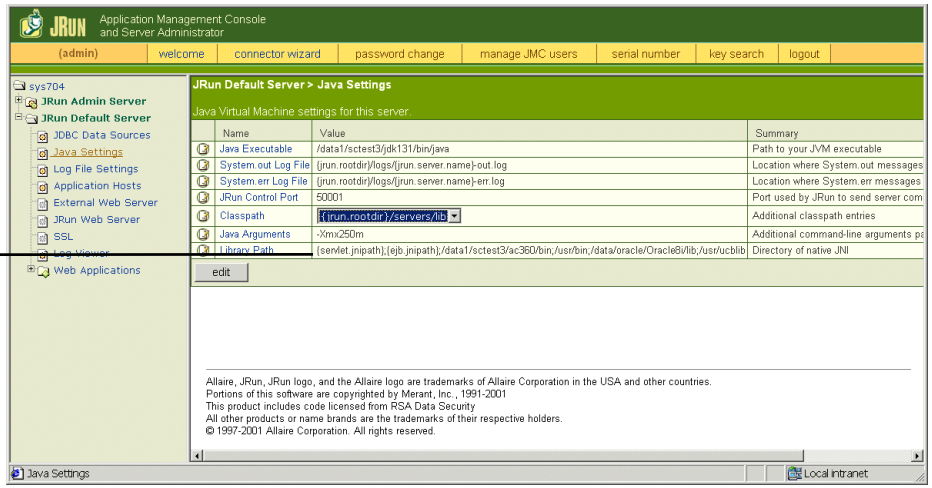
3 Click **Library Path**.

4 Add the following library paths if needed:

- If you are running Get-Services on Solaris, add:
  - /usr/bin
  - /usr/ucblib

5 Click **update**.

JRun displays the updated library paths.



The updated path appears here.

6 Logout of the Management Console.

Creating a virtual directory for Get-Services

To run Get-Services, you need to create a virtual directory in your Web server that maps to your JRun deployment folder. The typical installation creates a virtual directory called **oaa**, but you may specify a different virtual directory name.

Requirements for Get-Services virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Directory access rights	anonymous
Map to physical path	**<JRun>/oaa**
Remove "allow anonymous access" to	**default.asp, login.asp, e_login_main_start.asp**
Set security access to only allow "System" and "Authenticated Users" to	**default.asp, login.asp, e_login_main_start.asp**

For *<oaa>*, enter the name of the virtual directory you want to use for Get-Services. Whatever name you enter here you will need to replicate in your application server configuration.

For *<JRun>*, enter the path to your JRun installation. The recommended installation path is:

/JRun/servers/default

Typical Installation Option

A typical installation of Get-Services installs the most commonly used components of the product and saves application files and data in default destination directories. Most users choose Typical installation.

Typical Installation Components

Following is a brief description of the components that are automatically installed with a Typical installation of Get-Services:

Applications and File Locations

Get-Services Component	Default Installation Directory
Apache Web Server	/usr/local/peregrine/common/Apache2
Tomcat Application Server	/usr/local/peregrine/common/Tomcat4
Java Development Kit	/usr/local/peregrine/common/jdk1.3.1
OAA Platform and Get-Services	/usr/local/peregrine/oaa

Communications Ports

Get-Services uses the following communications ports in a typical installation. After installation, you can configure Get-Services to use one or more of the alternate communications ports if your local network already uses these communications ports.

Default Port	Component used by	Alternate Port
80	Apache Web Server	8081
8005	Tomcat application server administration	8015
8009	Tomcat application server worker file	8019
8011	Tomcat application server worker file for load balancing (optional).	8021

Default Port	Component used by	Alternate Port
8013	Tomcat application server worker file for load balancing (optional).	8023
8015	Tomcat application server worker file for load balancing (optional).	8025

Note: To change settings for these components or to use or install different components, use the Custom installation option for Get-Services.

This completes the installation. If you have not already done so, you will now need to configure your system to connect to the back-end database you are using. This is done on the Settings page of the Admin module.

Typical Installation Procedures

This section explains how to install Get-Services with a Tomcat application server and an Apache web server on a AIX, Linux, or Solaris operating system.

To perform a typical installation of Get-Services on UNIX:

1 Login to your server.

Important: On servers running AIX, you must login with an account that has root privileges.

2 Insert the Get-Services installation CD into your computer's CD ROM drive. Your computer should automatically launch the installation program.

If the installation program does not automatically start, mount your CD ROM drive. For example:

```
mount /cdrom
```

Change directories to your CD ROM. For example:

```
cd /cdrom
```

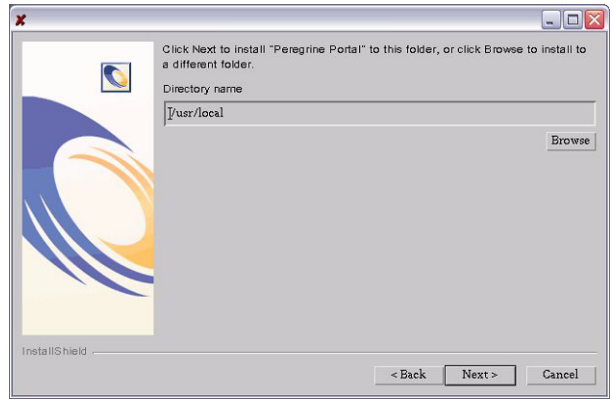
Enter the installer script specific for your operating system:

Operating system	Shell script to run
AIX 5.1	`./setupaix`
Red Hat Linux 7.3	`./setuplinux`
Solaris 2.7	`./setupsolaris`
Solaris 2.8	`./setupsolaris`

The installer welcome page opens.

3 Click **Next** to continue to the next page of the wizard.

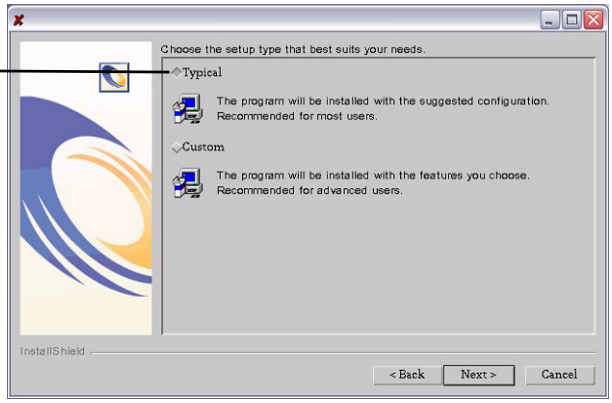
The installation location page opens.



4 Click **Browse** to change the default installation location of /usr/local.

5 Click **Next** to continue to the next page of the wizard.

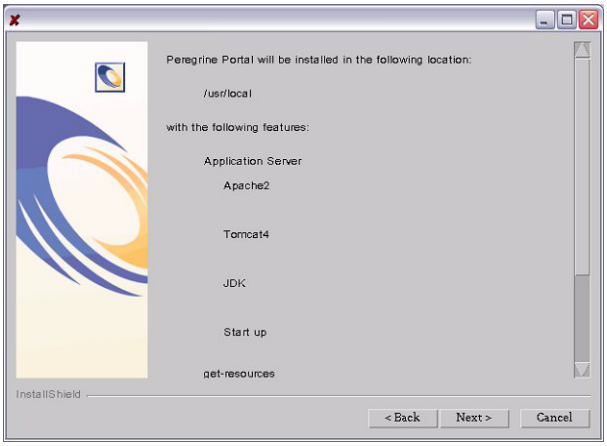
The setup type page opens.



6 Select **Typical**.

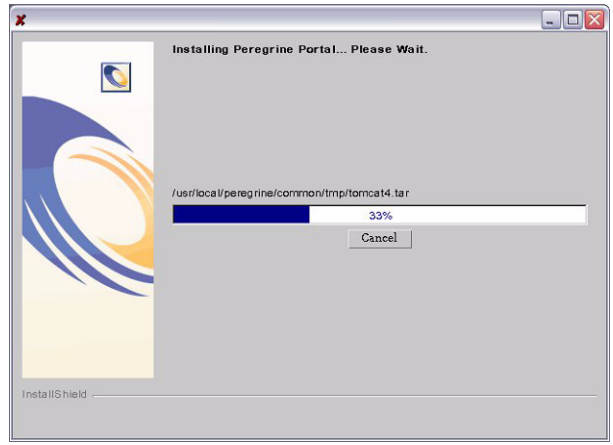
7 Click **Next** to continue to the next page of the wizard.

The review components page opens.

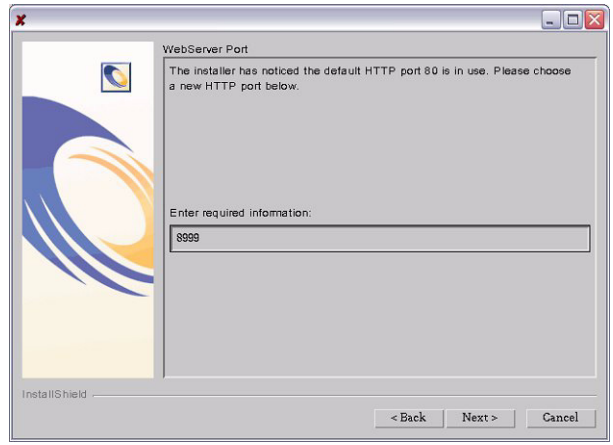


8 Click **Next** to continue to start installing Get-Services components.

The installation progress page opens.



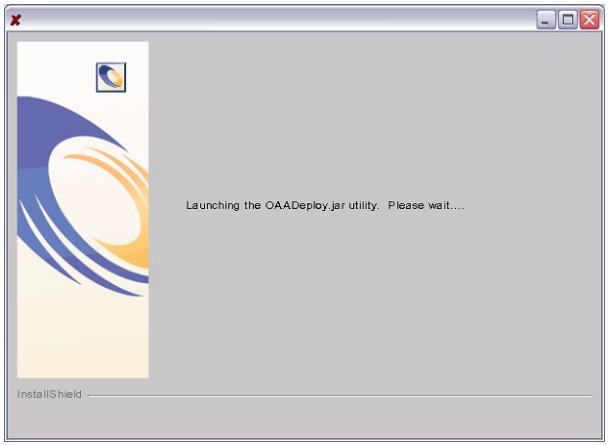
After installation is complete, the installer verifies the availability of port 80 for the Apache Web server. If the installer finds a port conflict on port 80 the Web server port page opens.



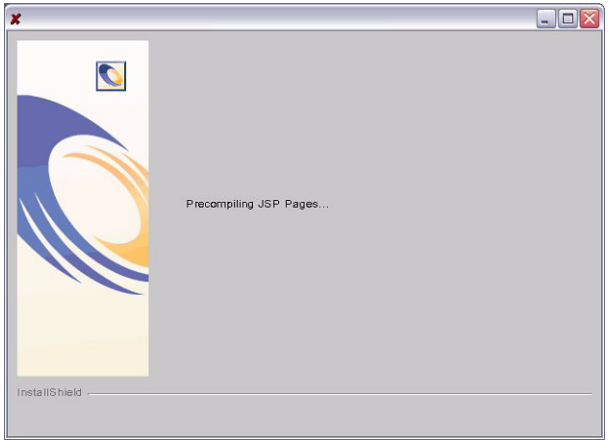
9 If required, enter the new Web server communications port.

10 Click **Next** to start deploying Get-Services components.

The Get-Services deployment utility page opens.

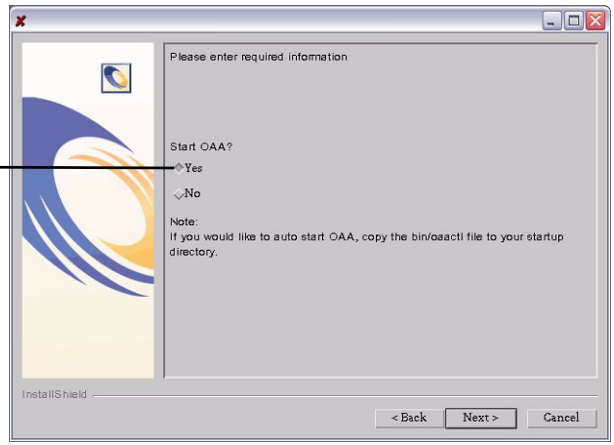


During deployment, the installer precompiles the Java server pages for use by Tomcat.



After the installer has successfully deployed Get-Services, the start OAA page opens.

Select **Yes** to start Get-Services immediately.



11 Click **Yes** to start Get-Services immediately or select **No** to manually start Get-Services after installation is complete.

If you want Get-Services start every time the server is started, then copy the file oaactl into your startup directory. By default this file is located at:

/usr/local/peregrine/bin/

You have now completed installing Get-Services.

Custom Installation Option

The following section describes how to perform a custom installation of Get-Services on a Unix operating system server, including overview steps for a Development and Production environment.

Custom Installation Components

Following is a brief description of the components that are available for a custom installation of Get-Services:

Application options

Get-Services Component	Options
Web Server	■ Apache 2.0.43 ■ IBM HTTP Server 1.3.19 ■ Microsoft IIS 5.0
Application Server	■ Tomcat 4.1.12 ■ WebSphere 4.02 ■ WebLogic 6.1 SP3 ■ JRun 3.1
Java Development Kit	Java 2 SDK
OAA Platform and Get-Services	Change Management

Communications Ports

The communications ports used by a custom installation of Get-Services depend upon the application components that you select. Refer to your Web and application server documentation to determine what communications port they require. After installation, you can configure Get-Services to use alternate communications ports if your local network already uses particular communications ports.

Get-Services on servers running Oracle 9.2.0.1

If you are running Get-Services on a server using Oracle 9.2.0.1 you may experience a port conflict over communications ports 8009 and 8080. Consult your Web and application server documentation to see if they use either of these two ports.

If you are using Tomcat as your application server, then by default, there will be a port conflict over port 8009. It is recommended that you change Tomcat to use a different communications port on servers running Oracle 9.2.0.1.

Custom Installation Procedures

To perform a custom installation of Get-Services on UNIX:

1 Login to your server.

Important: On servers running AIX, you must login with an account that has root privileges.

2 Insert the Get-Services installation CD into your computer's CD ROM drive. Your computer should automatically launch the installation program.

If the installation program does not automatically start, mount your CD ROM drive. For example:

```
mount /cdrom
```

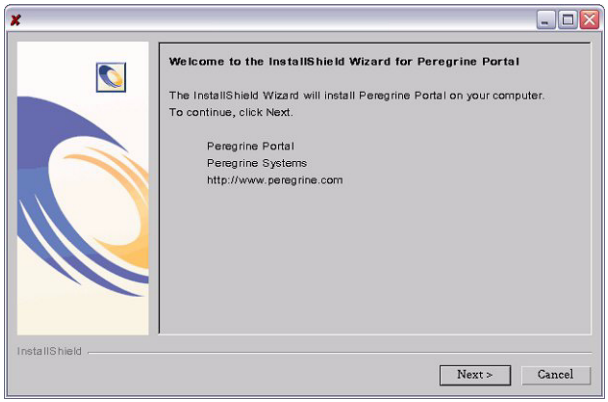
Change directories to your CD ROM. For example:

```
cd /cdrom
```

Enter the installer script specific for your operating system:

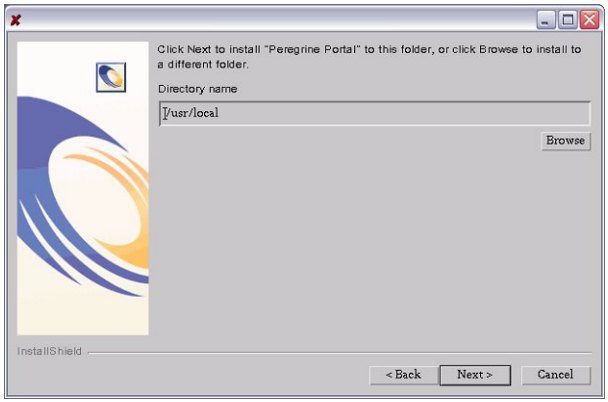
Operating system	Shell script to run
AIX 5.1	`./setupaix`
Red Hat Linux 7.3	`./setuplinux`
Solaris 2.7	`./setupsolaris`
Solaris 2.8	`./setupsolaris`

The installer welcome page opens.



3 Click **Next** to continue to the next page of the wizard.

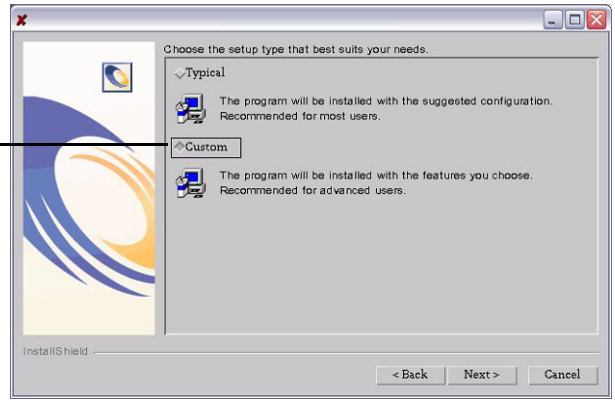
The installation location page opens.



4 Click **Browse** to change the default installation location of /usr/local.

5 Click **Next** to continue to the next page of the wizard.

The setup type page opens.

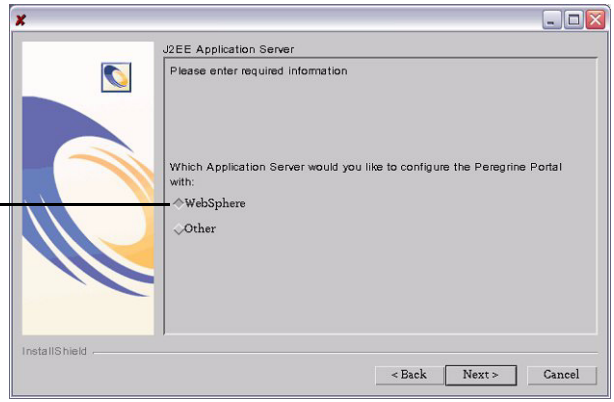


Click **Custom**.

6 Select **Custom**.

7 Click **Next** to continue to the next page of the wizard.

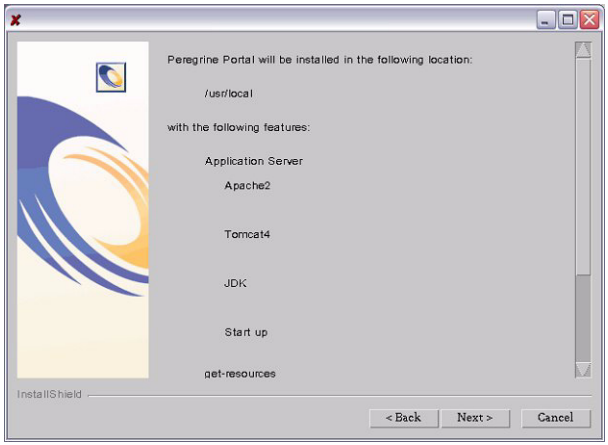
The select J2EE application server page opens.



Select your application server from the list provided.

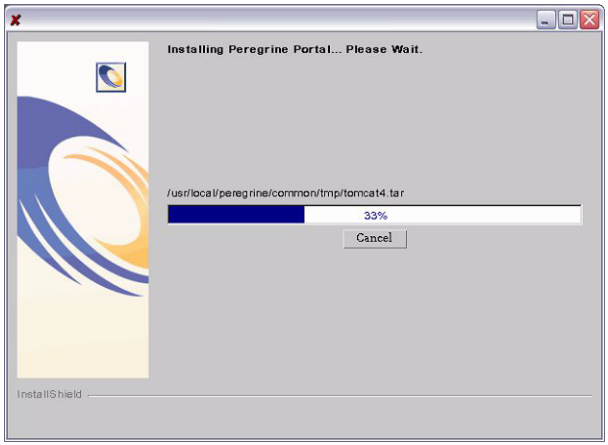
8 Click **WebSphere** to configure a WebSphere application server, or click **Other** to configure any other application server. Click **Next** to continue to the next page of the wizard.

The review components page opens.



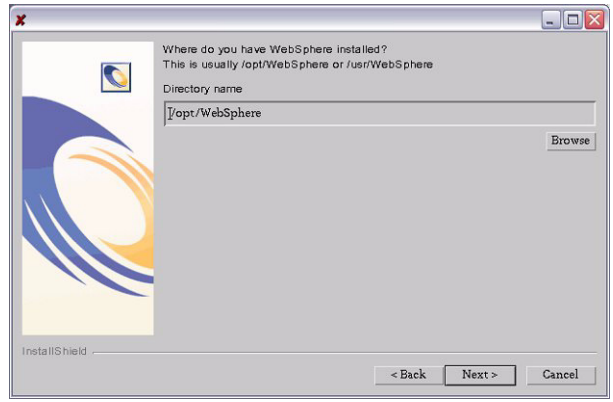
9 Click **Next** to continue to start installing Get-Services components.

The installation progress page opens.



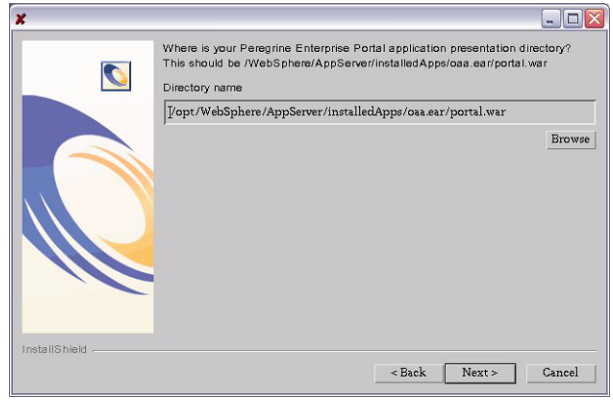
If you have selected to configure a WebSphere application server, you will see the screens in step a through step e.

The WebSphere installation location page opens.



a Click **Browse** to locate the directory where you installed WebSphere. Click **Next** to continue.

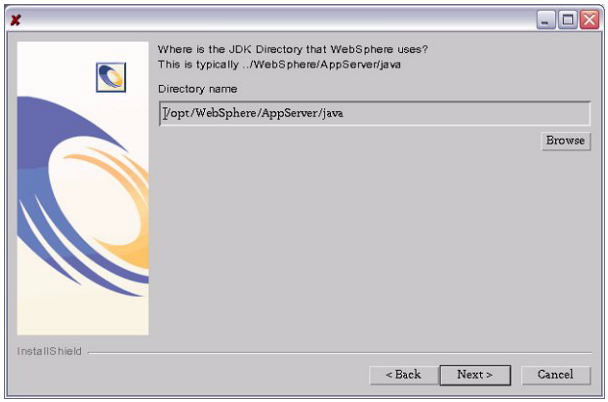
The presentation directory location page opens.



b Click **Browse** to locate the directory where you deployed the portal.war file. Click **Next** to continue.

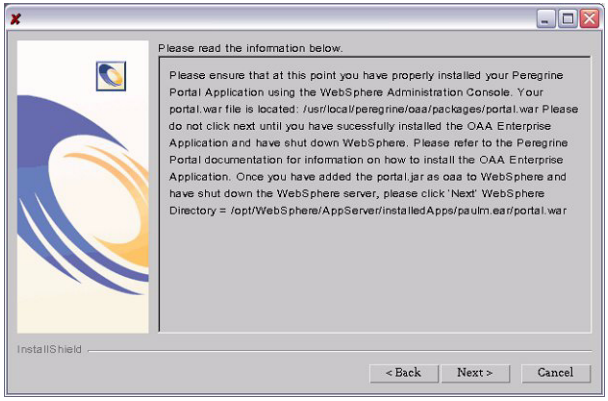
WebSphere automatically created this directory when you deployed the Get-Services portal.war as an enterprise application. See *WebSphere 4.0.2* on page 92 for more information on deploying a WAR file.

The WebSphere JDK installation location page opens.



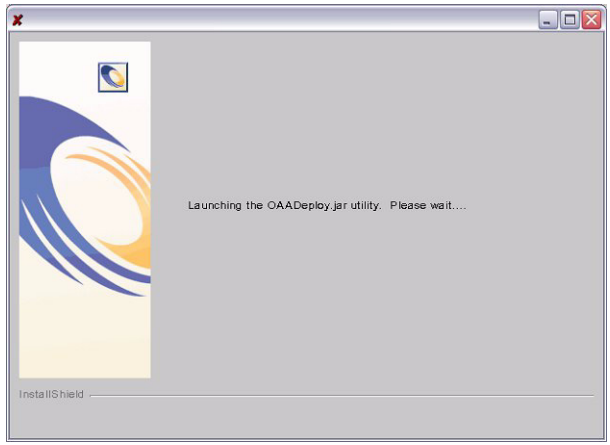
c Click **Browse** to locate the directory where you installed the Java development kit used by WebSphere. Click **Next** to continue.

The deployment confirmation page opens.

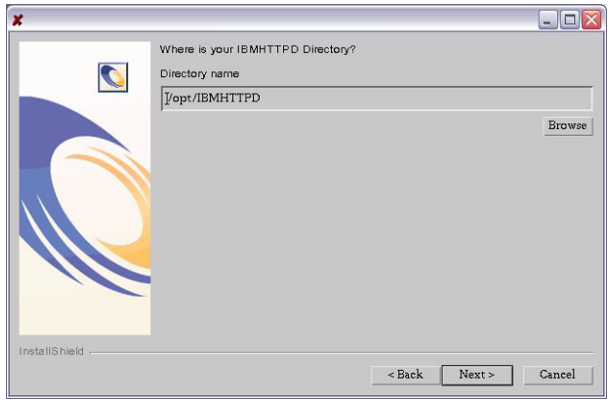


d After you have successfully deployed **portal.war** in WebSphere, click **Next** to continue.

The Get-Services deployment utility page opens.



The IBM HTTP Server location page opens.

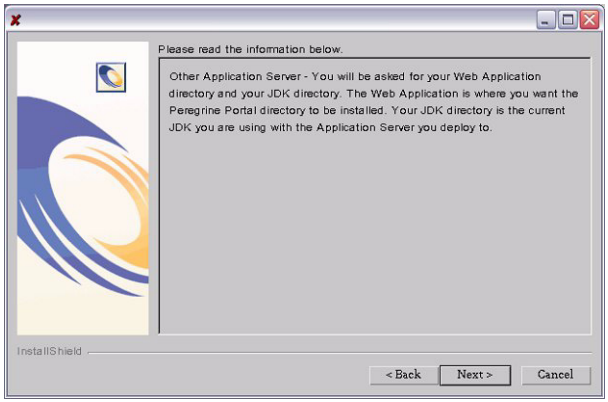


e Click Browse to locate where you installed the IBM HTTP Server. Click **Next** to continue.

The Get-Services installer will automatically configure a Web server virtual directory called **oaa**. If you want to define a different Web server virtual directory, see *WebSphere 4.0.2* on page 92 for a list requirements.

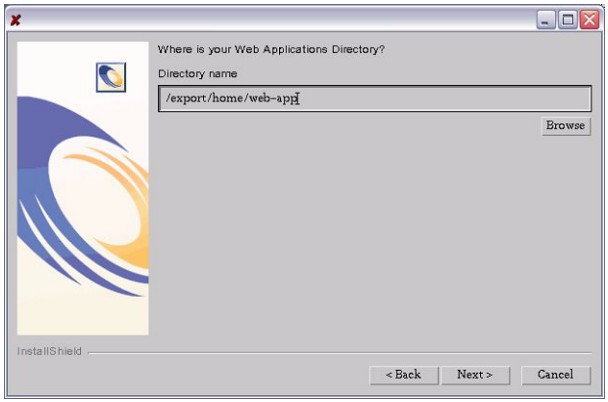
If you have selected to configure another application server, you will see the screens in step f through step h.

The other application server information page opens.



f Click **Next** to configure your application server.

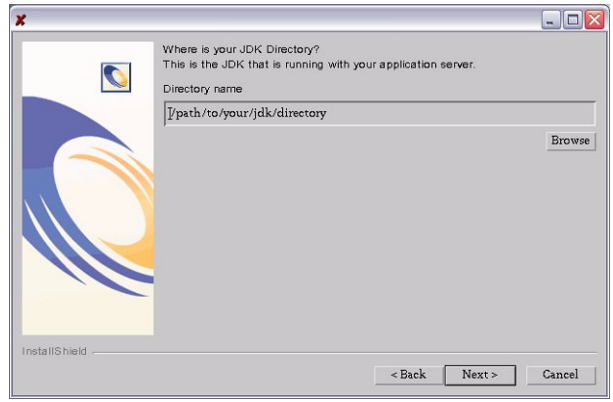
The Web applications directory page opens.



g Click **Browse** to locate the directory where you deployed the portal.war file. Click Next to continue.

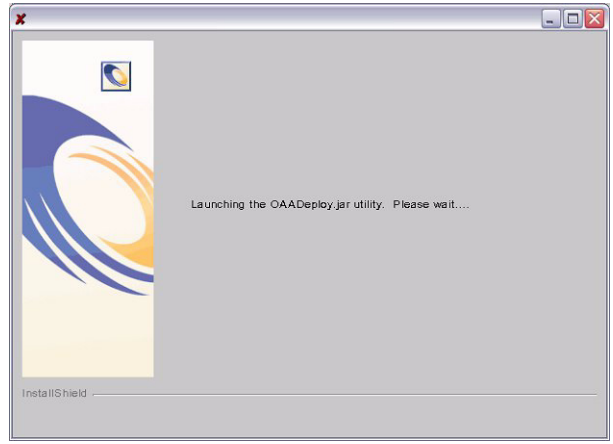
For more information on deploying a WAR to application servers, see *Configuring alternate application servers* on page 89.

The application server JDK installation location page opens.



h Click **Browse** to locate the directory where you installed the Java development kit used by your application server. Click **Next** to continue.

The Get-Services deployment utility page opens.



10 Click Finish to close the installer.

11 Enter the library path environment variables required by your operating system and application server.

Operating system	Environment variable	Add these path values
AIX	LIBPATH	■ /<App server>/WEB-INF/lib/AIX ■ /<App server>/WEB-INF/lib/AIX/ServiceCenter4
Red Hat Linux 7.3	LD_LIBRARY_PATH	■ /<App server>/WEB-INF/lib/Linux ■ /<App server>/WEB-INF/lib/Linux/ServiceCenter4
Solaris 2.7 Solaris 2.8	LD_LIBRARY_PATH	■ /<App server>/WEB-INF/lib/SunOS ■ /<App server>/WEB-INF/lib/SunOS/ServiceCenter4 ■ /usr/ucblib

For <*App server*>, enter the path to your application server's context root including where you installed the **oaa** application. Use the following table to determine your application server's context root.

Application server	Context root
WebSphere	/WebSphere/AppServer/installedApps/<application>.ear/portal.4.0.0.<x>/oaa
WebLogic	/bea/wlserver6.1/config/<mydomain>/applications/oaa
JRun	/JRun/servers/default/oaa

For <*application*>, enter the folder created for Get-Services.

For <*x*>, enter the version number of your **portal** WAR file.

For <*my domain*>, enter the WebLogic domain you created.

Configuring the WebSphere environment on Linux and Solaris

If you are running on a Linux or Solaris server, you can configure your WebSphere environment using the WebSphere Advanced Management Console.

To configure the WebSphere environment on Linux or Solaris:

1. Verify that the WebSphere Admin Server has been started.
2. Open the WebSphere Advanced Administrator's Console:

   /WebSphere/AppServer/bin/adminclient.sh
3. On the left side of the console, click **Nodes** > <*Server Name*> > **Application Servers**.
4. Click on the application server where you installed Get-Services.
5. Click the **General** tab.
6. Click **Environment**.
7. Click **Add**.
8. Enter the following environment details:
   a. **Name**. Type in LD_LIBRARY_PATH.
   b. **Value**. Type in the paths values for your operating system. This value consists of the context root and the relative path to your LIB directories.
9. Click **OK**.
10. Click **Apply**.
11. Close the WebSphere Advanced Administrator's Console.

Configuring the WebSphere startupServer.sh on AIX

If you are running on an AIX server, you must configure your WebSphere environment by editing the startupServer.sh script.

To configure the WebSphere environment on AIX:

1. Open startupServer.sh in any text editor.
2. Add an entry for LIBPATH and set it to the path values for AIX.

   Example:

```
#!/bin/sh
LIBPATH=/usr/lib:/WebSphere/AppServer/installedApps/answer.ear/portal
.4.0.0.55.war/WEB-INF/lib/AIX:/WebSphere/AppServer/installedApps/answ
er.ear/portal.4.0.0.55.war/WEB-INF/lib/AIX/ServiceCenter4
export LIBPATH
```

3. Save the file.

Uninstall—AIX, Linux, or Solaris

Use the following instructions to uninstall Get-Services.

To uninstall Get-Services from AIX, Linux, or Solaris:

1. Open a command prompt.
2. Change directories to:

   <root>/usr/peregrine/_uninst
3. Enter the following command to uninstall Get-Services:

   ```
   ./uninstall.bin
   ```
4. Follow the on-screen instructions to complete the uninstall.

4 Load Balancing

CHAPTER

This chapter covers the following topics:

Load balancing application servers

A server running a Web application such as Peregrine's Get-Services or Get-Resources consumes approximately 256 MB of memory per application server instance. You should not set the maximum heap size of the JVM in excess of the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance.

Unlike other Adapters, the AssetCenter and ServiceCenter Adapters each create a single connection to the respective back end. Therefore, the memory consumed on the AssetCenter database server is the same as that consumed by a single client connection. The memory consumed on the ServiceCenter server is also the same as that of a single ServiceCenter client process.

Note that memory usage does not increase significantly per session, because the architecture is based on the sharing of a set of resources and database connections among all sessions handled by the same application server instance. The small amount of memory consumed for session-specific information is released as the users log off or as their sessions expire. Note that server sessions do not expire unless the browser is closed or the user navigates to a different domain.

Because ServiceCenter and AssetCenter adapters maintain a single connection to the back end, adding extra application server instances brings the added benefit of concurrent access to the back-end data store.

The need for extra application server instances and therefore JVMs is directly related to three variables:

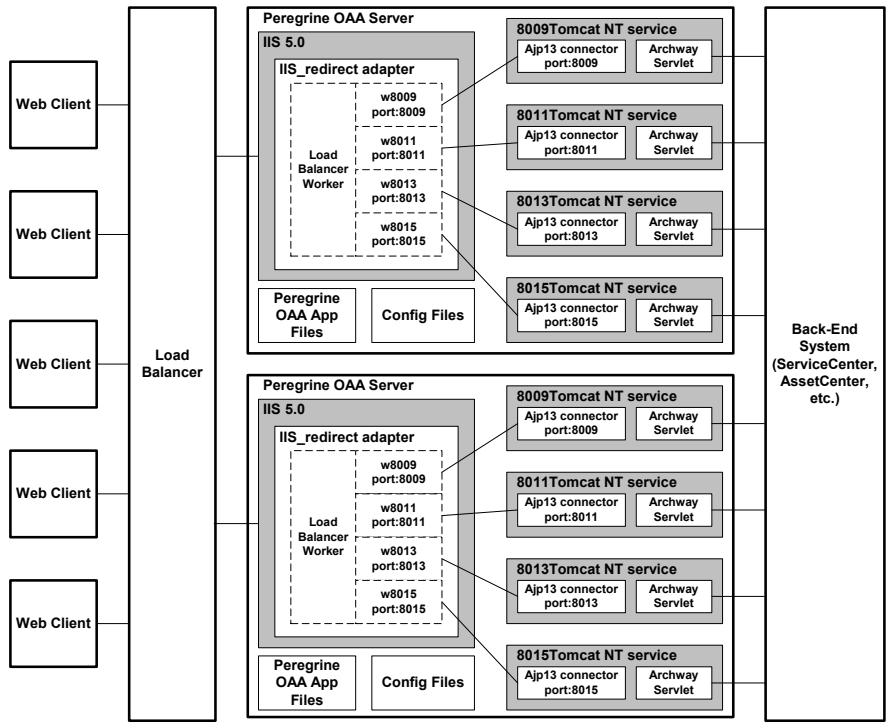
- The number of concurrent users.
- The processing power of the machine hosting the Get-Services Web server.
- The number of processors on the machine.

Each deployment may make different demands of the software and hardware, but, in any case, optimal back-end throughput for ServiceCenter and AssetCenter is achieved with the maximum number of application server instances that the server can handle without degraded performance due to lack of CPU headroom, file system swapping, and context switching.

Cache synchronization with Symmetric MultiProcessing (SMP) servers can, in most cases, be ignored as a performance tuning factor except in the case of the extremely large-scale systems.

To serve as a control guideline, low-end processors, such as a Pentium 450, should be capable of producing acceptable load handling for around 100 concurrent sessions on a single application server process. A dual Pentium 1000 with 2 gigabytes of RAM (a common data center configuration) should be capable of handling 400+ concurrent sessions using multiple application server instances. When using adapters capable of pooling, for example, the JDBCAdapter or BizDocAdapter, performance beyond the 400-concurrent-user benchmark can be achieved.

The following diagram illustrates the architecture of multiple JVMs:



Creating multiple instances of Tomcat for Apache

Multiple instances of Tomcat are installed as services. Although this is not required, it improves performance, makes the instances easier to manage, and provides extra functionality, including restarting the service if it fails or if the machine on which the instances are installed needs to be restarted.

For systems using IIS, see *Creating multiple instances of Tomcat for IIS* on page 152.

To creating multiple Tomcat instance for Apache:

Step 1 Create copies of the Tomcat directory. See *Copying the Tomcat directory* on page 145.

Step 2 Delete the \webapps\oaa directory from the newly copied instances of Tomcat. See *Copying the Tomcat directory* on page 145.

Step 3 Edit the workers.properties file of the first or master Tomcat instance to set the values for each additional Tomcat instance. See *Editing the workers.properties file* on page 145.

Step 4 Edit the mod_jk.conf-auto file of the first or master Tomcat instance to establish a connection between Tomcat and Apache. See *Editing the mod_jk.conf-auto file* on page 147.

Step 5 Edit the httpd.conf file to define the Tomcat workers available for Apache. See *Editing the httpd.conf file* on page 147.

Step 6 Edit the server.xml files for each Tomcat instance. See *Editing the server.xml files for Apache* on page 148.

Step 7 Edit the jk2.properties files for each Tomcat instance. See *Editing the jk2.properties files for Apache* on page 149.

Step 8 Install multiple instances of Tomcat as a service using installservice.bat. This file can be found in the Tomcat\bin directory. See *Installing Tomcat instances as services for Apache* on page 150.

Step 9 Testing the configuration. See *Testing load balancing on Apache* on page 150.

Copying the Tomcat directory

You must create a separate folder for each instance of Tomcat you want to use for load balancing.

To copy the Tomcat directory:

1 Open Windows Explorer and copy the Tomcat install folder. By default, this folder is located at:

C:\Program Files\Peregrine\Common\Tomcat4

2 Paste a copy into the same root path. By default this would be:

C:\Program Files\Peregrine\Common

3 Rename the new folder to a unique name.

Tip: Include the port number to be used by the Tomcat instance in the folder name. For example, if you are going to use 4 instances of Tomcat listening on ports 8009, 8011, 8013, and 8015, then you could create 4 copies of the Tomcat folder called \Tomcat4_8009, \Tomcat4_8011, \Tomcat4_8013, and \Tomcat4_8015.

4 Delete the \webapps\oaa subdirectory from the newly copied instance of Tomcat.

The additional instances will use the same document root as the first or primary Tomcat instance.

5 Repeat step 1 through step 4 for each instance of Tomcat you want to use.

Editing the workers.properties file

For each server on which Tomcat instances are installed, there is only one workers.properties file. Tomcat installs the workers.properties file in the conf directory of your primary Tomcat instance. This file will be shared by all other Tomcat instances on that particular server.

The workers.properties file specifies the worker threads that the Web server connector will create in order to communicate with the Tomcat instances. Each Tomcat instance must communicate on a different port. The host should be set to the name of the server running the Tomcat instances or localhost if they are running on the same server as Apache.

Cache size is the maximum number of user sessions that Apache should direct to the Tomcat instance at one time.

Lbfactor is a number greater than or equal to 1 that Apache uses to load balance the workers. If all the workers are running on servers that have equal performance strengths, the lbfactor numbers should be equal. Workers with a lower lbfactor will be assigned fewer user sessions by the load balancer worker in Apache.

To edit the workers.properties file:

1 Open the workers.properties file in any text editor.

This file is located in the /conf directory of your Tomcat installation.

2 Edit the following lines as shown. The paths for workers.tomcat_home and workers.java.home are the locations of your Tomcat installation and Java SDK installations.

Example:

```
workers.tomcat_home="c:\Program Files\Peregrine\common\Tomcat4"
workers.java.home="c:\Program Files\Peregrine\common\jdk1.3.1_05"
ps=\
worker.list=loadbalancer, w8009, w8011, w8013, w8015
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=w8009, w8011, w8013, w8015
```

Note: You can define the worker names any way you want as long as you continue the same naming convention throughout the procedure.

3 Add the following lines for each Tomcat instance you have installed, incrementing the port number for the values shown in step 2:

```
worker.w8009.port=8009
worker.w8009.host=localhost
worker.w8009.type=ajp13
worker.w8009.cachesize=40
worker.w8009.lbfactor=10
```

4 Comment out the following lines. These default workers will not be used.

```
worker.ajp12.port=8007
worker.ajp12.host=localhost
worker.ajp12.type=ajp12
worker.ajp12.lbfactor=1

worker.ajp13.port=8009
worker.ajp13.host=localhost
worker.ajp13.type=ajp13
worker.ajp13.lbfactor=1
worker.ajp13.cachesize=10
```

5 Save the file.

Editing the mod_jk.conf-auto file

The mod_jk.conf-auto file defines where the Worker files are available in Apache. This file is shared by all Tomcat instances on the server. It is important that you do this procedure after you have successfully deployed the necessary Get-Services files, otherwise the Get-Services mount points, file locations, and directories will not be included in the mod_jk.conf-auto file, and you will have to manually add them.

To edit the mod_jk.conf-auto file:

1. Make a copy of the mod_jk.conf-auto file and rename the copy to mod_jk.conf-local.

   The mod_jk.conf-auto file is located in the Tomcat conf directory.

2. Open the mod_jk.conf-local file in any text editor.
3. Change JKWorkersFile to point to the worker.properties file of the primary Tomcat instance.

   Example:

   ```
   JkWorkersFile "C:\Apache\Tomcat\conf\worker.properties"
   ```

4. Change all JkMounts to use *loadbalancer* instead of *default worker ajp12*.

   Usage: JkMount<file(s) or directory> <worker name>

   Example:

   ```
   JkMount/oaa/servlet/* loadbalancer
   JkMount/oaa/*.jsp loadbalancer
   ```

5. Save the file.

Editing the httpd.conf file

The httpd.conf file must include mod_jk.conf-local.

To edit the httpd.conf file:

1. Open the httpd.conf file in any text editor.
2. Add the following line:

   ```
   include "<Tomcat>/conf/mod_jk.conf-local"
   ```

   For <*Tomcat*>, enter the path to your tomcat installation. By default the path is:

C:\Program Files\Peregrine\Common\Tomcat4

3 Save the file.

Editing the server.xml files for Apache

The server.xml file contains the information Tomcat needs to connect to the Web server as well as to find the Peregrine OAA Platform Web application files.

To edit the server.xml files:

1 Each Tomcat instance has a server.xml file located in the \conf directory. Open this file in any text editor.

2 Update the port number attribute of the <Server> element to a unique value that will not conflict with other port numbers used by Tomcat. It is recommended that the port numbers 8005-8008 be used when configuring four Tomcat instances.

Example:

```
<Server port="8005" shutdown="SHUTDOWN" debug="0">
```

3 Update the port number used by the Coyote Connector to a unique, non-conflicting value. If you are configuring four Tomcat instance, the values 8009, 8011, 8013, and 8015 are recommended.

Example:

```
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8009" minProcessors="5" maxProcessors="75" enableLookups="true"
redirectPort="8443" acceptCount="10" debug="0"
connectionTimeout="20000" useURIValidationHack="false"
protocolHandlerClassName="org.apache.jk.server.JkCoyoteHandler" />
```

4 Update the jvmRoute attribute of the <Engine> element with the the server name and communications port used by each Tomcat instance.

Example:

```
<Engine jvmRoute="localhost:8009" name="Standalone"
defaultHost="localhost" debug="0">
```

The port number should follow the convention used elsewhere in the configuration (8009, 8011, etc.). These entries must be the same as the Tomcat ID entries you added to the workers2.properties file.

5 Update the appBase attribute of the <Host> element with the absolute path to the webapps directory of the priomary Tomcat instance.

Example:

```
<Host name="localhost" debug="0"
appBase="C:\Apache\Tomcat_w8009\webapps" unpackWARs="true"
autoDeploy="true">
```

6 Comment out a <Connector> tag with the className="org.apache.coyote.tomcat4.CoyoteConnector" using port 8080.

Tomcat uses this port to communicate with a browser for direct HTTP requests. Since Apache will be serving the static data, Tomcat does not need to listen on this connector. It will also prevent a user from directly accessing Tomcat instances.

Example:

```
<!--
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8080" minProcessors="5" maxProcessors="75" enableLookups="true"
redirectPort="8443" acceptCount="10" debug="0"
connectionTimeout="20000" useURIValidationHack="false" />
-->
```

7 Change the OAA context so that it is not reloadable.

This prevents Tomcat from reloading the servlet without restarting the service. This improves performance and helps keep the JSP code that the Tomcat instances are serving in sync during an update. All other contexts should be set to reload=false.

Example:

```
<Context path="/oaa"
docBase="oaa"
crossContext="false"
debug="0"
reloadable="false" >
</Context>
```

8 Save the file.

9 Repeat step 2 through step 7 for each copy of the server.xml file you made.

Editing the jk2.properties files for Apache

You will need to modify each Tomcat instance's jk2.properties This file sets the jk2 communication port.

To edit the jk2.properties files:

1 Open the jk2.properties file for a Tomcat instance in a text editor.

This file is located in the Tomcat conf directory.

2 Insert a line for the channelSocket port. The port number must match the port number defined in workers.properties file for this Tomcat instance.

Example:

```
channelSocket.port=8009
```

3 Save the file.

4 Repeat step 1 through step 3 for each Tomcat instance.

Installing Tomcat instances as services for Apache

After you have edited the Tomcat files, you can install each instances of Tomcat as Windows services using installservice.bat.

To install Tomcat instances as services on Apache:

1 Open a DOS command prompt and change directories to your Tomcat bin directory.

2 Enter the following command to create each Tomcat instance:

```
installservice <service name> <tomcat_home> <jvm_dll_path>
```

Where *<service name>* is the name you wish to give the Tomcat service, *<tomcat_home>* is the Tomcat install directory of the instance for which you are creating the service, and *<jvm_dll_path>* is the Java SDK install directory.

The second and third parameters are optional if you have already set the CATALINA_HOME and JAVA_HOME environment variables.

Example:

```
installservice Tomcat8009 C:\Apache\Tomcat8009 C:\jdk130_05
```

3 Repeat step 1 through step 2 for each Tomcat service you wish to create.

Testing load balancing on Apache

After you have created additional Tomcat instances, you can test if load balancing is occurring using the following steps.

To test load balancing:

1 Start all Tomcat instance services.

If you installed Tomcat as a service you can open the Windows Control Panel and start each instance from the Services dialog box.

2 Open a browser and log in to Get-Services.

3 Perform an action in Get-Services. For example, perform a search.

4 Logout of Get-Services.

5 Close your browser to clear the connection cache.

6 Repeat step 1 through step 5 one time for each Tomcat instance installed. For example, if you have 4 Tomcat instances, then you will need to login and logout a total of 4 times.

   The load balancing mechanism uses a Round-Robin algorithm. If load balancing is working successfully, each login attempt should use a different Tomcat instance.

7 Download the archway.log file.

   You can download the archway.log file from the **Administration** > **Server Log** page.

8 Open the archway.log file in a text editor.

9 Verify that connection details list a different Tomcat instance for each connection.

   If each connection uses a different Tomcat instance, then the system is load balancing properly.

   If each connection uses the same Tomcat instance, the system is not load balancing and needs troubleshooting.

Creating multiple instances of Tomcat for IIS

Multiple instances of Tomcat are installed as services. Although this is not required, it improves performance, makes the instances easier to manage, and provides extra functionality, including restarting the service if it fails or if the machine on which the instances are installed needs to be restarted.

To creating multiple Tomcat instance for IIS:

Step 1 Create copies of the Tomcat directory. See *Copying the Tomcat directory* on page 153.

Step 2 Delete the \webapps\oaa directory from the newly copied instances of Tomcat. See *Copying the Tomcat directory* on page 145.

Step 3 Configure the ISAPI Plugin for IIS. See *Configuring the ISAPI Plugin for IIS* on page 153.

Step 4 Create and configure a jakarta virtual directory in IIS. See *Creating and configuring a jakarta virtual directory in IIS* on page 154.

Step 5 Configure IIS to use isapi_redirector2.dll as an ISAPI Filter. See *Configuring the isapi_redirector2.dll as an ISAPI filter* on page 154.

Step 6 Create and configure an oaa virtual directory in IIS. See *Creating and configuring an oaa virtual directory in IIS* on page 155.

Step 7 Edit the workers2.properties file of the first or master Tomcat instance to set the values for each additional Tomcat instance. See *Editing the workers2.properties file for IIS* on page 155.

Step 8 Edit the server.xml files for each Tomcat instance. See *Editing the server.xml files for IIS* on page 157.

Step 9 Edit the jk2.properties files for each Tomcat instance. See *Editing the jk2.properties files for IIS* on page 158.

Step 10 Install multiple instances of Tomcat as a service using installservice.bat. This file can be found in the Tomcat\bin directory. See *Installing Tomcat instances as services for IIS* on page 159.

Step 11 Testing the configuration. See *Testing load balancing on IIS* on page 159.

Copying the Tomcat directory

You must create a separate folder for each instance of Tomcat you want to use for load balancing.

To copy the Tomcat directory:

1 Open Windows Explorer and copy the Tomcat install folder. By default, this folder is located at:

C:\Program Files\Peregrine\Common\Tomcat4

2 Paste a copy into the same root path. By default this would be:

C:\Program Files\Peregrine\Common

3 Rename the new folder to a unique name.

Tip: Include the port number to be used by the Tomcat instance in the folder name. For example, if you are going to use 4 instances of Tomcat listening on ports 8009, 8011, 8013, and 8015, then you could create 4 copies of the Tomcat folder called \Tomcat4_8009, \Tomcat4_8011, \Tomcat4_8013, and \Tomcat4_8015.

4 Delete the \webapps\oaa subdirectory from the newly copied instance of Tomcat.

The additional instances will use the same document root as the first or primary Tomcat instance.

5 Repeat step 1 through step 4 for each instance of Tomcat you want to use.

Configuring the ISAPI Plugin for IIS

The Get-Services installer automatically places a copy of the ISAPI plugin for IIS in the following folder:

c:\Program Files\Peregrine\Common\Tomcat4\bin

Use the following procedures to configure the plugin for your intranet environment.

To configure the ISAPI plugin for IIS:

1 Open the file jk2.reg in a text editor. This file is located at:

C:\Program Files\Peregrine\Common\Tomcat4\conf

2 Verify that the "serverRoot" and "workersFile" values list the proper installation path to Tomcat. By default these values are:

```
"ServerRoot"="C:\\Program Files\\Peregrine\\Common\\Tomcat4"
"workersFile"="C:\\Program Files\\Peregrine\\Common\\Tomcat4\\conf\\
workers2.properties"
```

3 Save and close the jk2.reg file.

4 Double-click on the jk2.reg file from Windows Explorer.

Windows adds the registry settings to the Windows registry.

Creating and configuring a jakarta virtual directory in IIS

The ISAPI plugin for IIS requires a specific IIS virtual directory in order to run. Use the following guidelines to create the IIS virtual directory. For specific instructions about IIS, refer to Windows Help.

Requirements for jakarta virtual directory

Requirement	Setting
Create virtual directory	jakarta
Directory access rights	execute
Map to physical path	<Tomcat>\bin\isapi_redirector2.dll

For <*Tomcat*>, enter the path to your tomcat installation. By default the path is:

C:\Program Files\Peregrine\Common\Tomcat4

Configuring the isapi_redirector2.dll as an ISAPI filter

To establish a connection between Tomcat and IIS, you will need to install the file isapi_redirector2.dll as an ISAPI filter.

To install isapi_redirect2.dll as an ISAPI filter:

1 Open Internet Services management console.

2 Right-click the **Default Web Site** node and then click **Properties**.

3 Click the **ISAPI Filters** tab.

4 Click **Add**.

5 Enter the following information:

a **Filter Name:** jakarta. The filter name must match the name you defined the jk2.reg registry file. By default the filter name is jakarta.

b **Executable:** isapi_redirector2.dll. By default this file is located at:

C:\Program Files\Peregrine\Common\Tomcat4\bin\isapi_redirector2.dll

6 Click **OK**.

7 Close the Internet Services management console.

Creating and configuring an oaa virtual directory in IIS

To run Get-Services from IIS, you need to create a virtual directory that maps to your Tomcat deployment folder.

Requirements for oaa virtual directory

Requirement	Setting
Create virtual directory	oaa
Directory access rights	anonymous
Map to physical path	<Tomcat>\webapps\oaa
Remove "allow anonymous access" to	default.asp, login.asp, e_login_main_start.asp
Set security access to only allow "System" and "Authenticated Users" to	default.asp, login.asp, e_login_main_start.asp

For <*Tomcat*>, enter the path to your tomcat installation. By default the path is:

C:\Program Files\Peregrine\Common\Tomcat4

Editing the workers2.properties file for IIS

For each server on which Tomcat instances are installed, there is only one workers2.properties file. Tomcat installs the workers.properties file in the conf directory of your primary Tomcat instance. This file will be shared by all other Tomcat instances on that particular server.

The workers2.properties file specifies the worker threads that the Web server connector will create in order to communicate with the Tomcat instances. Each Tomcat instance must communicate on a different port. The host should be set to the name of the server running the Tomcat instances or localhost if they are running on the same server as Apache.

To edit the worker2.properties file:

1. Open the workers2.properties file (located in the config directory of your Tomcat installation) in any text editor.
2. Create a channel.socket entry for each Tomcat instance (also known as a worker).

   Example:

   ```
   [channel.socket:<server>:<port>]
   info=Description of Tomcat instance
   debug=0
   tomcatId=<server>:<port>
   lb_factor=1
   disabled=0
   ```

   For <*server*>, enter the server name of where the Tomcat instance is located.

   For <*port*>, enter the communications port on which the Tomcat instance is listening.

   lb_factor is a number greater than or equal to 1 that IIS uses to load balance the workers. If all the workers are running on servers that have equal performance strengths, you should set the lb_factor numbers to equal values (typically 1). If you want to assign fewer user sessions to a given Worker, then assign it a lower lb_factor number relative to the other Workers.
3. Verify that the uri settings lists the proper IIS virtual directory. By default the virtual directory should be oaa.

   If you have defined a different virtual directory other than oaa to run Get-Services, you will need to change the uri values here.

   Example:

   ```
   [uri:/oaa/servlet/*]
   info=Prefix mapping

   [uri:/oaa/*.jsp]
   info=Extension mapping
   ```

4. Save the file.

Editing the server.xml files for IIS

You will need a separate **server.xml** file for each Tomcat instance that will be running concurrently. This file contains the information Tomcat needs to connect to the Web server as well as to find the Peregrine OAA Platform Web application files.

To edit the server.xml files:

1. Each Tomcat instance has a server.xml file. Open it in any text editor.
2. Update the port number attribute of the <Server> element to a unique value that will not conflict with other port numbers used by Tomcat.

   It is recommended that the port numbers 8005-8008 be used when configuring four Tomcat instances.

   Example:

   ```
   <Server port="8005" shutdown="SHUTDOWN" debug="0">
   ```

3. Update the port number attribute of the Coyote Connector <Connector> element to a unique value that will not conflict with other port numbers used by Tomcat.

   It is recommended that the port numbers 8009, 8011, 8013, and 8015 be used when configuring the Coyote Connector.

   Example:

   ```
   <Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
   port="8009" minProcessors="5" maxProcessors="75" enableLookups="true"
   redirectPort="8443" acceptCount="10" debug="0"
   connectionTimeout="20000" useURIValidationHack="false"
   protocolHandlerClassName="org.apache.jk.server.JkCoyoteHandler" />
   ```

4. Create a <Context> element entry from the first or master Tomcat instance to the Get-Services deployment directory.

   Add the entry just above the "examples" Context entry.

   Example:

   ```
   <Context path="/oaa"
   docBase="<First Tomcat install>/webapps/oaa"
   crossContext="false"
   debug="0"
   reloadable="false" >
   </Context>
   ```

   For the **docBase** attribute, set *<First Tomcat install>* to the absolute path of the first or master Tomcat instance.

5 Update the <Engine> element with the server name and communications port used by each Tomcat instance.

List the server information in the jvmRoute attribute.

Example:

```
<Engine jvmRoute="localhost:8009" name="Standalone"
defaultHost="local host" debug="0">
```

6 Update the <Host> element with the webapps directory used by the first or master Tomcat instance.

List the server information in the appBase attribute.

Example:

```
<Host name="localhost" debug="0" appBase="<First Tomcat
install>/webapps" unpackWARs="true" autoDeploy="true">
```

For the appBase attribute, set *<First Tomcat install>* to the absolute path of the first or master Tomcat instance.

7 Save the file server.xml.

8 Repeat step 2 through step 7 for each copy of the server.xml file you made.

Editing the jk2.properties files for IIS

You will need to modify each Tomcat instance's jk2.properties This file sets the jk2 communication port.

To edit the jk2.properties files:

1 Open the jk2.properties file for a Tomcat instance in a text editor.

2 Insert a line for the channelSocket port. The port number must match the port number defined in workers2.properties file for this Tomcat instance.

Example:

```
channelSocket.port=8009
```

3 Save the file.

4 Repeat step 1 through step 3 for each Tomcat instance.

Installing Tomcat instances as services for IIS

After you have edited the Tomcat files, you can install each instances of Tomcat as Windows services using installservice.bat.

To install Tomcat instances as services on IIS:

1. Open a DOS command prompt and change directories to your Tomcat bin directory.
2. Enter the following command to create each Tomcat instance:

```
installservice <service name> <tomcat_home> <jvm_dll_path>
```

Where *<service name>* is the name you wish to give the Tomcat service, *<tomcat_home>* is the Tomcat install directory of the instance for which you are creating the service, and *<jvm_dll_path>* is the Java SDK install directory.

The second and third parameters are optional if you have already set the CATALINA_HOME and JAVA_HOME environment variables.

Example:

```
installservice Tomcat8009 C:\Program Files\Peregrine\Common\Tomcat4
C:\Program Files\Peregrine\Common\jdk1.3.1_05
```

3. Repeat step 1 through step 2 for each Tomcat service you wish to create.

Testing load balancing on IIS

After you have created additional Tomcat instances, you can test if load balancing is occurring using the following steps.

To test load balancing:

1. Start all Tomcat instance services.

   If you installed Tomcat as a service you can open the Windows Control Panel and start each instance from the Services dialog box.
2. Open a browser and log in to Get-Services.
3. Perform an action in Get-Services. For example, perform a search.
4. Logout of Get-Services.
5. Close your browser to clear the connection cache.
6. Repeat step 1 through step 5 one time for each Tomcat instance installed. For example, if you have 4 Tomcat instances, then you will need to login and logout a total of 4 times.

The load balancing mechanism uses a Round-Robin algorithm. If load balancing is working successfully, each login attempt should use a different Tomcat instance.

7 Download the archway.log file.

You can download the archway.log file from the **Administration** > **Server Log** page.

8 Open the archway.log file in a text editor.

9 Verify that connection details list a different Tomcat instance for each connection.

If each connection uses a different Tomcat instance, then the system is load balancing properly.

If each connection uses the same Tomcat instance, the system is not load balancing and needs troubleshooting.

5 ServiceCenter Administration

CHAPTER

You configure Get-Services to use ServiceCenter as the back-end system. With ServiceCenter, Get-Services accesses ServiceCenter Incident Mangement. Using the Get-Services Admin module, you can add the Service Management component as well. Change Management is also available with ServiceCenter 5.0.

Important: To continue the installation for Get-Services, an administrator must perform the procedures in this chapter within the ServiceCenter back-end system. For detailed information on how to perform any of the steps listed in the following sections, refer to the *ServiceCenter Administration Guide*.

This chapter includes the following sections:

Configuring ServiceCenter for Get-Services

Using ServiceCenter, Get-Services accesses the ServiceCenter 4.x and ServiceCenter 5.x Incident Management and Service Management modules. In addition, Get-Services can access the ServiceCenter 5.x Change Management module.

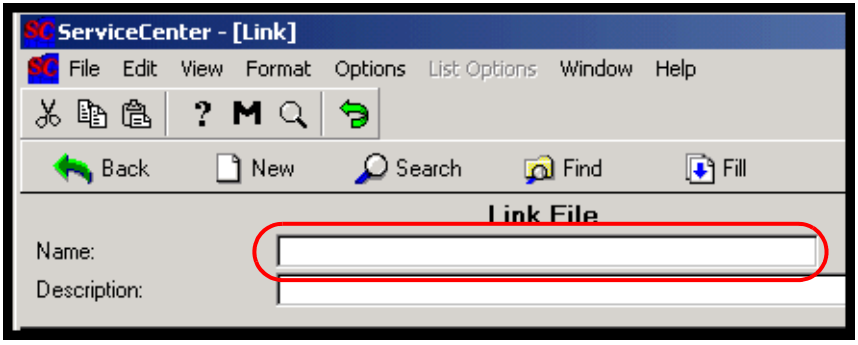
Users are authenticated using ServiceCenter Operator records, with special capability words provided for use with Get-Services. Refer to the Security chapter in the *Get-Services Administration Guide* for more information about user authentication.

Updating ServiceCenter

You must update ServiceCenter 4.x to take advantage of the Get-Services 4.0 functionality of opening a related Incident from a Call. This change facilitates transfer of data from Calls to Incidents. This procedure is not necessary for ServiceCenter 5.x.

To update ServiceCenter 4.x to include new Get-Services functionality:

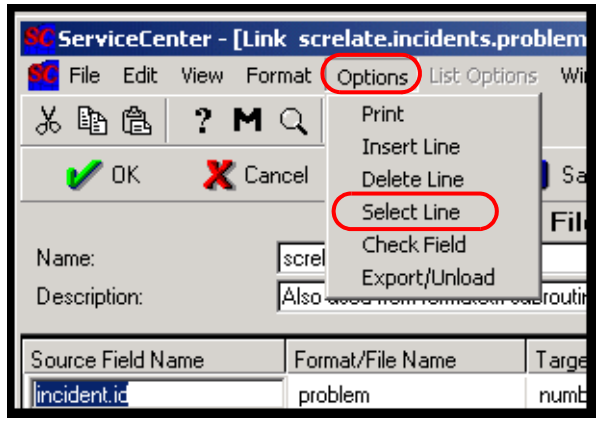
1 From the ServiceCenter Utilities tab, click Tools, then Links to open the Link dialog box.



Type the name of the Link File in the Name text box.

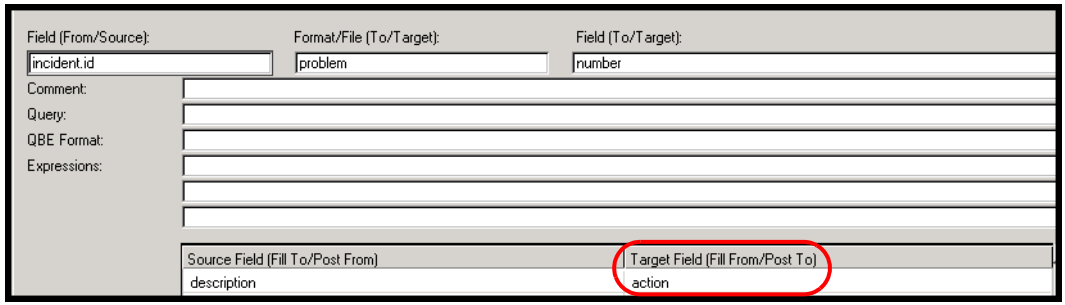
2 In the **Name** text box, type **screlate.incidents.problem**, then click **Search**.

3 Click **incident.id** to highlight it, then select **Options** > **Select Line**.



Highlight incident.id in the Source Field Name.

4 In the **description** row, change the Target Field (Fill From/Post to) text from **Action, action** to **action**.



Click **Save**.

Applying the unload files to ServiceCenter

The next step in the Get-Services installation involves installing new Formats, Display Options, and RAD codes that are contained in unload files. The unload files that you apply depend on how you intend to use Get-Services. You apply different unload files to ServiceCenter 4.x and ServiceCenter 5.x. See the respective tables in the following sections.

The **\oaa** directory referenced in the following tables is a subdirectory of the directory where the Peregrine OAA Server files are installed. For example, if you installed Tomcat 4.1.12 as your Peregrine OAA server, the **\oaa** directory is a subdirectory of the **\Tomcat4\webapps** directory on the machine where you installed Tomcat 4.1.12.

Warning: This procedure overwrites existing files when you apply the new unload files. If you tailored your system, you will lose your changes.

ServiceCenter 4.x unload files with Get-Services 4.0

The path for the unload files is in oaa\WEB-INF\etc\Version4\.

This file	Contains this information
axcessm.unl	rad: axces.sm
callmgmt.unl	eventmap: e service management eventregister: esmin
callupdate.unl	eventmap: e service management
epmx.unl	capability: getit.answers getit.service contacts: Hartke Tossi eventmap: e problem close e problem open e problem update eventregister: epmc epmo epmu operator: Hartke Tossi
gs22upd.unl	rad: apm.bg.edit apm.get.display.format
openrelated.unl	eventregister: epmosmu eventmap: e problem open smu rad: axces.apm.epmosmu cc.open.related.incident

ServiceCenter 5.0 unload files with Get-Services 4.0

The path for the unload files is in oaa\WEB-INF\etc\Version5\.

This file	Contains this information
axcessm.unl	rad: axces.sm
callmgmt.unl	eventmap: e service management eventregister: esmin
callupdate.unl	eventmap: e service management
epmx.unl	capability: getit.answers getit.service contacts: Hartke Tossi eventmap: e problem close e problem open e problem update eventregister: epmc epmo epmu operator: Hartke Tossi
gs22upd.unl	rad: apm.bg.edit apm.get.display.format
openrelated.unl	eventregister: epmosmu eventmap: e problem open smu rad: axces.apm.epmosmu cc.open.related.incident

ServiceCenter 5.0 unload files with Get-Services 4.0 Change Management

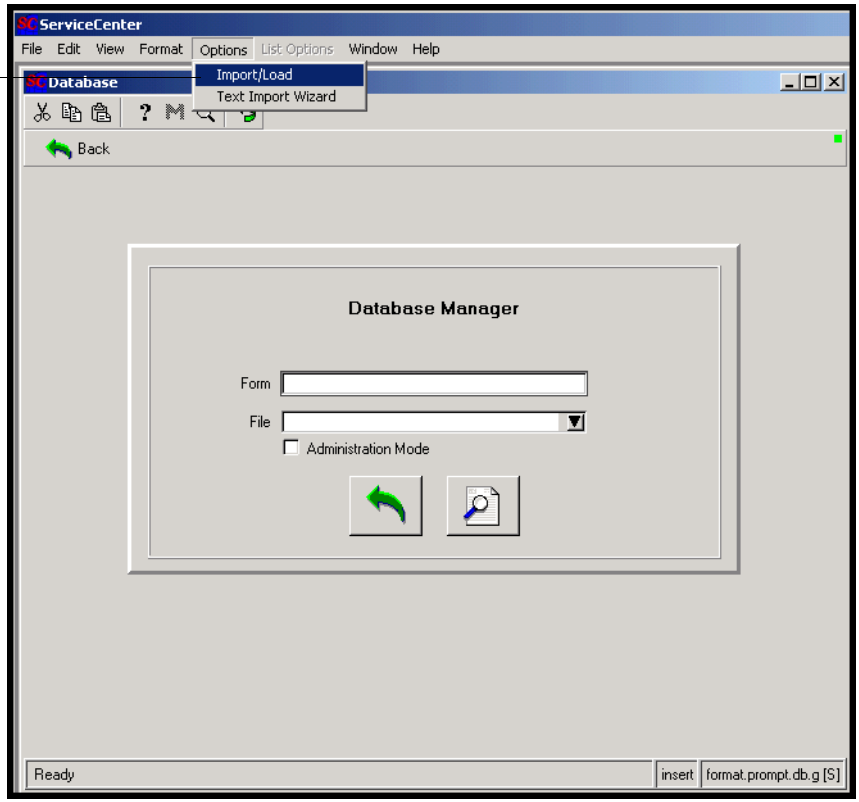
If you intend to use Get-Services 4.0 with Change Management, you need to apply the following unload files. The path for the unload files is in oaa\WEB-INF\etc\Version5\.

This file	Contains this information
sc5_cmreopenall.unl	rad: cm3.unlock axces.cm3 cm3.lock cm3r.main cm3t.main process: cm.update.save eventmap: cm3r
sc5_cmsctructure.unl	eventmap: cm3t

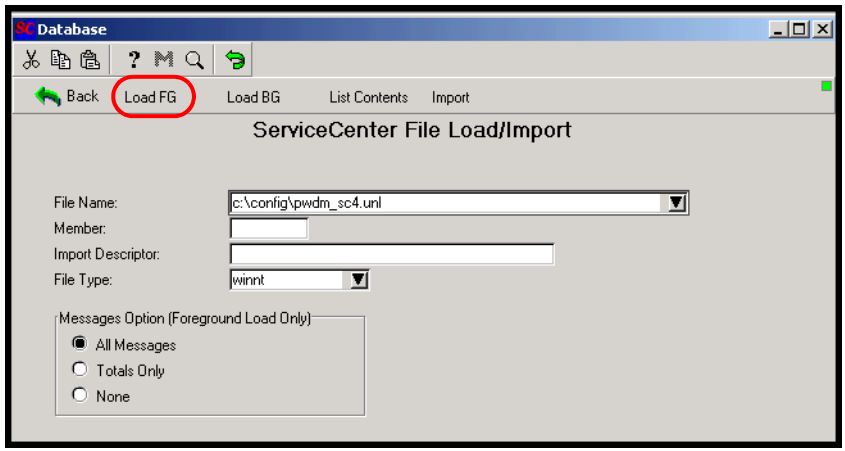
To apply unload files to ServiceCenter:

1 In ServiceCenter, go to Toolkit and choose Database Manager. The ServiceCenter Database Manager dialog box opens.

Choose Import/Load from the Options command.



2 From **Options**, choose Import/Load to open the **File Load/Import** dialog box.



3 In the **Filename** text box, type the path for the unload file that pertains to your particular integration. See the previous tables to determine which unload files pertain to your integration.

4 Click **Load FG** and note any errors.

5 Repeat step 3 and step 4 for each unload file that pertains to your integration.

6 Configuring the Adapters

CHAPTER

Final configuration settings necessary to complete the installation of Get-Services are performed in the Peregrine Portal Administration module. This includes parameter configuration and login information for the back-end system, as well as verification of adapter connectivity for Get-Services.

A more detailed description of additional features necessary for administration and maintenance using the Peregrine Portal Administration utility is available in the *Get-Services Administration Guide*.

Note: Incident Management is called Problem Management in ServiceCenter versions prior to 4.x. Some parameters in Incident Management use *problem* terminology because they are mapped to *problem* tables in ServiceCenter.

This chapter covers the following topics:

Accessing the Peregrine Portal Admin module

The Peregrine Portal administrator login page accesses the Peregrine Portal Admin module. You use the Admin module to define the settings for your Peregrine system.

Note: After installing and building Get-Services, an Admin must log in as a ServiceCenter user with **getit.admin** rights to access the Admin module and administer the Get-Services integration with ServiceCenter. The list of access capability words is in the *Get-Services Administration Guide*.

A default administrator, Admin, gives you access to the Admin module without being connected to a back-end system. After you configure your user name on the Common tab, you can also access the Admin module from the Navigation menu.

Important: When you change parameters using the Admin module, a local.xml file is created in the \<*application server*>\webapps\oaa\WEB-INF directory to store these parameters. If you reinstall Get-Services, make a copy of this file and store it outside your Get-Services installation. Failure to do this will result in your parameter values being lost during the new installation.

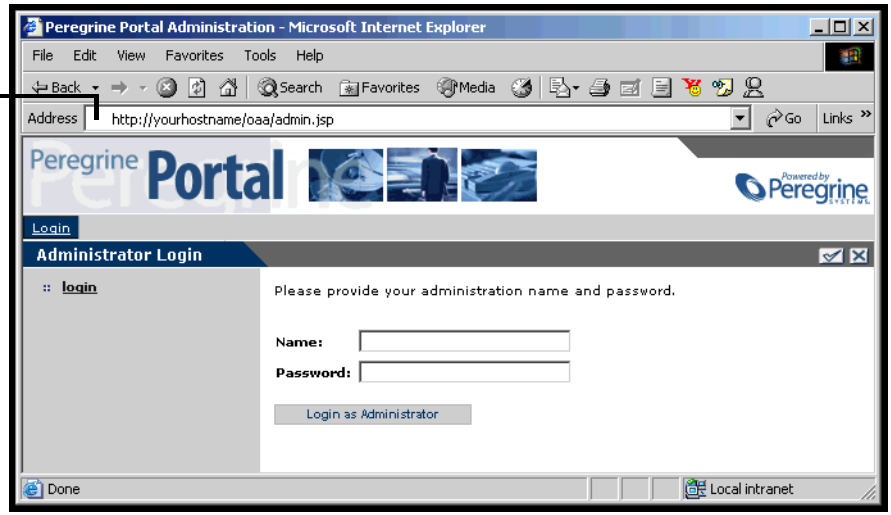
To access the Peregrine Portal administrator login page:

1. Verify that your application server (for example, Tomcat) is running.
2. In your Web browser Address field, type:

   <*hostname*>**/oaa/admin.jsp**

3 Press Enter to open the Portal administrator login page.

Type your hostname to connect to your local server.



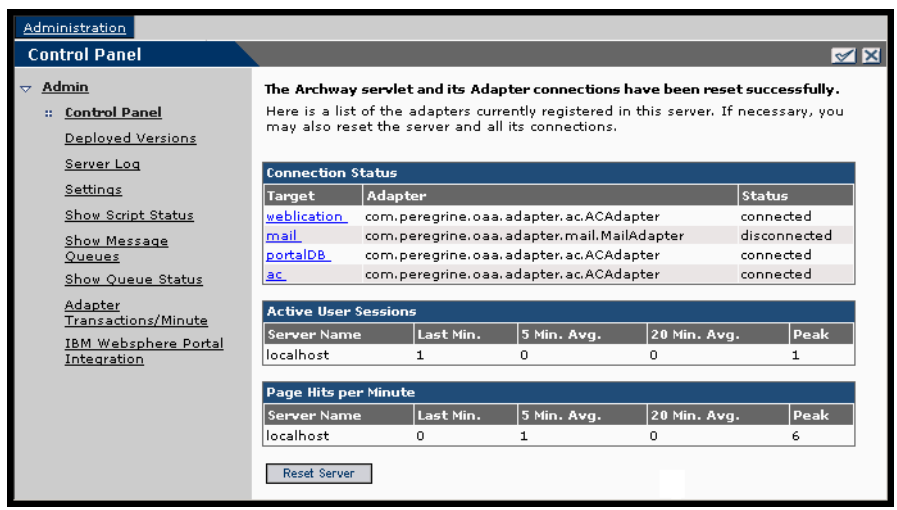
Admin is the default administrator name.

4 In the Name field, type **Admin**.

No password is required on initial login.

5 Click **Login as Administrator** to open the **Control Panel** page.

Use the Admin module to define settings to Get-Services.



Activity menu

The left pane Activity menu provides access to a number of different administrative utilities for Get-Services. You need to access some of these during this installation process. The utilities and their functions are described in detail in the *Get-Services Administration Guide*.

Using the Control Panel

Use the Control Panel page to check the status of the connections to the databases you are accessing with Get-Services and your Web applications. You can also reset the connection between the Archway servlet and the adapters to the back-end systems.

Note: When you first access the Control Panel page, the status for all targets is *disconnected*. This will change when you define the targets later on in this section.

After making any configuration changes in the Peregrine Portal Administration module, the system returns to the Admin Control Panel and displays an informational message at the top of the page.

To reset the connection between the Archway servlet and back-end system:

1 Click **Reset Server.**

The following message indicates that the connections are reset.

The Archway servlet and its Adapter connections have been reset successfully.

2 Verify your changes in the Connection Status table.

Using the Settings page

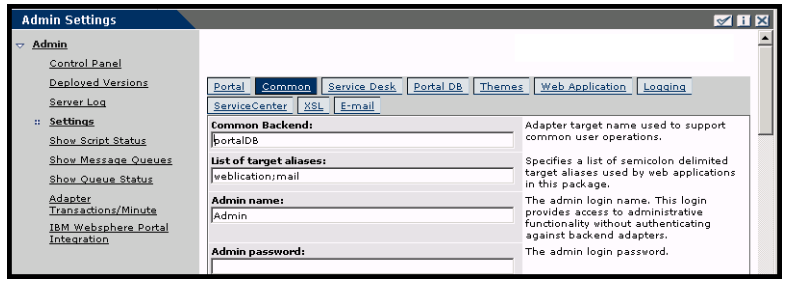
From the Activity menu, click **Settings** to open the current parameter settings. The Settings page is divided into tabs. The tabs that you see depend on the Web applications that you installed and the adapters that you use. The Common tab is available for all installations.

To open the Settings page:

1 From the Admin module Activity menu, click **Settings**.

The Settings page opens, providing access to the various tabs used to configure settings for Get-Services.

By default, the Common settings page opens.



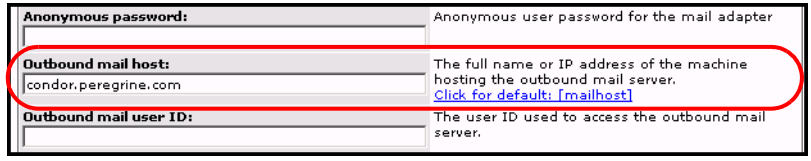
2 Click the appropriate tab to change setting parameters.

When you change the parameter default value for any setting and save it, the corresponding description column adds a link with the default value listed in brackets so that you can return to the default at any time.

To revert to the default setting:

1 Navigate to the appropriate setting parameter.

2 From the description column next to the changed parameter, select the **Click for default** link.



3 To return to the default setting, select the **Click for default** [**mailhost**] link and click **Save**.

The parameter returns to the default state.

Configuring connections to the back-end system

By default, Get-Services installs with all of the connectivity required to interface with ServiceCenter through a special adapter, the SCAdapter.

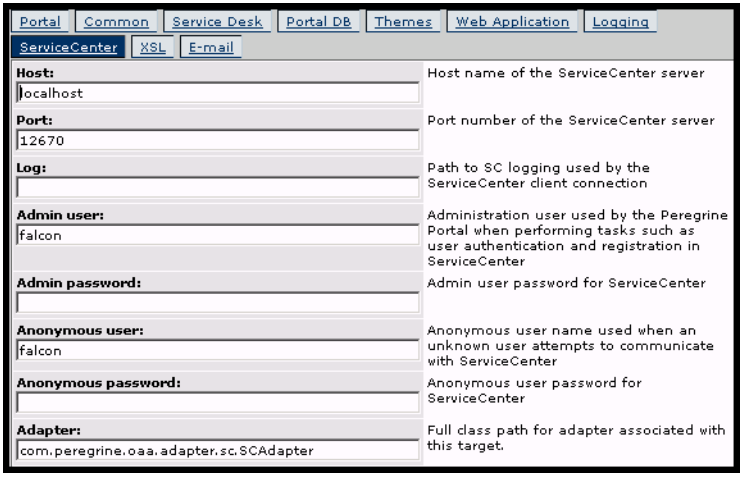
Setting the ServiceCenter adapter

This section describes how to configure and verify back-end system settings and connect to the ServiceCenter database.

To set the ServiceCenter adapter:

1 From Admin, click **Settings**.

2 At the top of the page, click the **ServiceCenter** tab to open the ServiceCenter Settings page.



3 Update the following fields, as needed:

Field name	Default value	Value description
Host	localhost	Host name of your ServiceCenter server
Port	12670	Port number of your ServiceCenter server
Log	(none)	The path to SC logging that the ServiceCenter client connection uses

Field name	Default value	Value description
Admin user	falcon	Administration login name that the Peregrine Portal uses when performing tasks such as user authentication and registration
Admin password	(none)	Administrator password
Anonymous user	falcon	Name to use when unknown user attempts to connect with ServiceCenter
Anonymous password	(none)	Anonymous user password
Adapter	com.peregrine.oaa.adapter.sc.SCAdapter	Full class path for adapter associated with this target.

4 If you make any changes, click **Save** to return to the Admin Control Panel page, then click **Reset Server**.

Troubleshooting the ServiceCenter database connection

If you are having trouble making a connection between Peregrine OAA and the ServiceCenter database, verify the following.

To troubleshoot the ServiceCenter database connection:

1 From the Admin module, check the Control Panel page to confirm the database connectivity status.

- If **sc** is disconnected, verify that the ServiceCenter service is running and the ServiceCenter console has been started.

2 From the Settings page ServiceCenter tab:

a Verify that the parameters for **Host** and **Port** are correct.

b Verify that the Admin user name and password defined for Get-Services are the same login values used when logging directly into the ServiceCenter back-end system as an Administrator.

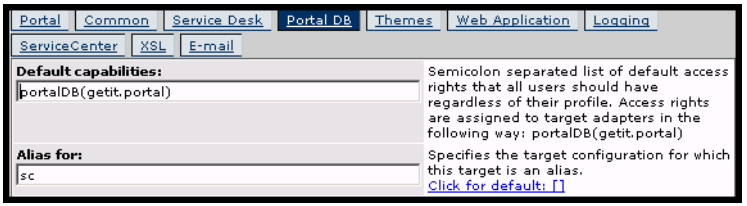
3 Verify that you have ServiceCenter full client connectivity by starting a client that points to the port listed on the Settings page in the Admin module.

Setting the Portal DB adapter

The Peregrine Portal requires a database adapter connection to store settings and customizations to the portal interface. Until a database adapter is defined for the portal page, users are unable to make see or make personalization changes to the Peregrine portal home page.

To set the Portal DB adapter:

1 From the Admin Settings page, click **Portal DB** to open the PortalDB Settings page.



The following table describes the parameters that define the Portal DB adapter for the Portal page.

Field name	Default value	Value description
Default capabilities	getit.portal	A list of default access rights for all users regardless of their user role. The Portal DB default capability word can accept any OAA capability word. Peregrine recommends that you use the default setting to maintain optimal security.
Alias for	sc	The database adapter name you want to use to store settings and changes to the Peregrine Portal page. Use the same database adapter that your application requires.

2 In the **Alias for** field, type **sc** to set the ServiceCenter database adapter.

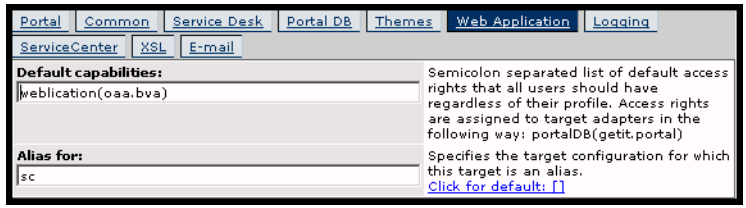
3 Click **Save** to return to the Admin Control Panel page, then click **Reset Server**.

Setting the Web Application database adapter

Get-Services requires a database adapter connection to store settings and customizations to the web application's interface screens. Until a database adapter is defined for the web application, users cannot make personalization changes to the web application.

To set the Web Application database adapter:

1 From the Admin Settings page, click **Web Application** to open the Web Application Settings page.



2 In the **Alias for** field, type **sc**, then click **Save** to return to the Admin Control Panel page.

3 Click **Save** to return to the Admin Control Panel page, then click **Reset Server**.

Setting Service Desk parameters

This section lists parameters that are specific to Get-Services. You configure these settings with the **Service Desk** tab on the Admin Settings page.

Incident Management is the default module used for incident (problem) tickets opened in Get-Services with the ServiceCenter adapter. If you want end users also to create ServiceCenter call tickets, you must enable the Service Management module and configure the appropriate Get-Services settings.

To enable Service Management for Get-Services:

1 From the Peregrine Portal Admin module, click Settings, then click the **Service Desk** tab.

2 Select **Yes** in the Enable Service Management parameter if you want tickets created from Services to be opened in the Service Management module of your ServiceCenter installation.

3 Click **Save** to return to the Control Panel.

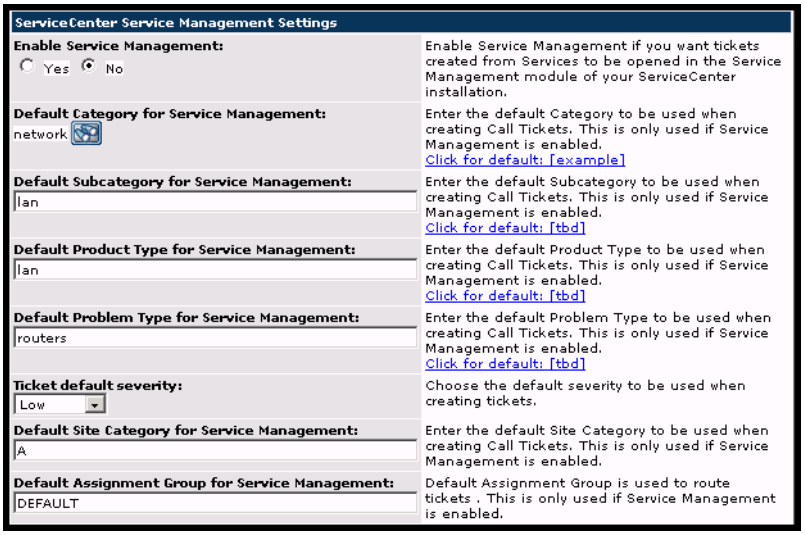
4 Click **Reset Server** to save your changes.

Service Management

When you enable the Service Management module, end users can open and view both Incident tickets and Call tickets. You can then change Service Management parameters as needed.

To set Service Management for Get-Services:

1 From the Peregrine Portal Admin module, click Settings, then click the **Service Desk** tab and scroll to the ServiceCenter Service Management Settings.



2 Update the following fields as needed to define the ServiceCenter Service Management settings for Get-Services.

Field name	Default value	Value description
Enable Service Management	No	When set to Yes, tickets created in Get-Services are opened using the Service Management module of your ServiceCenter installation.
Default Category for Service Management	example	The default ServiceCenter category used for creating Call tickets. This parameter is used only if Service Management is enabled.
Default Subcategory for Service Management	tbd	The default ServiceCenter Subcategory used for creating Call tickets. This parameter is used only if Service Management is enabled.
Default Product Type for Service Management	tbd	The default Product Type used when creating Call tickets. This parameter is used only if Service Management is enabled
Default Problem type for Service Management	tbd	Defines the default ServiceCenter category for Problem tickets.
Ticket default severity	Low	The default severity used when creating tickets from the drop-down list. This parameter is used only if Service Management is enabled.
Default Site Category for Service Management	A	The default Site Category used when creating Call tickets. This parameter is used only if Service Management is enabled.
Default Assignment Group for Service Management	Default	Used to route tickets, this is only used if Service Management is enabled.

3 Click **Save** to return to the Control Panel.

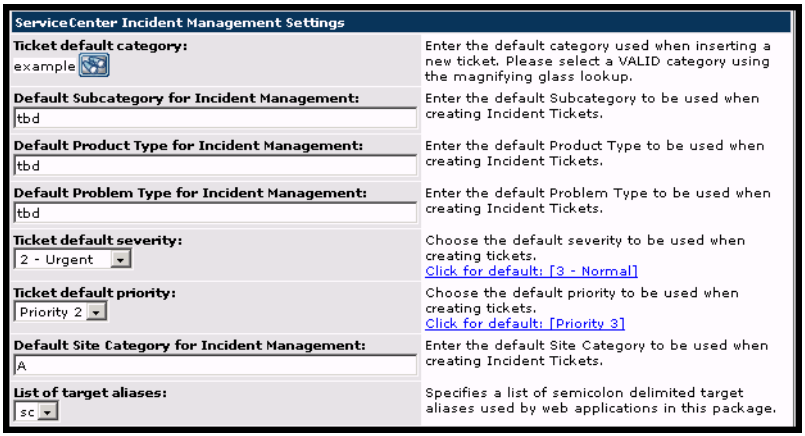
4 Click **Reset Server** to save your changes.

Incident Management

The following parameters control the default settings that Incident Management uses when a user opens a Get-Services ticket. These settings are in the Service Desk tab on the Admin Settings page.

To set Incident Management for Get-Services:

1 From the Peregrine Portal Admin module, click Settings, then click the **Service Desk** tab and scroll to the ServiceCenter Incident Management Settings.



The following table describes the parameters that define the ServiceCenter Incident Management settings for Get-Services.

Field name	Default value	Value description
Ticket default category	example	The default Category used when creating new Incident tickets.
Default Subcategory for Incident Management	tbd	The default Subcategory used when creating Incident tickets.
Default Product Type for Incident Management	tbd	The default Product Type used when creating Incident tickets.
Default Problem Type for Incident Management	tbd	The default Problem Type used when creating Incident tickets.
Ticket default severity	3 - Normal	The default Severity used when creating Incident tickets.

Field name	Default value	Value description
Ticket default priority	Priority 3	The default priority when Creating tickets.
Default Site Category for Incident Management	A	The default Site Category used when creating Incident tickets.
List of target aliases	sc	A list of semicolon delimited target aliases that the Web application uses.

2 Change the parameters, as needed.

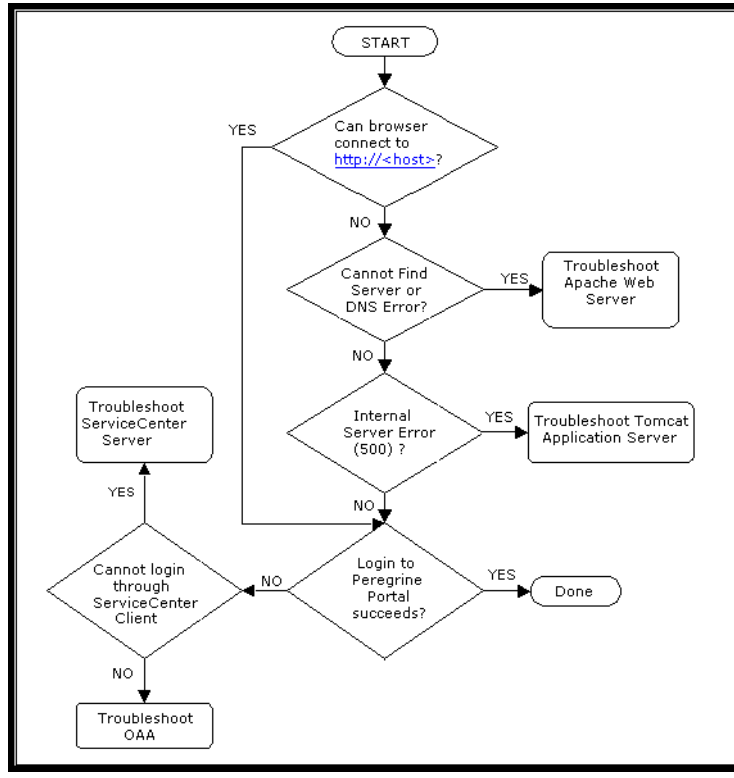
3 Click **Save** to return to the Control Panel.

4 Click **Reset Server** to save your changes.

7 Troubleshooting

CHAPTER

The following troubleshooting diagram shows areas to consider when trying to resolve installation problems.



This chapter covers the following topics:

Troubleshooting Apache Web server for Windows

If you are having trouble with the Apache Web server for Windows, follow these instructions.

The Web server is not responding

If the Web server is not responding:

Step 1 Verify that the network connections are enabled.

Step 2 Verify that the apache.exe program is running.

Step 3 Restart Apache service.

Step 4 Make sure the port that Apache uses is not in use by another network service (Apache uses port 80 by default).

To verify that the network connections are enabled:

1. Click Start.
2. Point to Settings.
3. Click Network and Dial-up connection.
4. Click Local area connection.
5. In the dialog box, verify that under Connection, Status is listed as Connected.

To verify that the apache.exe program is running:

1. Press Ctrl+Alt+Del.
2. Click Task Manager.
3. On the Processes tab, verify that the Apache.exe program is listed in the Image Name column.

To restart Apache service:

1. Click Start.
2. Click Programs.
3. Click Administrative Tools.
4. Click Services.
5. Locate the Apache service in the list and restart it.

To make sure the port that Apache uses is not in use by another network service (Apache uses port 80 by default):

1. Stop Apache.
   a. Click Start.
   b. Click Programs.
   c. Click Administrative Tools.
   d. Click Services.
   e. Locate the Apache service in the list and stop it.
2. Click Start.
3. Click Run.
4. Enter **cmd** and click OK.
5. In the command line window, enter **netstat -a** and press return.
6. Make sure that an entry with Proto=TCP, Local Address=<host>:http does not exist.

   Note: This ensures that when Apache is not running, no other service is listening on the http port (80).

7. Correct the problem by either changing Apache's default port (refer to customization documentation) or disabling/changing the conflicting service.

Users cannot access the Web server even though the server is running, and the network and Internet connections are enabled

If users cannot access the Web server, follow these instructions:

Step 1 Verify that the WINS server is installed.

Step 2 Verify that the DNS server is installed.

Step 3 Check Apache log files for additional errors.

To verify that the WINS server is installed:

1. Click Start.
2. Point to Settings.
3. Click Control Panel.
4. Click Add/Remove program.
5. Click Add/Remove Windows Components.
6. Click Networking Services.
7. Click Details.
8. Verify that the WINS Server check box is selected and properly configured on the network. Also verify that it is functioning.

To verify that the DNS server is installed:

1. Click Start.
2. Point to Settings.
3. Click Control Panel.
4. Click Add/Remove program.
5. Click Add/Remove Windows Components.
6. Click Networking Services.
7. Click Details.
8. Verify that DNS is installed, and that the DNS servers (or server) are connected and working on the network.

To view Apache log files for additional errors:

- From a text editor, open the Apache log files.

  The default files are in c:\Program Files\Peregrine\Common\Apache2\logs.

Troubleshooting Apache Web server for Unix

If you are having trouble with the Apache Web server for Unix, follow these instructions.

The Web server is not responding

If the Apache Web server is not responding, check the network setup.

To check the network setup:

1 Make sure the port that Apache uses is not in use by another network service.

Note: Apache uses port 80 by default. You can change this by using the **Port** directive in the **httpd.conf** file. Use the **netstat** command to list all ports being listened to after shutting down Apache.

```
$ /etc/init.d/oaactl stop
$ netstat -a | grep 80
```

2 Make sure the IP address and hostname of the server are configured correctly. If so,

- The Ping command successfully gets a response from the server.
- The **nslookup hostname** displays the correct mapping from the hostname to the IP address.
- The **telnet hostname 80** successfully connects to the server.

```
$ /usr/sbin/ping hostname -n 5

$ telnet hostname 80
Trying...
Connected to hostname
Escape character is '^]'.
```

View Apache log files for advanced errors

If you are having trouble with the Apache Web server, view the log files.

To view Apache log files for advanced errors:

▶ From a text editor, open the Apache log files.

The default Apache log files are in:
<base install directory>/peregrine/common/apache2/logs.

Troubleshooting Tomcat

Before you can troubleshoot problems on Tomcat, you must become familiar with starting and stopping Tomcat on your operating system. You also need know where the Tomcat log files are located.

To start/stop Tomcat on Windows

1. Click Start.
2. Click Programs.
3. Click Administrative Tools.
4. Click Services.
5. Locate the PeregrineTomcat service in the list and start/stop/restart it.

To start/stop Tomcat on UNIX

- $ /etc/init.d/oaactl <start/stop/restart>

The following table contains the default Tomcat log file locations.

Operating system	Default Tomcat log files location
Windows	C:\Program Files\Peregrine\Common\Tomcat4\logs
Unix	/<installed base directory>/peregrine/common/tomcat4/logs

Check for Tomcat port conflicts

The following table displays the default Tomcat port usage.

Port number	Tomcat service
8005	Tomcat Administration
8009	Tomcat AJP13 Worker Port

In the Tomcat log file stderr.log, the following line indicates the currently succeeded AJP13 port being used:

[INFO] ChannelSocket - -JK2: ajp13 listening on tcp port 8009

To check for Tomcat port conflicts:

1. Shutdown Tomcat.

2 Use **netstat -a** to list ports being listened on to make sure that there are no conflicts.

3 Make necessary modifications to Tomcat port configuration or disable (or modify) the conflicting service.

Check for Tomcat errors

Make sure that you are working with clean files.

To ensure a clean environment for troubleshooting:

1 Shutdown the Apache and Tomcat services.

2 Remove all log files.

3 Restart Apache and Tomcat.

4 Use a browser to connect to the Web server.

5 Stop.

File mod_jk.log

This file contains log information regarding the out-of-process TCP connection between the Apache Web server and Tomcat.

This file is empty when there are no errors. It contain hints about connection failures when the AJP13 port is in conflict with another service, or when the Tomcat mod_jk connector is configured incorrectly.

File stdout.log

The following is a normal output of this log file:

```
Bootstrap: Create Catalina server
Bootstrap: Starting service
Starting service Tomcat-Standalone
Apache Tomcat/4.1.12
Instantiating Archway Servlet...
2002-12-10 12:22:13,079 INFO  [main]  - Using application preferences in
/C:/Program Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/local.xml
2002-12-10 12:22:13,119 INFO  [main]  - Using default preferences in /C:/Program
Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/default/archway.xml
2002-12-10 12:22:13,200 INFO  [main]  - Using default preferences in /C:/Program
Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/default/common.xml
2002-12-10 12:22:13,240 INFO  [main]  - Using default preferences in /C:/Program
Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/default/logging.xml
2002-12-10 12:22:13,270 INFO  [main]  - Using default preferences in /C:/Program
Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/default/themes.xml
2002-12-10 12:22:13,280 INFO  [main]  - Using default preferences in /C:/Program
Files/Peregrine/Common/Tomcat4/webapps/oaa/WEB-INF/default/xsl.xml
Bootstrap: Service started
```

Look for the following in this file during an error:

- Archway Servlet is not instantiated.
- The webapps location is incorrect.
- Bootstrap service failed to start.

File stderr.log

The following is a normal output of this file:

```
Created catalinaLoader in: C:\Program Files\Peregrine\Common\Tomcat4\server\lib
[INFO] Registry - -Loading registry information
[INFO] Registry - -Creating new Registry instance
[INFO] Registry - -Creating MBeanServer
 [INFO] ChannelSocket - -JK2: ajp13 listening on tcp port 8009
[INFO] JkMain - -Jk running ID=0 time=0/120  config=C:\Program
Files\Peregrine\Common\Tomcat4\conf\jk2.properties
```

Look for the following problems in this file during an error:

- catalinaLoader was not created or is pointing to an incorrect location.
- ChannelSocket - JK2: ajp13 failed to connect or is connecting on an incorrect port number.
- JkMain is not using the right jk2.properties.

File localhost_log.<date>.txt

There should not be any Java errors in this log file. This file logs application manager activity in deploying Peregrine OAA Web applications.

Troubleshooting OAA

If you are having trouble with your Peregrine OAAWeb application, verify your application's back-end server and view the OAA logs.

OAA back-end configuration

Make sure that the Peregrine OAA application is connecting to the right back-end server and that it is currently functional.

To check back-end configuration:

1. Browse to http://hostname/oaa/admin.jsp.
2. Login as **Admin** and no password (providing this has not changed after installation).
3. From the Administration module, verify the connection status of the listed adapters.
4. Click on the target for the back-end server, for example, **sc**.
5. Verify that the host and port for the back-end server are correct.

OAA log files

The following table lists the default file locations of the Peregrine OAA log files.

Operating system	Default Peregrine OAA log files location
Windows	C:\Program Files\Peregrine\Common\Tomcat4\bin\archway.log
Unix	/<installed base directory>/peregrine/common/tomcat4/archway.log

Make sure that the log files contain:

- A listing of installed OAA components and their version numbers.
- A correct listing of registered packages.
- An **Archway initialization complete** statement.

If the file contains Java ClassNotFound exceptions, check to see if all the required jar files are found.

Troubleshooting ServiceCenter server

If you are having trouble with the ServiceCenter server:

Step 1 Check the ServiceCenter Auth code and port setting.

Step 2 Check the ServiceCenter log.

Before you troubleshoot problems, you must become familiar with starting and stopping the ServiceCenter server on your operating system. You also need to know where the ServiceCenter log files are located.

To start/stop ServiceCenter on Windows:

1. Click Start.
2. Click Programs.
3. Click Administrative Tools.
4. Click Services.
5. Locate the PeregrineServiceCenter service in the list and start/stop/restart it.

To start/stop ServiceCenter on UNIX:

- $ /etc/init.d/oaactl <start/stop/restart>

The following table contains the default ServiceCenter log file locations.

Operating system	Default ServiceCenter log files location
Windows	C:\Program Files\Peregrine\ServiceCenter\sc.log
Unix	/<installed base directory>/peregrine/servicecenter/sc.log

Check ServiceCenter Auth code and port setting

The following table contains the ServiceCenter setting file location.

Operating system	ServiceCenter setting file location
Windows	C:\Program Files\Peregrine\ServiceCenter\RUN\sc.ini
Unix	/<installed base directory>/peregrine/servicecenter/RUN//sc.ini

To check the ServiceCenter Auth code and port setting:

1. Make sure the auth code set by the **auth:** tag is correct.
2. Make sure the port setting for **system:** matches the setting for the OAA back-end.

View ServiceCenter log

To view the ServiceCenter log:

1. View the log file for auth code expiration errors.
2. View the log for resource attachment errors.
3. Refer to *ServiceCenter Administration Guide* for further troubleshooting if required.

Index