Get-Resources 4.1

Installation Guide

For Windows, AIX, Linux, and Solaris

Contents

About this Guide

Get-Resources, part of the Get-It suite of Employee Self-Service products from Peregrine Systems, provides a way for businesses to make the procurement of resources available to employees through the corporate intranet.

Get-Resources integrates with AssetCenter Portfolio, AssetCenter Procurement, or ServiceCenter Request Management to enable employees to create requests for resources and to streamline the approval workflow of those requests throughout the organization.

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

- Install the Peregrine OAA Platform and Get-Resources.
- Configure AssetCenter or ServiceCenter for use as a back-end system with Get-Resources.
- Utilize the Peregrine Portal Administration module to connect to the back-end system and configure Get-Resources settings.

Book audience

This guide is for Get-Resources 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 (if you use the tailoring kit)
- Operating guides, reference manuals, and other documentation for your PC hardware and operating system
- Web server and Application server administration
- AssetCenter administration and functionality
- ServiceCenter administration and functionality

Related documentation

Refer to the following documentation for additional information:

- *Get-Resources Administration Guide* describes the Peregrine OAA platform and Get-Resources administration.
- *Get-Resources Release Notes* covers any last-minute documentation or known issues with Get-Resources. These documents are constantly updated and posted to the Customer Support web site. See *Need further assistance?* on page 13 for details on accessing the Customer Support website.

Associated applications

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

Note: A back-end system, either AssetCenter or ServiceCenter, must be installed and configured before you can install and configure Get-Resources. Peregrine OAA installs with Get-Resources, and only the installation of Peregrine OAA for Get-Resources is included in this guide.

Terminology

The terminology used in this guide and in the Get-Resources interface is based on AssetCenter 3.6 and 4.x, and 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

Element	Usage
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, Introduction	Installation requirements and description of different types of installations for Get-Resources.
Chapter 2, Installation for Windows Platforms	Detailed instructions for installing Get-Resources in a Windows environment.
Chapter 3, Installation for UNIX Platforms	Detailed instructions for installing Get-Resources in a Unix environment.
Chapter 4, Load Balancing	Information on load balancing the application server for optimum system memory management.
Chapter 5, Back-end System Configuration	Detailed instructions for configuring ServiceCenter or AssetCenter as a back-end system with Get-Resources.
Chapter 6, Get-Resources Configuration	Final configuration settings within the Peregrine Portal Administration module to complete the installation of Get-Resources.
Chapter 7, Troubleshooting	Troubleshooting installation problems with Apache Web server, Tomcat, OAA, and AssetCenter and ServiceCenter.

Need further assistance?

For further information and assistance with this release, you can download documentation or schedule training.

Customer Support

For further information and assistance, contact Peregrine Systems' Customer Support at the Peregrine CenterPoint Web site.

To contact customer support:

1 In a browser, navigate to http://support.peregrine.com

2 Log in with your user name and password.

3 Follow the directions on the site to find your answer. The first place to search is the KnowledgeBase, which contains informational articles about all categories of Peregrine products.

4 If the KnowledgeBase does not contain an article that addresses your concerns, you can search for information by product; search discussion forums; and search for product downloads.

Documentation Web site

For a complete listing of current Get-Resources documentation, see the Documentation pages on the Peregrine Customer Support Web.

To view the document listing:

1 In a browser, navigate to http://support.peregrine.com.

2 Log in with your login user name and password.

3 Click either **Documentation** or **Release Notes** at the top of the page.

4 Click the Get-Resources link.

5 Click a product version link to display a list of documents that are available for that version of Get-Resources.

6 Documents may be available in multiple languages. Click the Download button to download the PDF file in the language you prefer.

You can view PDF files using Acrobat Reader, which is available on the Customer Support Web site and through Adobe at http://www.adobe.com.

> **Important:** Release Notes for this product are continually updated after each release of the product. Ensure that you have the most current version of the Release Notes.

Education Services Web Site

Peregrine Systems offers classroom training anywhere in the world, as well as "at your desk" training via the Internet. For a complete listing of Peregrine's training courses, refer to the following web site:

http://www.peregrine.com/education

You can also call Peregrine Education Services at +1 858.794.5009.

1 Introduction

CHAPTER

This chapter covers the following topics for Get-Resources:

Installation requirements

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

Component	for Windows	for Unix
Java run-time environment	■ Java 2 SDK Standard Edition v1.3.1_05. *Available on the Get-Resources Installation CD.*	■ Java 2 SDK Standard Edition v1.3.1_05. *Available on the Get-Resources Installation CD.*
Application server	Any one of the following: ■ Tomcat 4.1.24 *Available on the Get-Resources Installation CD.* ■ WebSphere Application Server 4.0, 5.0 ■ WebLogic 6.1 SP3 ■ JRun 3.1	Any one of the following: ■ Tomcat 4.1.24 *Available on the Get-Resources Installation CD.* ■ WebSphere Application Server 4.0, 5.0 ■ WebLogic 6.1 SP3 ■ JRun 3.1
Back-end database(s)	■ ServiceCenter 4.0.x, 5.0.x, 5.1 ■ AssetCenter 3.6, 4.1, 4.2, 4.3 ■	■ ServiceCenter 4.0.x, 5.0.x, 5.1 ■ AssetCenter 3.6, 4.1, or 4.2, 4.3 ■
Operating systems	■ Windows 2000 Server SP2 ■ AIX 5.1, 5.2 ■ Solaris 2.6, 7, 8, 9 ■ Redhat Linux 7.3, 8.0, 9.0	■ AIX 5.1 ■ Red Hat Linux 7.3 ■ Solaris 2.7 or Solaris 2.8

Component	for Windows	for Unix
Web Server	One of the following: ■ Apache 2.0.43 *Available on the Get-Resources 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-Resources 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	■ Linux: Pentium, 400 MHz or faster ■ AIX: POWER 3, 375 MHz or faster ■ Solaris: Ultra SPARC II, 300 MHz or faster
RAM	■ 512 MB or more	■ 512 MB or more
Hard disk space	■ 100 MB for Get-Resources	■ 100 MB for Get-Resources

Note: The AssetCenter API must be installed on the same system as your Web application. However, the AssetCenter database can be on another system. This means that when you install AssetCenter on the Get-Resources server, you must either use a full installation or a custom installation where you select the "AssetCenter API" option.

Types of installations

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

■ Typical installation

■ Custom installation

A *typical installation* installs Tomcat as the Application server and Apache as the Web server with Get-Resources, 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-Resources 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-Resources implementers to review application functionality and test customizations before deploying to a production environment. By default, the Get-Resources installer uses the development environment installation.

Tip: Install the same Web server and application server in the development environment that you plan to use in the production environment.

A *production environment* installation is optimized for performance and scalability. Each component, such as an application server and Web server, is installed on different servers. In addition, there may be multiple instances of any component. Given the amount of flexibility involved in such an installation, users must manually set up a production environment.

Back-end systems

Important: In order to use Get-Resources, you must have a properly configured back-end system.

Get-Resources uses the back-end system to:

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

Refer to *Installation requirements* on page 16 for a complete list of the versions compatible with Get-Resources.

2 Installation for Windows Platforms

CHAPTER

This chapter covers the following topics:

Choosing an installation environment

You can install Get-Resources in one of two environments:

- Development environment
- Production environment

Development environment

The Get-Resources 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-Resources on one computer system.

You have two choices of development environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Resources deployed on Tomcat 4.1.24 application server
- Custom installation
  - Choice of Web server
  - Get-Resources deployed on your choice of application server

Production environment

The Get-Resources 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-Resources on different servers to maximize performance.

You have two choices of production environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Resources deployed on multiple instances of Tomcat 4.1.24 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Resources

Development Environment

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

To install Get-Resources in a typical development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Verify that the back-end database required for Get-Resources is installed.

Step 3 Run the Get-Resources installer, follow any upgrade prompts, and then select the Typical installation option. See *Typical installation option* on page 73.

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

To install Get-Resources in a custom development environment:

Step 1 Acquire all necessary hardware and software.

Step 2 Verify that the back-end database required for Get-Resources is installed.

Step 3 Install alternate application and Web servers.

Step 4 Configure the alternate application server for Get-Resources.

Step 5 Run the Get-Resources installer, follow any upgrade prompts, and then select the Custom installation option. See *Custom installation option* on page 81.

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

Production Environment

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

To install Get-Resources in a typical production environment:

Step 1 Acquire all necessary hardware and software.

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

Step 3 Run the Get-Resources installer, follow any upgrade prompts, and then select the Typical installation option. See *Typical installation option* on page 73.

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-Resources users.

To install Get-Resources in a custom production environment:

Step 1 Acquire all necessary hardware and software.

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

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

Step 4 Configure the alternate application server for Get-Resources.

Step 5 Run the Get-Resources installer, follow any upgrade prompts, and then select the Custom installation option. See *Custom installation option* on page 81.

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

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

Migrating Get-Resources from previous versions

To migrate older versions of Get-It or Get-Resources to Get-Resources 4.1 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-Resources and back-end system data prior to performing any steps for migration.

To migrate previous versions to Get-Resources 4.1:

Step 1 Review the customizations of previous version and determine which customizations need to be recreated in Get-Resources 4.1. See *Recreating customizations in Get-Resources 4.1* on page 22.

Step 2 Install Get-Resources 4.1 on a new system. See *Choosing an installation environment* on page 19.

Step 3 Apply any required configuration changes to the back-end database you want to migrate to Get-Resources 4.1. See *Configuring a Back-end Database* in this guide.

Recreating customizations in Get-Resources 4.1

You cannot directly migrate customizations implemented before version 4.0.1 to Get-Resources 4.1. Instead, you must recreate your changes using the new features and methods available in Get-Resources 4.1.

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

No customizations

If you have made no customizations to Get-Resources, you can simply install Get-Resources 4.1 on a new system and migrate your data from your existing back-end database.

Note: Get-Resources 4.1 requires ServiceCenter 4.0.x or higher.

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-Resources 4.1 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-Resources 4.1 using DocExplorer. Refer to the *Get-Resources Administration Guide* for information on personalization.

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-Resources 4.1 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-Resources 4.1, 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-Resources 4.1 version of the classic theme as your template. Refer to the *Get-Resources Administration Guide* for information on customizing.

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-Resources 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-Resources 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 from a form	Add or remove fields from Personalization
Added a new language or locale to the Get-Resources 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 must recreate them in the current version of the Get-Resources tailoring kit.

Configuring an existing back-end database for Get-Resources 4.1

The following table lists the options available for data migration.

Get-Resources 2.5 to Get-Resources 4.1

Back-end version	Migration required
AssetCenter 3.6	Apply Get-Resources 4.1 scripts to existing AssetCenter 3.6.
AssetCenter 4.x	Apply Get-Resources 4.1 scripts to existing AssetCenter 4.x.
ServiceCenter 3.0	Upgrade to ServiceCenter 4.x or 5.0.x.
ServiceCenter 4.x	Apply Get-Resources 4.1 unload files to the existing ServiceCenter 4.x and apply the unload files under the upgrade folder.

Get-Resources 4.0.1 to Get-Resources 4.1

Back-end version	Migration required
ServiceCenter 4.x	Apply all unload files delivered with Get-Resources 4.1, including the file under the upgrade folder. **Note:** Delete the old unload files (scgr.unl, scgrdata.unl, scrmdemo.unl, 201patch6.unl).
ServiceCenter 5.0.x	Apply all unload files delivered with Get-Resources 4.1 for ServiceCenter 5.0.

Upgrading from Get-Resources 4.0.1 to 4.1

To upgrade from Get-Resources 4.0.1 to Get-Resources 4.1, you simply need to run the 4.1 installer. See either *Typical installation option* or *Custom installation option* for more information.

Note: Note that the **<appsrv>\WEB-INF\web.xml** file has been renamed to **<appsrv>\WEB-INF\web.xml.XXX.bak**, where *<appsrv>* is the path to your application server, and **XXX** represents a unique sequence of characters generated by File.createTempFile(). This will preserve any customizations that you might have (see *Preserving customized web.xml file settings*, next).

After upgrading Get-Resources from 4.0.1 to 4.1, a new file called web.xml.xxx.bak is created in the *<appsrv>*/WEB-INF directory. You must reconcile the contents of this file with the contents of the new web.xml file. Any portion of web.xml.xxx.bak file that does not exist in the new web.xml file needs to be added to the web.xml file.

Preserving customized web.xml file settings

You can open the web.xml.xxx.bak file, copy its customized configuration settings into the new web.xml file, and save the new file to preserve your customizations.

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:

- *Existing Tomcat and Apache servers*
- *Tomcat 4.1.24 connecting to IIS 5.0*
- *WebSphere 4.0.2*
- *WebSphere 5.0*
- *WebLogic 6.1 SP3 or SP4*
- *JRun 3.1*

The Get-Resources typical installation option installs Tomcat 4.1.24 and connects it to an Apache 2.0 web server. You can also install Tomcat 4.1.24 using the custom installation option.

> **Important:** If you want to use an application server other than Tomcat
> 4.1.24, then you must configure your application and Web
> servers *prior* to running the Get-Resources installer.

Existing Tomcat and Apache servers

If you use the typical installation option, the Get-Resources installer
configures Tomcat to connect to a new instance of the Apache Web server. If
you have existing instances of Tomcat or Apache Web Server installed, you
can configure Get-Resources to use these existing instances by copying the
necessary files from a typical installation.

To configure an existing Tomcat server to connect to an Apache server:

1 Stop existing instances of both Tomcat and Apache.

2 Copy the following files from the installation CD **\SupportFiles**… directory
to the directories indicated below.

Copy this file	To the following location
■ mod_jk.conf	The **\conf** directory of your existing Tomcat installation. The default source file path is: **C:\Program Files\Apache Group\Tomcat 4.1\conf**
■ workers.properties	The **\conf** directory of your existing Tomcat installation. The default source file path is: **C:\Program Files\Apache Group\Tomcat 4.1\conf**
■ mod_jk.dll	The **\modules** directory of your existing Apache installation. The default source file path is: **C:\Program Files\Apache Group\Apache2\modules**

Note: The mod_jk.dll included with this release is compatible with
Apache 2.0.43 and Tomcat 4.1.25. If you are using other versions, refer
to the jakarta.apache.org/builds/jakarta-tomcat-connectors/jk/doc
site to download the compatible version.

3 Using a text editor, open the files mod_jk.conf and workers.properties.
These files are located in the **\conf** directory of your Tomcat installation.

   a Find all instances where the path to Tomcat appears and edit these to
   reflect your current Tomcat 4.1 installation path.

   b Find all instances where the path to JDK appears and edit these to reflect
   your current JDK installation path.

4 Using a text editor, open the httpd.conf file. This file is located in the \conf directory of your Apache installation.

   a Add the path to your existing Tomcat installation to the include statement in the Global Environment section:

   ```
   ### Section 1: Global Environment
   ...
   include "<Tomcat_path>/conf/mod_jk.conf"
   ```

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

   b Add login.jsp to the list of files in the DirectoryIndex section:

   ```
   # DirectoryIndex: Name of the file or files to use as a pre-written
   # HTML directory index.  Separate multiple entries with spaces.
   #
   <IfModule mod_dir.c>
    DirectoryIndex index.html login.jsp
   </IfModule>
   ```

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

   ```
   Alias <Tomcat>\webapps\oaa
   ```

   where *<Tomcat>* is the path to your Tomcat installation.

5 Install Get-Resources using the **Custom** option. See *Custom installation option* on page 81.

6 Restart Tomcat and Apache.

7 Browse to the Get-Resources login URL and verify that you can successfully connect.

   Note: Depending on your Web server configuration, if you browse to http://servername/oaa, the Web server may display a list of all the OAA files instead of the login page.

   If your server displays this behavior, follow these steps to configure your Web server to display the OAA login page instead of a directory listing.

   To configure Apache to display login.jsp by default:

1 Open Apache's conf/httpd.conf file in a text editor.

2 Find the existing line that reads DirectoryIndex index.html.

3 Add login.jsp to the end:

   DirectoryIndex index.html login.jsp

4 Save httpd.conf.

5 Restart the Apache Web server.

Tomcat 4.1.24 connecting to IIS 5.0

You can use the Get-Resources installer to install the Tomcat application server. If you use the typical installation option, the Get-Resources 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.

Note: These instructions are for setting up Tomcat to use a single Java Virtual Machine (JVM). See the *Installation Guide* chapter on *Load Balancing* for installing multiple JVMs.

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

Step 1 Run the Get-Resources installer. See *Running the installer* on page 29.

Step 2 Configure the ISAPI Plug-in for IIS. See *Configuring the ISAPI Plugin for IIS* on page 30.

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 30.

Step 4 Create and configure a jakarta virtual directory in IIS. See *Configuring a jakarta virtual directory in IIS* on page 31.

Step 5 Create and configure an oaa virtual directory in IIS. See *Configuring an oaa virtual directory in IIS* on page 32.

Step 6 Edit the server.xml file to add performance settings and configure alternate communications ports (Optional). See *Editing the server.xml file for IIS* on page 33.

Step 7 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 35.

Running the installer

Run the Get-Resources installer and select the Custom installation option. See *Custom installation option* on page 81.

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-Resources installer automatically places a copy of the ISAPI plug-in 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. The default file path is:

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"
```

Tip: You do not need to make any changes if you installed this file to the default location.

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 From Windows Control Panel > Administrative Tools, 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. The default file path is:

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

6 Click **OK**.

   Note: You must stop and then start the IIS service for changes to take effect. You must also restart Peregrine Tomcat.

7 From the Internet Services management console, right-click the **Default Web Site** node, then select **Properties**>**Isapi Filters** again.

   The ISAPI filter in IIS displays a green status arrow to indicate that it is running.

8 Close the Internet Services management console.

Configuring a jakarta virtual directory in IIS

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

To configure a jakarta virtual directory in IIS:

1 Use the following guidelines to create the virtual directory on the Default Web Site.

Requirements for a jakarta virtual directory

Requirement	Setting
Create virtual directory	jakarta
Map to physical path	<Tomcat>\bin
Directory access rights	Read, Run scripts, Execute

2 For <Tomcat>, enter the path to your Tomcat installation. The default file path is:

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

Configuring an oaa virtual directory in IIS

To run Get-Resources from IIS, you need to create a virtual directory that maps to your Tomcat deployment folder. For specific instructions about configuring IIS, refer to your Windows Help.

To configure an oaa virtual directory in IIS:

▶ Use the following guidelines to create the virtual directory.

Requirements for an oaa virtual directory

Requirement	Setting
Create virtual directory	`<oaa>`
Map to physical path	`<Tomcat>\webapps\<oaa>`
Directory access rights	Read, Run scripts

For *<oaa>*, enter the name of the virtual directory you want to use for Get-Resources. The recommended virtual directory name is oaa. If you choose to use another virtual directory name, you must enter the new name in the following places:

- Rename the folder `<Tomcat>\webapps\oaa` to `<Tomcat>\webapps\<new name>`
- Rename the [uri] mappings in `workers2.properties` from oaa to the new virtual directory name.
- Rename all the oaa context entries in `mod_jk2.conf` from oaa to the new virtual directory name.
- Rename the `<Context>` path and docBase attributes in `server.xml` from oaa to the new virtual directory name.

Important: The virtual directory name you choose will become part of the URL users enter to connect to Get-Resources. For example: `http://server_name/<new name>/login.jsp`

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

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

Note: Depending on your Web server configuration, if you browse to http://servername/oaa, the Web server may display a list of all the OAA files instead of the login page.

If your server displays this behavior, follow these steps to configure your Web server to display the OAA login page instead of a directory listing.

To configure IIS:

1 Open the Internet Services Manager.

2 Expand the Default Web Site.

3 Right-click on the **OAA** virtual directory and click Properties.

4 Click the Documents tab.

5 Verify that **Enable Default Document** is checked.

6 Click the **Add** button.

    a Type login.htm.

    b Click **OK**.

7 Highlight login.htm and using the up and down arrows, move login.htm to the top of the file list.

8 Click **OK** to accept the changes to the OAA directory properties.

Editing the server.xml file for IIS

A default Tomcat installation is sufficient for most Get-Resources installations. However, if you are experiencing performance problems or communications port conflicts, you may need to edit the Tomcat server.xml file to correct these problems.

Performance settings

The Tomcat server.xml file allows you to determine how Tomcat processes Get-Resources files. If you are experiencing performance problems, you can change the <Context> setting for Get-Resources to disable page reloading.

Tip: Make a back up copy of the server.xml file before editing.

To edit the server.xml performance settings:

1 Open the file server.xml in any text editor. The default file path is:

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

2 Create a <Context> element entry from Tomcat to the Get-Resources deployment directory to establish a point of reference for docBase.

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

Example:

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

Setting the reloadable attribute to false results in faster JSP page processing.

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

Communications port settings

If your Get-Resources server already uses communications ports 8005 and 8009, you will have a port conflict if you install Tomcat with the default settings. To avoid a port conflict, you must edit the server.xml file to change the communications ports used by Tomcat.

Important: You do not need to perform these optional steps if Tomcat's default communication ports are available on your server.

Tip: Make a back up copy of the server.xml file before editing.

To edit the server.xml communications port settings:

1 Open the file server.xml in any text editor. The default file path is:

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

2 Update the port number attribute of the <Server> element to a free communications port.

Note: By default, Tomcat uses port 8005 for shutdown requests.

Example:

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

3 Update the port attribute of the Coyote Connector <Connector> element to a free communications port.

Note: By default, Tomcat uses port 8009 for 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 Save the server.xml file.

5 Restart Tomcat for your new settings to take effect.

Installing Tomcat as a service

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

Note: The installer does not reset the JAVA_HOME environment variable when installing on systems where a previous instance of Tomcat is installed. Manually redefine the JAVA_HOME environment variable to point to the new Java Development Kit. The default path is: C:\Program Files\Peregrine\Common\jdk1.3.1_05

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.

WebSphere 4.0.2

Use the following procedures to configure WebSphere 4.0.2 to run Get-Resources on Windows.

Note: The Get-Resources installer creates duplicate alias entries in the IBM HTTP Server when you install more than one Peregrine OAA Platform application on WebSphere.

Duplicate entries can also occur if you reinstall Get-Resources or install another Peregrine OAA Platform application on a system that formerly had Get-Resources installed on it.

Remove any duplicate alias entries from the IBM HTTP Server httpd.conf file.

To configure WebSphere 4.02:

Step 1 Install WebSphere 4.02. Your version of WebSphere 4.0.2 includes the IBM HTTP Server. See *Installing WebSphere 4.0.2* on page 36.

Step 2 Deploy the Portal WAR file to WebSphere to create the necessary folder structure for Get-Resources. See *Deploying the Portal WAR file to WebSphere* on page 37.

Step 3 Set the JVM Java heap size for each WebSphere instance running Get-Resources. See *Setting the Java heap size* on page 39.

Step 4 Create the virtual directory you want to use for Get-Resources in your Web server. See *Configuring a virtual directory for IBM HTTP Server* on page 41.

Step 5 Run the Get-Resources installer. See *Running the installer* on page 41.

Step 6 Regenerate and configure. See *Regenerating the plug-in configuration* on page 41.

If you plan on setting up a WebSphere Portal Server or a WebSphere Translation Server, see *Installing WebSphere Portal Server* on page 45 or *Configuring WebSphere Translation Server for Get-Resources* on page 55.

Installing WebSphere 4.0.2

Purchase and install IBM WebSphere 4.0.2. Your version of WebSphere 4.0.2 includes the IBM HTTP Server.

Verify that you install fix pack 2. To check this, go to the default_server_Stdout.log file under \Websphere\AppServer\logs.

Deploying the Portal WAR file to WebSphere

The Portal WAR file creates the folder structure necessary to deploy Get-Resources in your application server. After you have deployed this file to WebSphere you will be ready to run the Get-Resources 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 file path 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-Resources 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.
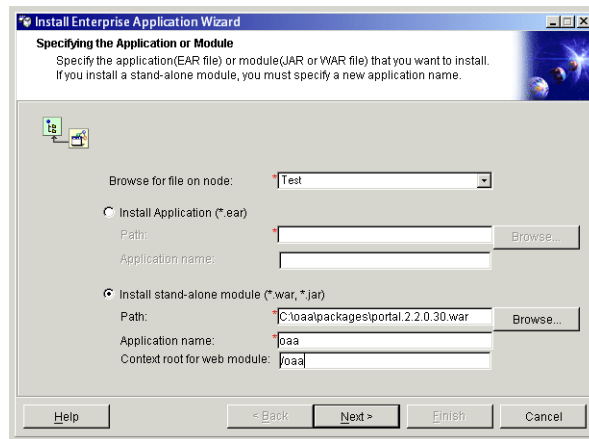
To set the Java heap size:

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-Resources 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. The default file path is:

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.

Running the installer

Run the Get-Resources installer and select the Custom installation option. See *Custom installation option* on page 81.

Regenerating the plug-in configuration

You must regenerate the plug-in configuration using the WebSphere Admin console after running the Get-Resources installer.

To regenerate the plug-in configuration:

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

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

The server settings page opens.

Right-click on your system name and select Regen Webserver Plugin.
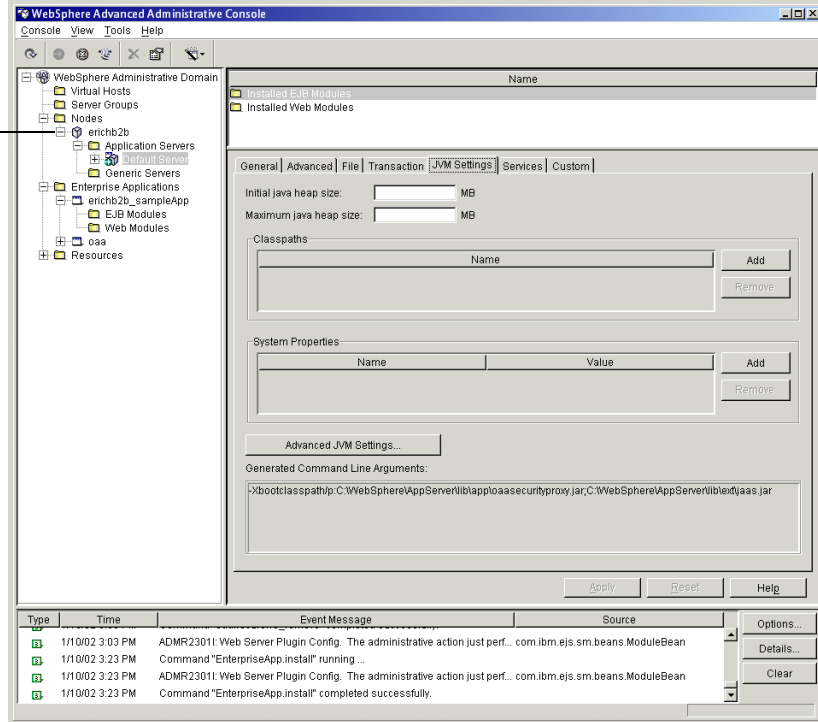
3 Right-click on the *<System Name>*, then select **Regen Webserver Plugin**.

4 Copy the following lines from the <settings> section of **<appserver>\WEB-INF\default\archway.xml** to the <settings> section in your **<appserver>\WEB-INF\local.xml** file, where *<appserver>* is the location of your application server:

<SSLProvider>com.ibm.jsse.JSSEProvider</SSLProvider>
<HTTPSHandlerPkg>com.ibm.net.ssl.internal.www.protocol</HTTPSHandlerPkg>
<CryptoProvider>com.ibm.crypto.provider.IBMJCE</CryptoProvider>

5 Restart your application server.

WebSphere 5.0

Use the following procedures to configure WebSphere 5.0 to run Get-Resources on Windows. WebSphere Application Server 5.0 is referred to as WAS5.

Note: The Get-Resources installer creates duplicate alias entries in the IBM HTTP Server when you install more than one Peregrine OAA Platform application on WebSphere.

Duplicate entries can also occur if you reinstall Get-Resources or install another Peregrine OAA Platform application on a system that formerly had Get-Resources installed on it.

Remove any duplicate alias entries from the IBM HTTP Server httpd.conf file.

To run Get-Resources 4.x on WebSphere Application Server 5.0:

1 Set the WAS_HOME environment variable to your WAS5 home directory.

The default location is: C:\Program Files\WebSphere\AppServer.

2 With WAS5 running, log in to the Admin console and create a new Enterprise Application using portal.war from the packages directory on the Get-Resources 4.0.1 CD.

Note: The important option to specify is the context root, typically /oaa or /getit.

   a Specify the context root.
   b Keep all the other settings at their defaults.
   c Save the server configuration.

3 Run setup from the CD.

   a Choose the Custom install.
   b Clear the Tomcat, JDK, and Apache options.

4 When prompted for the deployment directory, browse to it under the WAS5 installed applications directory; typically, C:\Program Files\WebSphere\AppServer\installedApps\[hostname]\oaa.ear\portal.war.

5 When the installation is complete, delete pop3.jar from C:\Program Files\WebSphere\AppServer\java\jre\lib\ext.

It is no longer needed because WAS5 includes it in mail.jar.

6 Copy js.jar from …portal.war\WEB-INF\lib to
C:\Program Files\WebSphere\AppServer\java\jre\lib\ext.

7 From the IBMHttpServer\conf\httpd.conf file:

a Add the following lines.

```
LoadModule ibm_app_server_http_module
"<AppServerPath>/bin/mod_ibm_app_server_http.dll"
WebSpherePluginConfig
"<AppServerPath>/config/cells/plugin-cfg.xml"
```

b Verify that the following line is in the file; if not, add it.

```
Alias /oaa/ "opt/WebSphere/AppServer/installedApps/[hostname]
/oaa.ear/portal.war/"
```

The installer fails to do this because earlier versions were named differently.

Note: This alias needs to match the context root specified in step 2. The
portal.war deployment folder has the same name as the portal.war file
you deployed in step 2.

8 Using a text editor:

a Add the following UriGroup mappings to the plugin-cfg.xml file located in
WebSphere\AppServer\config\cells.

```
 <UriGroup Name="default_host_server1_sys_Cluster_URIs">
 :
 :
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/archway"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/rpcrouter"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/messagerouter"/
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/download/*"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/*"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.do"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsp"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsv"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsw"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/j_security_check"/>
 </UriGroup>
```

b Save and close the file.

9 Restart WAS5.

10 Log in to the Admin console again.

   a From Environment on the left side, click **Update Web Server Plugin**.

   b Click **OK**.

11 Restart the IBM HTTP Server.

12 Log in to admin.jsp and continue configuring the system as usual.

Installing WebSphere Portal Server

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

■ All Get-Resources and WebSphere components running on a single system. See *Recommended WebSphere Portal Server configuration* on page 45.

■ Get-Resources components running on one system and WebSphere components running on another. See *Alternate WebSphere Portal Server configuration* on page 47.

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-Resources for the recommended WebSphere Portal Server configuration:

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 49.

Step 2 Generate a Get-Resources WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Resources WAR file* on page 50.

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

Step 4 Modify the local.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the local.xml file* on page 51.

Step 5 Modify the web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 51.

Step 6 Modify the ibm-web-ext.xmi file to set the fileServingEnabled parameter. See *Modifying the ibm-web-ext.xmi file* on page 52.

Step 7 Start the WebSphere application server. See *Starting the WebSphere application server* on page 52.

Step 8 Deploy the Get-Resources WAR file to WebSphere Portal Server. See *Deploying the Get-Resources WAR file to WebSphere Portal Server* on page 53.

Step 9 Create places and pages in WebSphere Portal Server to display Get-Resources portlets. See *Configuring WebSphere Portal Server places and pages* on page 53.

Step 10 Enable edit rights for Get-Resources portlets. See *Enabling edit rights for Get-Resources portlets* on page 54.

When complete, your installation will have the following configuration:
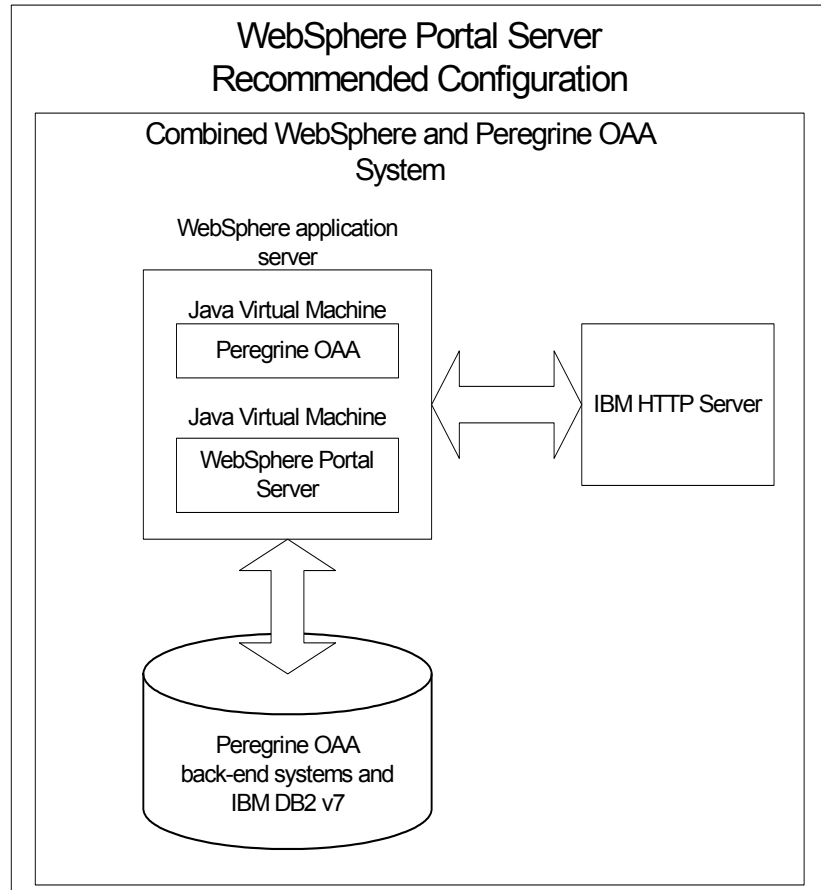

Alternate WebSphere Portal Server configuration

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

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 49.

Step 2 Generate a Get-Resources WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Resources WAR file* on page 50.

Step 3 Login to the Get-Resources server and stop the WebSphere application server. See *Stopping the WebSphere application server* on page 51.

Step 4 Modify local.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the local.xml file* on page 51.

Step 5 Modify web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 51.

Step 6 Modify the ibm-web-ext.xmi file to set the fileServingEnabled parameter. See *Modifying the ibm-web-ext.xmi file* on page 52.

Step 7 Modify setDomain.js to call the SetDomain function. See *Modifying the setDomain.js file* on page 52.

Step 8 Start the WebSphere application server. See *Starting the WebSphere application server* on page 52.

Step 9 Deploy the Get-Resources WAR file to WebSphere Portal Server. See *Deploying the Get-Resources WAR file to WebSphere Portal Server* on page 53.

Step 10 Create places and pages in WebSphere Portal Server to display Get-Resources portlets. See *Configuring WebSphere Portal Server places and pages* on page 53.

Step 11 Enable edit rights for Get-Resources portlets. See *Enabling edit rights for Get-Resources portlets* on page 54.

Step 12 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 54.

When complete, your installation will have the following configuration:


```
                  WebSphere Portal Server
                  Alternate Configuration

  WebSphere System

         WebSphere application
                server
       Java Virtual Machine
        WebSphere Portal     <───>    IBM HTTP Server
             Server


                IBM DB2 v7


  ────── Forward and reverse proxy settings ──────

  Peregrine OAA System

       Tomcat or other application
                server
       Java Virtual Machine
          Peregrine OAA      <───>    Apache Web
                                        Server


            Peregrine OAA
           back-end systems
```
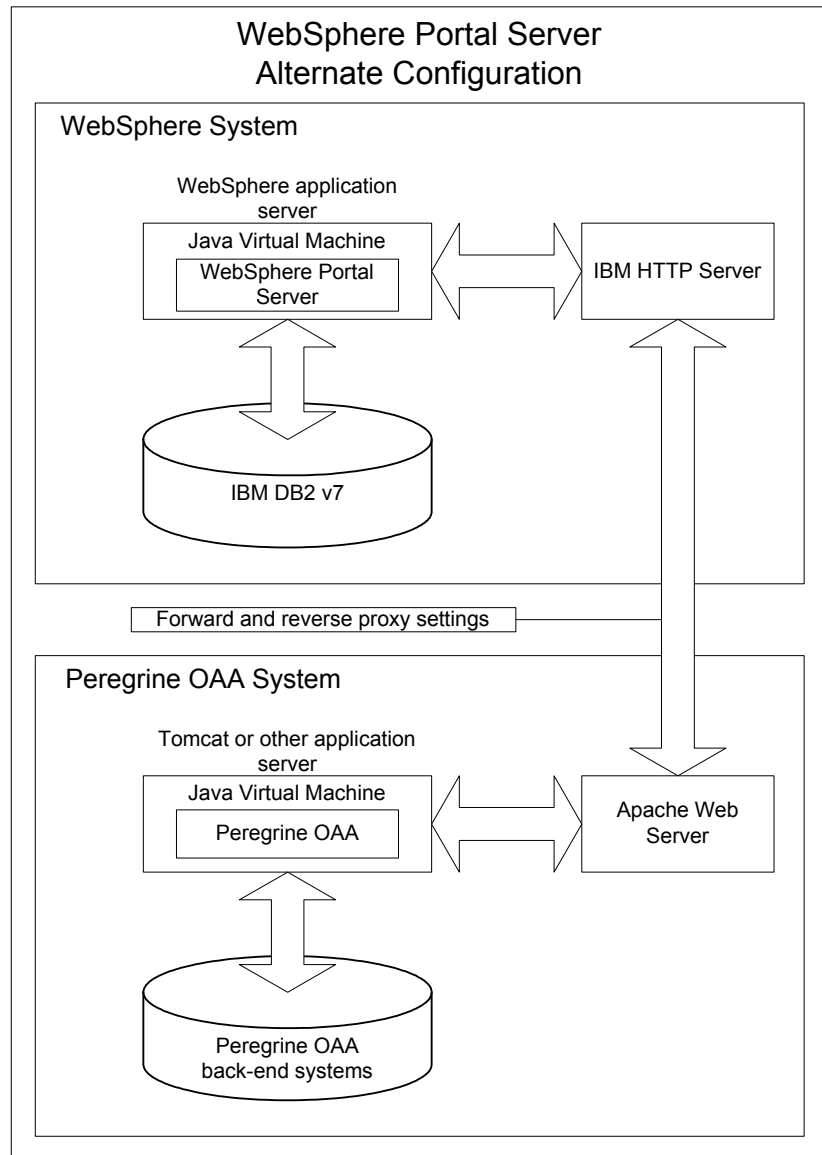
WebSphere Portal Server installation requirements

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-Resources 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-Resources compatible application server
    - Web server
    - Back-end database for Get-Resources
    - An installation of Get-Resources

Generating a Get-Resources WAR file

In order to display Get-Resources in WebSphere Portal Server, you must first export the Get-Resources 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-Resources WAR file:

1 Login to the Get-Resources 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-Resources `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-Resources WAR file.

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

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

4 Click **Generate WAR file.**

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

Stopping the WebSphere application server

Login to the Get-Resources server and stop the WebSphere application server before modifying the configuration.

Modifying the local.xml file

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

To modify the local.xml file:

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

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

2 Add the following on a separate line anywhere between <settings> and <\settings>:

```
<httpauthclass>HttpAlternateAuthenticationManager</httpauthclass>
```

3 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>\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 ibm-web-ext.xmi file

You need to set the fileServingEnabled parameter to true to handle static content.

To modify the ibm-web-ext.xmi file:

1 Using a text editor, open the ibm-web-ext.xmi file. The default file path is:

```
c:\WebSphere\AppServer\installedApps\getit.ear\getit.war\WEB-INF
```

2 Find the fileServingEnabled parameter and set it to **true**.

```
fileServingEnabled="true"
```

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-Resources server.

2 Stop your application server.

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

<application server>\oaa\js.

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

```
setDomain();
```

5 Save the file.

Starting the WebSphere application server

Start the WebSphere application server for changes to take effect.

Deploying the Get-Resources WAR file to WebSphere Portal Server

After you deploy the Get-Resources 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-Resources 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-Resources WAR file.

5 Click Next to load the Get-Resources 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

Note: Refer to your WebSphere Portal documentation for details on places and pages.

You can deploy Get-Resources 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-Resources portlets that you display in a page must grant "all authenticated users" the minimum edit permission.

Enabling edit rights for Get-Resources portlets

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

To enable edit rights for Get-Resources 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-Resources 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-Resources.

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-Resources 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-Resources is installed.

5 Save the file.

Configuring WebSphere Translation Server for Get-Resources

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

Note: The OAA interface to the WebSphere Translation Server requires a mouse to use. The translation interface will be made 508 accessible in a future release.

To configure WebSphere Translation Server for Get-Resources:

Step 1 Copy the file wts.jar to the Get-Resources deployment folder. See *Copying wts.jar to the Get-Resources deployment folder* on page 55.

Step 2 Configure Get-Resources to use the WebSphere Translation Server. See *Configuring Get-Resources to use the WebSphere Translation Server* on page 56.

Copying wts.jar to the Get-Resources deployment folder

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

To copy wts.jar to the Get-Resources 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-Resources deployment folder located at:

<Application server install>\WEB-INF\lib

5 Restart your application server.

Configuring Get-Resources to use the WebSphere Translation Server

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

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

1 Login to the Get-Resources 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.

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-Resources data in a mixture of languages, you can configure Get-Resources 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-Resources 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-Resources to use the WebSphere Translation Server* on page 56.

The translate button appears in the upper right tool bar.

The Translation button.

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

Click on the text you want to translate.

3 Click the translate button.
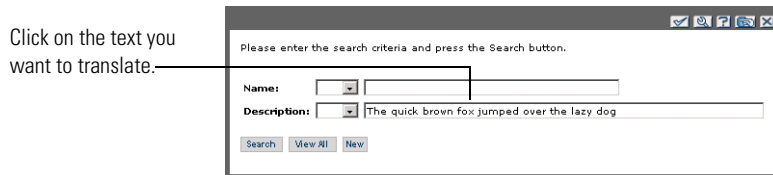
The Translation window opens.

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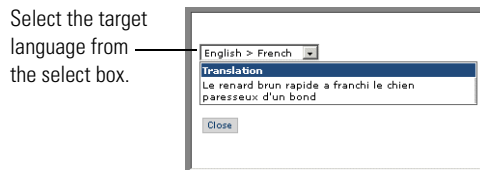
4 Select the target language to which you want to translate from the drop down select box.

The translation of your selection displays in the Translation box.

WebLogic 6.1 SP3 or SP4

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

To configure WebLogic 6.3 SP3 or SP4 with IIS:

Step 1 Stop both WebLogic and your Web server. *Stopping the servers* on page 58.

Step 2 Edit the startWebLogic.cmd file to set the system password, memory settings, and start mode. See *Editing startWebLogic.cmd* on page 59.

Step 3 Edit the Server.Policy file to set the debug to true. See *Editing Server.Policy* on page 60.

Step 4 Run the Get-Resources installer. See *Running the installer* on page 60.

Step 5 Move .jar files to the Java development kit ext folder. See *Moving .jar files to the Java development kit ext folder* on page 60.

Step 6 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 61.

Step 7 Configure IIS to use iisproxy.dll as an extension. See *Configuring the iisproxy.dll as an extension* on page 62.

Step 8 Create a virtual directory for Get-Resources in your Web server. See *Creating a virtual directory for Get-Resources* on page 63.

Step 9 Restart WebLogic and your Web server. See *Restarting the Servers* on page 63.

Stopping the servers

Before you begin configuring WebLogic, you must close your WebLogic server and Web server.

To begin configuring WebLogic:

1 Stop the WebLogic application server.

2 Stop the Web server.

Editing startWebLogic.cmd

To edit startWebLogic.cmd:

1 Open the file startWebLogic.cmd file in any text editor. The default file path is:

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. The default file path is:

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.

Running the installer

Run the Get-Resources installer using the Custom install option. See *Custom installation procedures* on page 83.

Moving .jar files to the Java development kit ext folder

To move .jar files:

1 Verify that the following directory exists. If it does not exist, create it:
c:\bea\jdk131\jre\lib\ext

2 Go to the Peregrine OAA Platform lib folder (typically bea\wlserver6.1\config\<my domain>\applications\oaa\WEB-INF\lib), where *<my domain>* is the WebLogic domain of the system on which WebLogic is installed. Move the following file to the \bea\jdk131\jre\lib\ext folder:

log4j-1.2.6.jar

3 Verify that oaasecurityproxy.jar is in the \bea\jdk131\jre\lib\ext folder.

4 Go the Peregrine OAA Platform external folder (typically
Peregrine\oaa\external) and copy the following files to the
\bea\jdk131\jre\lib\ext folder:

```
jaas.jar
jai_codec.jar
jai_core.jar
jce1_2_1.jar
jcert.jar
jnet.jar
jsse.jar
local_policy.jar
mlibwrapper_jai.jar
sunjce_provider.jar
US_export_policy.jar
```

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 <*Machine name*> and then click **Properties**.

Note: This is not one of the web sites; it is the parent node in the tree for the
web sites.

A dialog box opens.

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. The default file path is:

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 Verify that there is a mapping for the .wlforward extension. If it does not exist, click **Add** to map the .wlforward extension.

11 Enter the following information:

a   Executable: issforward.dll. The default file path is:

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

b   Extension: .wlforward.

12 Close the Internet Services management console.

Configuring the iisproxy.dll as an extension

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

To install iisproxy.dll as an extension:

1   Open Internet Services management console.

2   Right-click the **Default Web Site** node and then click **Properties**.

3   Click the **Home Directory** tab.

a   Select the **Read** check box to enable it.

b   From the **Execute Permissions** drop-down list, select **Scripts and Executables**.

4   Click **Configuration**.

The Application Configuration page opens on the App Mappings tab.

5   Verify that there is a mapping for the .jsp extension and that it is mapped to c:\bea\wlserver6.1\bin\iisproxy.dll.

a   If the .jsp extension mapping is not there, do the following:

b   Click **Add**.

c   Enter the following information:

▪   Executable: iisproxy.dll. The default file path is:

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

▪   Extension: .jsp.

Note: Ensure that the iisproxy.dll is located in the same directory as the iisforward.dll.

6 Close the Internet Services management console.

Creating a virtual directory for Get-Resources

To run Get-Resources, 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 can specify a different virtual directory name.

To configure a virtual directory:

1 Use the following guidelines to create the virtual directory on the Default Web Site.

Requirement	Setting
Create virtual directory	\<oaa\>
Access Permissions	Read, Run scripts
Map to physical path	\<Weblogic\>\applications\oaa
Set Execute Permissions to	Scripts and Executables

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

    b For *\<WebLogic\>*, enter the path to your WebLogic installation. The default file path is:

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

2 Right-click the newly created virtual directory under Default Web Site.

    a Select **Properties**.

    b Verify that the permission is **Scripts and Executables**.

3 Click **Configuration** on the Home Directory tab.

The Application Configuration page opens on the App Mappings tab.

4 Verify that there is a mapping for the **.jsp** extension and that it is mapped to c:\bea\wlserver6.1\bin\iisproxy.dll.

Restarting the Servers

You must restart the servers to activate the new WebLogic configurations.

To activate the WebLogic configurations:

1 Restart your Web server.
2 Restart the WebLogic server.
3 Start Get-Resources.

JRun 3.1

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

To configure JRun 3.1:

Step 1 Install a Java run-time environment. See *Installing a Java run-time environment* on page 64.

Step 2 Install JRun from the Macromedia Web site to the root of your hard drive (for example, C:\). See *Installing JRun* on page 65.

Step 3 Apply the latest JRun update. See *Applying the latest JRun update* on page 65.

Step 4 Deploy the Portal WAR file to JRun to create the necessary folder structure for Get-Resources. See *Deploying the Portal WAR file to JRun* on page 65.

Step 5 Run the Get-Resources installer. See *Running the installer* on page 68.

Step 6 Move js.jar to the Java development kit ext folder. See *Moving js.jar to the Java development kit* on page 68.

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 68.

Step 8 Configure your JRun Java settings. See *Configuring Java settings* on page 69.

Step 9 Configure JRun for IIS and create a virtual directory for Get-Resources in your Web server. See *Configuring JRun for IIS* on page 71.

Step 10 Restart JRun and your Web server. See *Restarting the Servers* on page 73.

Installing a Java run-time environment

The Get-Resources 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 81.

Installing JRun

Before you install Get-Resources, you must install JRun to your root directory.

To install JRun:

1 Browse to the following URL:

   http://www.macromedia.com/software/jrun/

2 Click the link **JRun 3.1 Available for Purchase**.

3 Follow the installation instructions provided.

Applying the latest JRun update

Before you install Get-Resources, 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-Resources in your application server. After you have deployed this file to JRun you will be ready to run the Get-Resources installer.

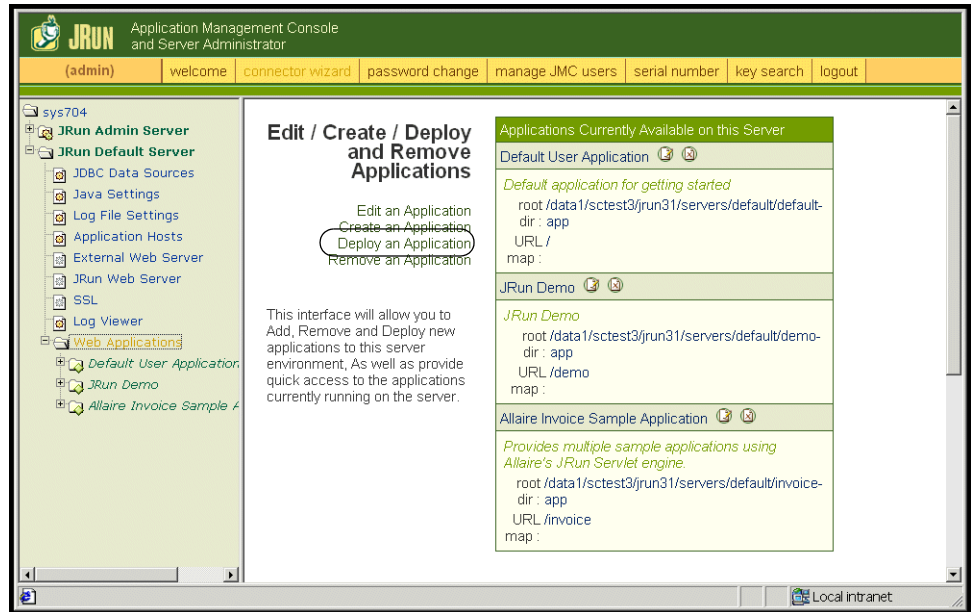### To deploy the Get-Resources Portal WAR file to JRun:

1 Open the JRun Management Console and log in.

2 Select JRun Default Server > Web Applications.
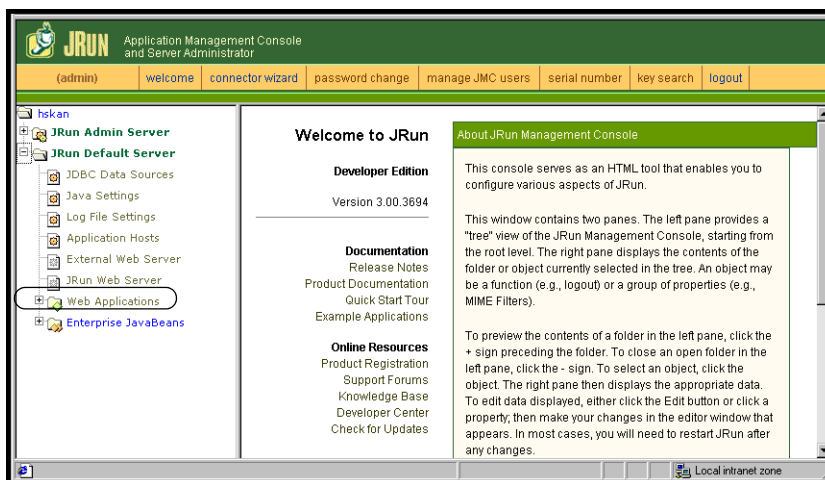
The Edit / Create / Deploy and Remove Applications page opens.

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.

Running the installer

Run the Get-Resources installer using the Custom install option. See *Custom installation procedures* on page 83.

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. The default file path is:

<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.

You will get a message that JRun is successfully connected to your Web server.
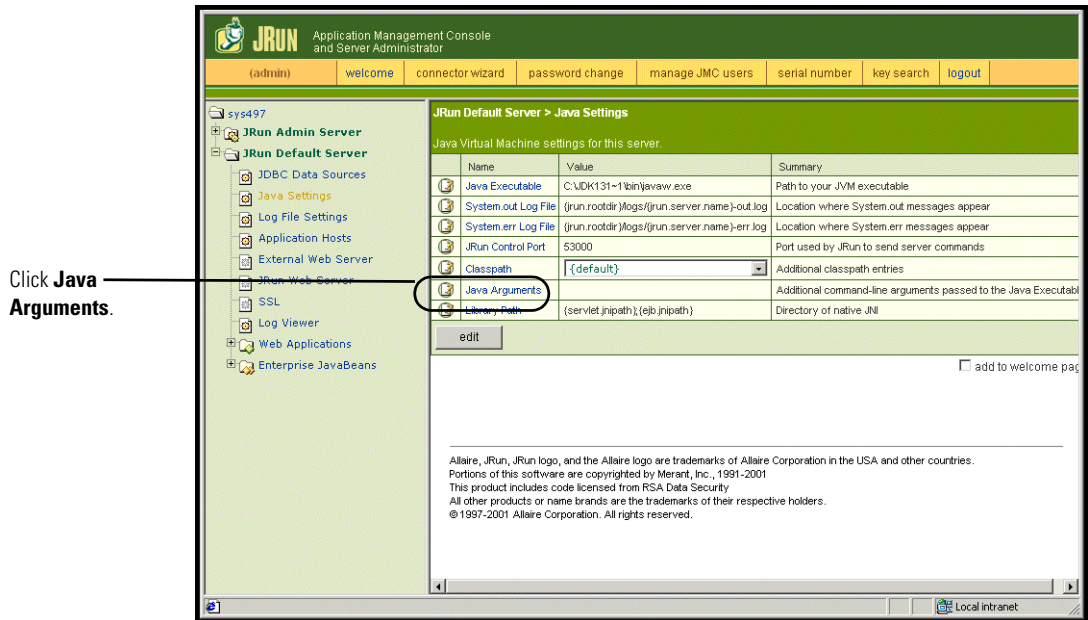
Configuring Java settings

After you have installed Get-Resources, 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**.

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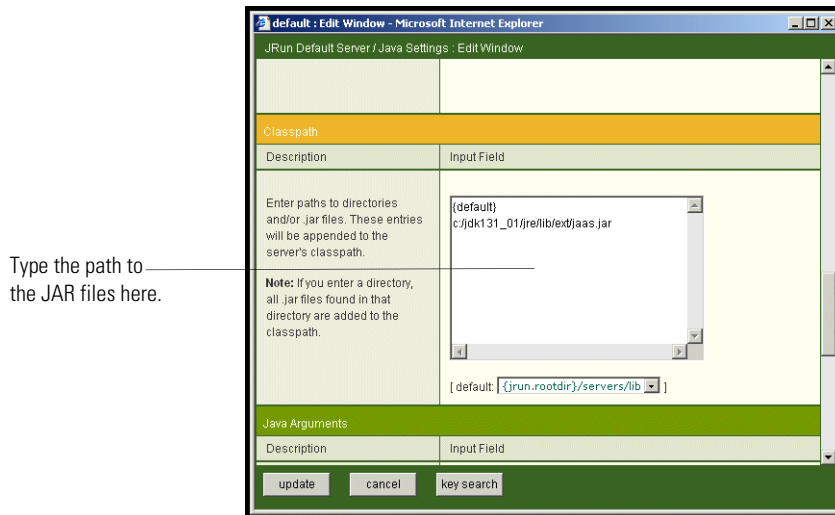
5 On the Java Settings page, click **Classpath**.

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:

   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.

Configuring JRun for IIS

To establish a connection between JRun and IIS, you must first verify the connection.

To configure JRun for IIS:

1 Open the IIS Management console.

2 Right-click the <*Machine name*> and then click **Properties**.

   Note: This is not one of the web sites; it is the parent node in the tree for the web sites.

A dialog box with an **Edit** button opens.

3 Click **Edit>ISAPI Filters**.

4 Verify that the JRun entry is correct and that the value contains the full path to the scripts/jrun.dll.

Filter Name: JRun Connector Filter Executable: /jrun.dll

5 Verify the following:

- A SCRIPTS virtual directory is defined in IIS.
- It points to your Inetpub/Scripts directory.
- It has **Scripts and Executables** permission.

You then 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 can specify a different virtual directory name.

To configure a virtual directory:

1 Use the following guidelines to create a virtual directory for oaa under Default Web Site.

Requirements for Get-Resources virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Map to physical path	<JRun>\oaa
Set Execute Permissions to	Scripts and Executables

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

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

C:\JRun\servers\default\oaa

3 After creating the oaa virtual directory, right-click on it and select **Properties**.

4 Verify that the permission is **Scripts and Executables**.

Restarting the Servers

You must restart the servers for the new JRun configurations to take effect.

To activate the Jrun configurations:

1 Restart your Web server.
2 Restart the JRun Default server.
3 Start Get-Resources.

Typical installation option

A typical installation of Get-Resources 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-Resources:

Applications and File Locations

Get-Resources 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-Resources	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-Resources 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-Resources.

Communications ports

Get-Resources uses the following communications ports in a typical installation. After installation, you can configure Get-Resources 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

| Default port	Component used by	Alternate port
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-Resources.

Typical installation procedures

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

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

1 Insert the Get-Resources 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**.
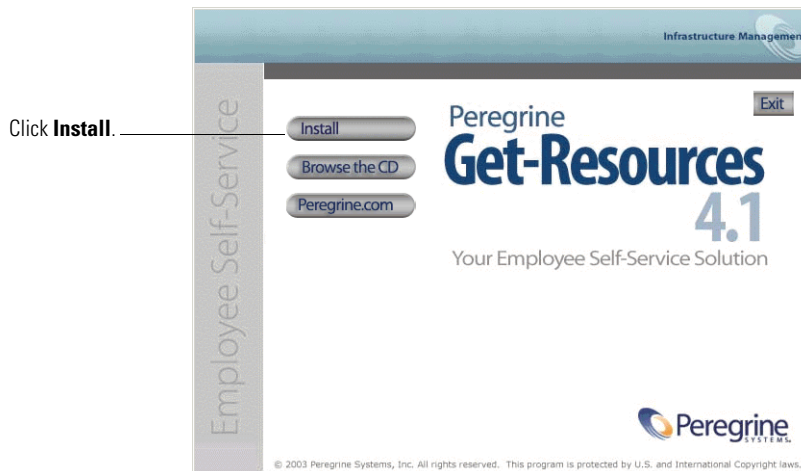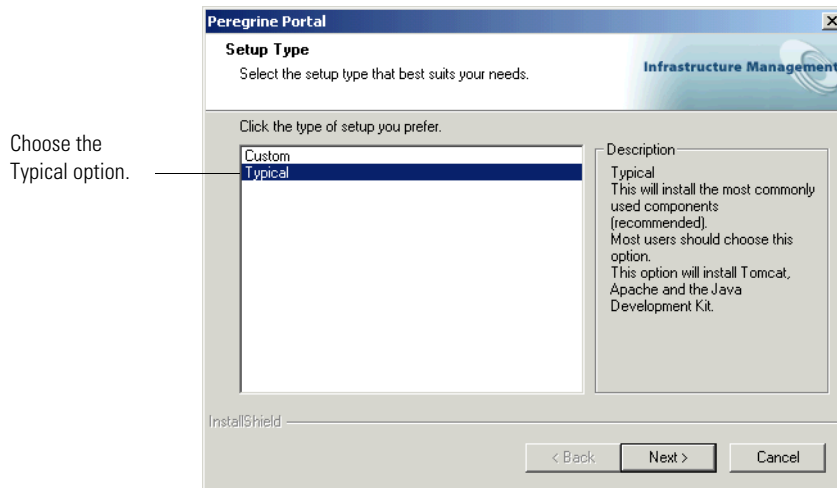

Click **Install**.

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

Note: Setup stops Apache and Peregrine Tomcat services. Follow the prompts to continue.

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

Note: This screen does not display during a maintenance or upgrade installation.
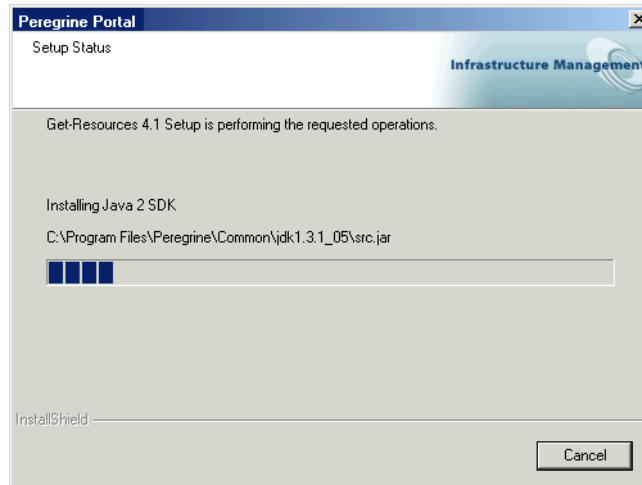

Choose the Typical option.

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

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

Note: This screen does not open during a maintenance or upgrade installation.

Click **Next**.
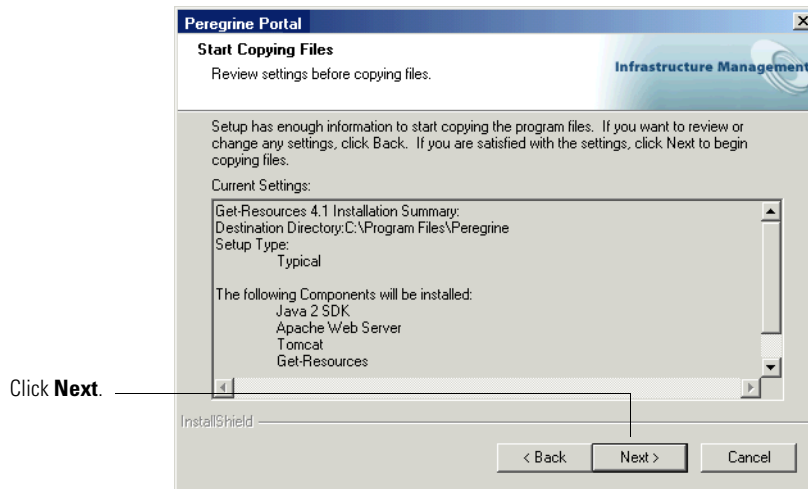
The Setup Status screen that appears shows that Get-Resources is installing the files to your system.
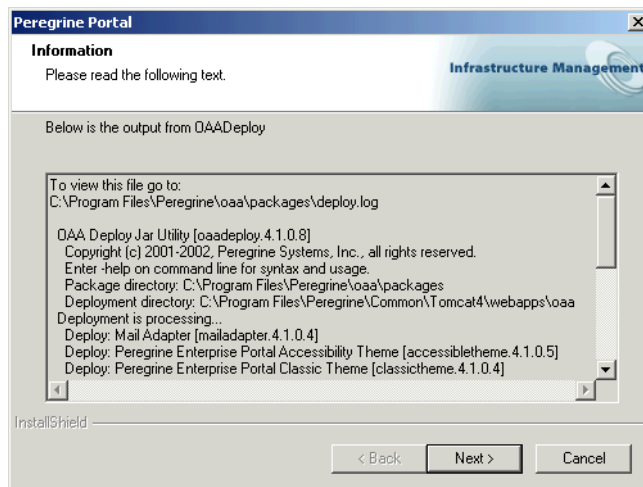
A status message indicates that the Get-Resources packages are being deployed to your server. This phase can take several minutes.
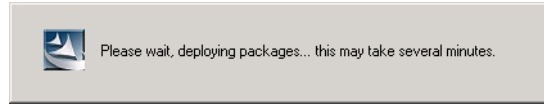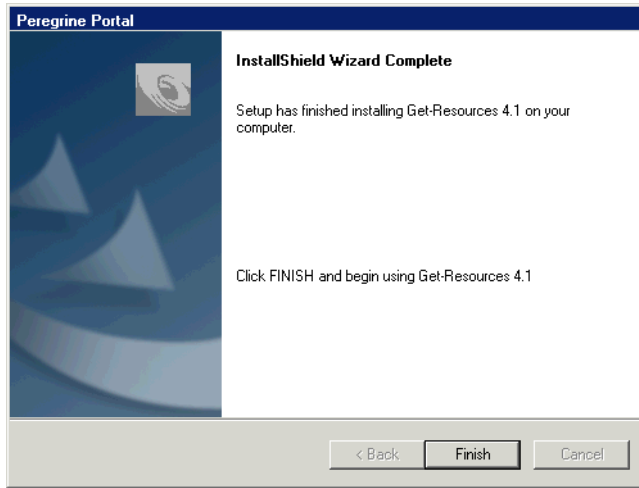

At this time, the setup program is performing the following:

- Copying the documents
- Creating and starting the "Apache" and "PeregrineTomcat" services

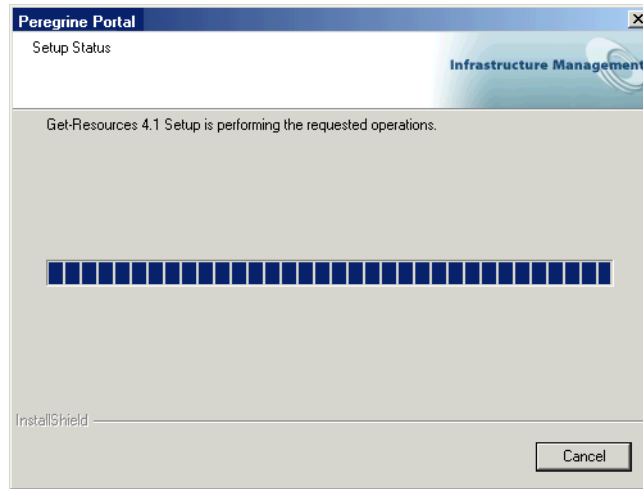**5** Read the information from OAAdeploy, then click **Next**.

6 In the InstallShield Wizard Complete screen click **Finish**.

Note: The text in the dialog box reads, "Maintenance Complete" during a maintenance or upgrade installation.

The setup program completes the final steps of the installation.

7 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-Resources on a Windows operating system server.

Custom installation option

The following section describes how to perform a custom installation of Get-Resources on a Windows operating system server.

Custom installation components

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

Application options

Get-Resources Component	Options
Web Server	■ Apache 2.0.43 (runs on port 80.) ■ IBM HTTP Server 1.3.19 (runs on port 80.) ■ Microsoft IIS 5.0 (runs on port 80.)
Application Server	■ Tomcat 4.1.24 ■ WebSphere 4.02 ■ WebLogic 6.1 SP3 ■ JRun 3.1
Java Development Kit	■ Java 2 SDK

Communications ports

The communications ports that a Get-Resources custom installation uses depends on the application components that you select. Refer to your Web and application server documentation to determine what communications ports they require. After installation, you can configure Get-Resources to use alternate communications ports if your local network already uses particular communications ports.

Default port	Component used by	Alternate port
8005	Tomcat application server administration	8015
8009	Tomcat application server worker file	8019
80	IIS Web server	8081

Port conflicts

If you are using Tomcat as your application server and running Oracle 9.2.0.1, then by default, there will be a port conflict over port 8009. To resolve a port conflict, you can do one of the following:

■ Install Oracle 9.2.0.1 on a separate server (recommended).

■ Configure Oracle 9.2.0.1 to use a port other than 8009.

- Configure the primary instance of Tomcat (and all other instances) to use a port other than 8009. For more information about setting Tomcat ports, see the *Load balancing* chapter.

For more information about checking for port conflicts, see *Check for Tomcat port conflicts* in the *Troubleshooting* chapter.

Custom installation procedures

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

1 Insert the Get-Resources 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**.
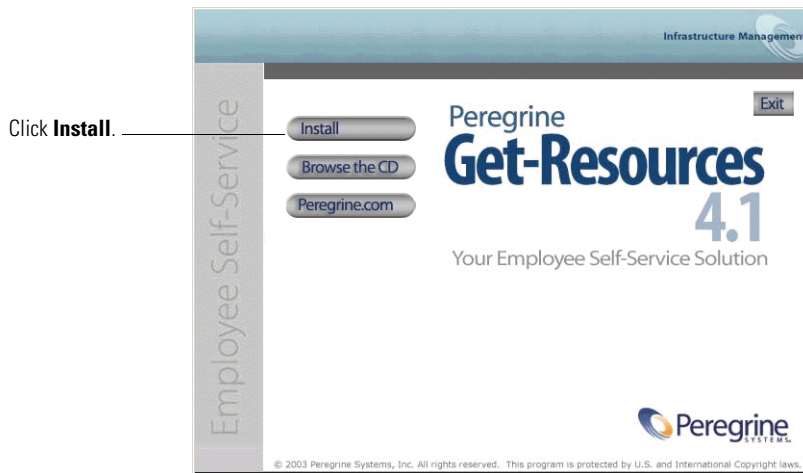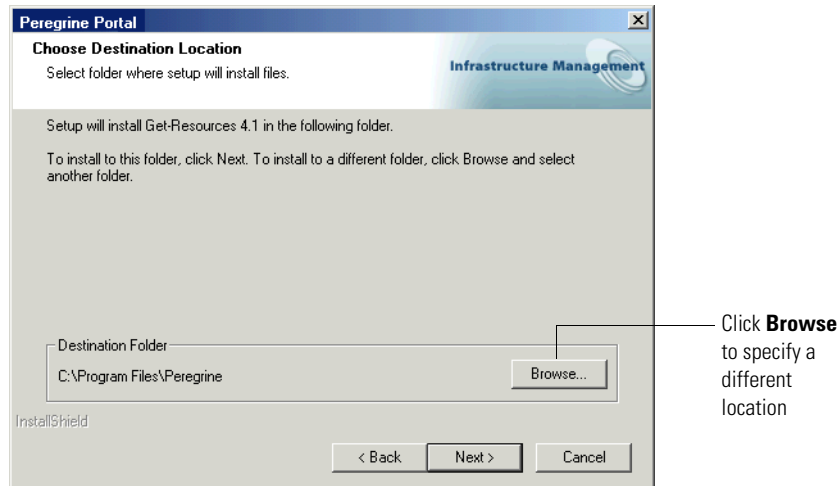
Click **Install**.

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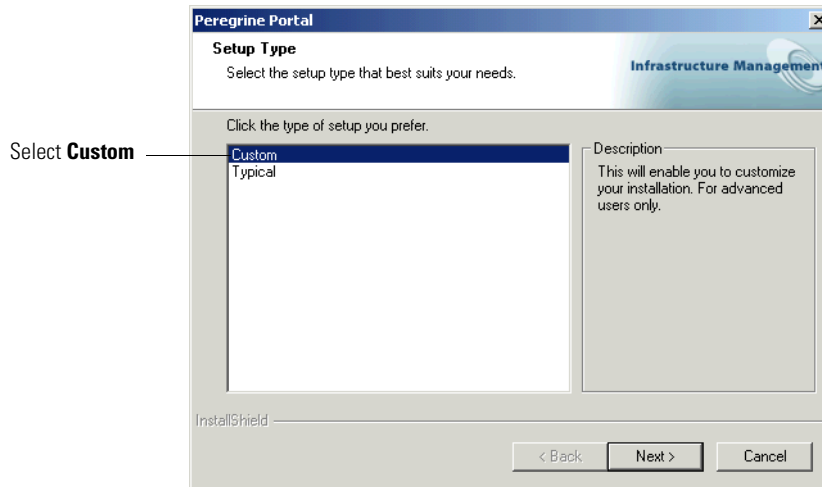
4 In the Choose Destination Location screen, browse to the location where you want Get-Resources installed. Click **Next** to continue.

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

Click **Browse** to specify a different location

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

Select components to install ——
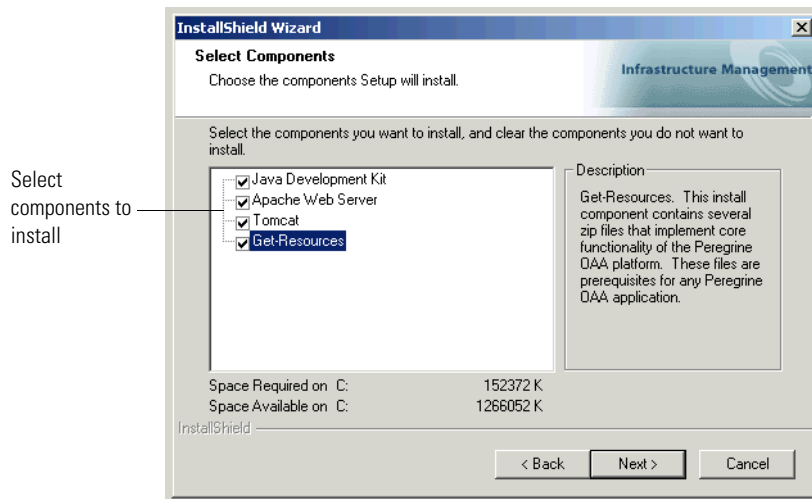
Important: Clear the check box next to components that you do *not* want to install.

For a development environment, select or clear the components that you will install manually or for which you have alternate software. For example, clear the Apache Web Server option to use an alternate Web server.

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

- Get-Resources. Installs the program files necessary for Get-Resources. You must install the Get-Resources files on the same machine as the primary application server.

- Apache Web Server. Get-Resources requires a Web server in order to serve pages. Clear the check box in this option so that the installation procedure does not install the Apache Web server.

- Java Development Kit. Installs the Sun Microsystems Java Development Kit (Java 2 SDK). You must install this component on every machine running an application server.

■ Tomcat. Installs the Tomcat application server. Get-Resources requires at least one application server to process Java applications. You may also install multiple instances of your application server for load balancing. You must install this component on the same machine as the Get-Resources files.

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

6 If you do not choose Tomcat as the application server, select the application servers you want Get-Resources to configure from the Select Application Servers Type dialog box, and click **Next**.
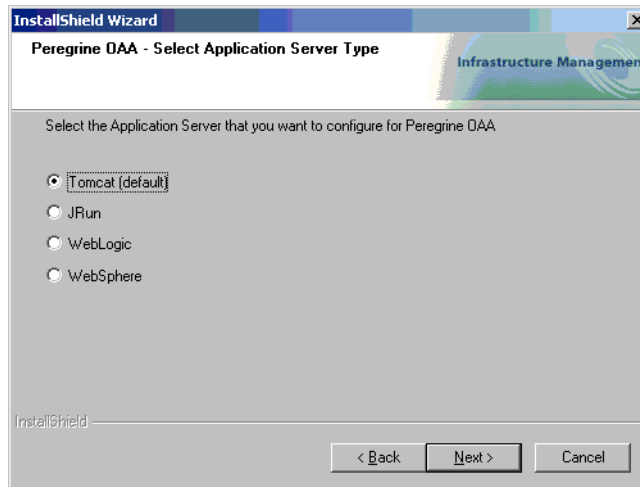
7 If you do not choose the Apache Web Server option, select the Web server you want to install from the Select Web Server dialog box.
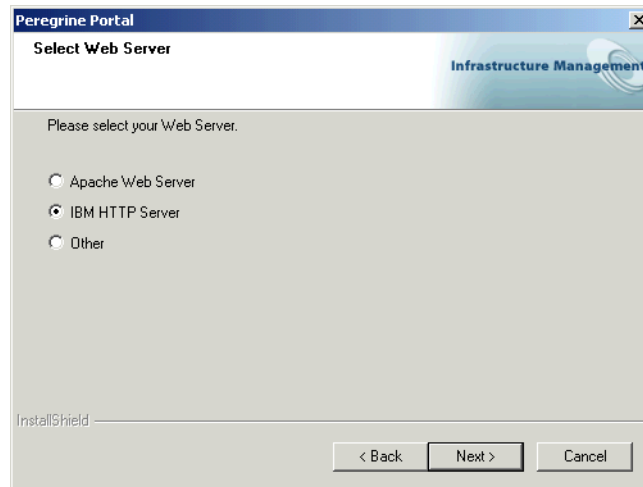
8 In the Start Copying Files screen, verify the Custom installation components. To review or change settings, click **Back**. To continue with the installation, click **Next**.
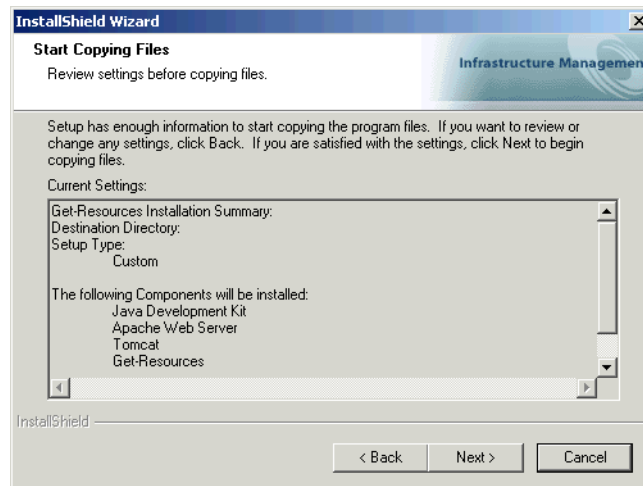
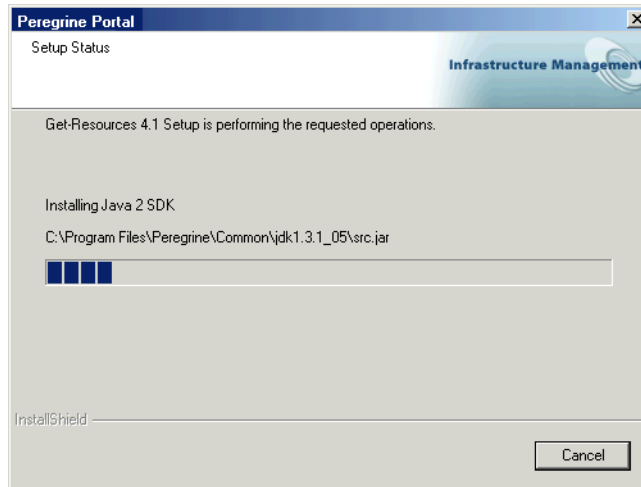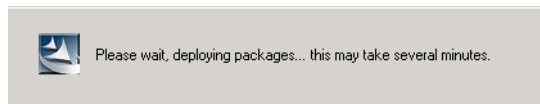The Setup Status screen is displayed while the installation program performs the requested operations.

9 In the Setting CATALINA_OPTS screen, select **Yes** if you want to configure this parameter for Tomcat memory settings, then click **Next**.

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

This phase may take several minutes.

If you are using Tomcat, the Tomcat Installation Directory window appears.
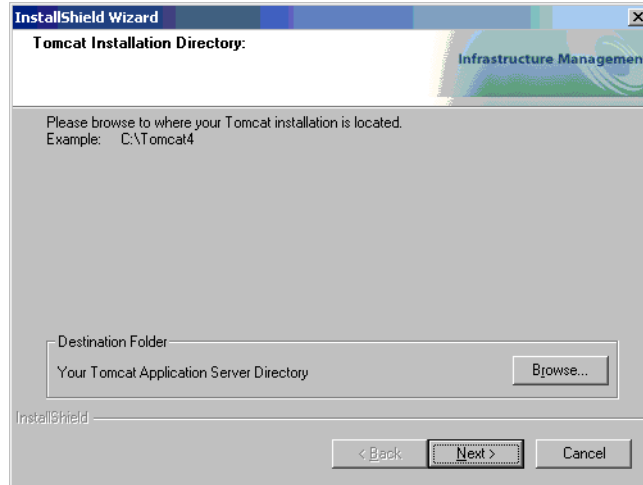
10 Choose your installation directory, and click **Next**.

Note: If you are not using Tomcat, you will not see this window. Go to step 11.

11 In the Presentation Location dialog box that opens, choose a location from which to run Get-Resources, and then click **Next**.
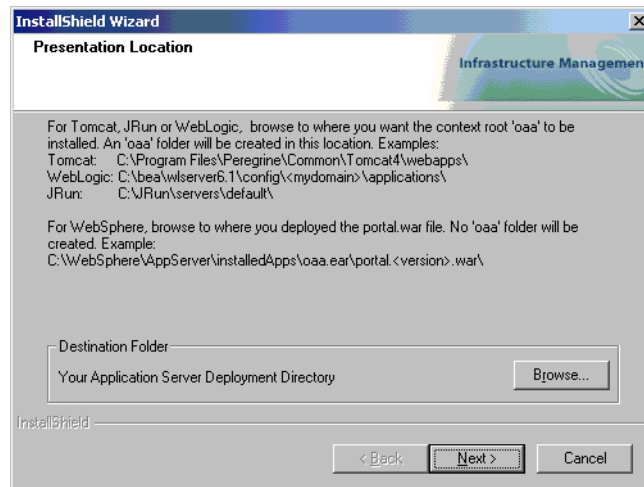
The setup program deploys the packages.

12 Read the information from OAAdeploy, then click **Next**.
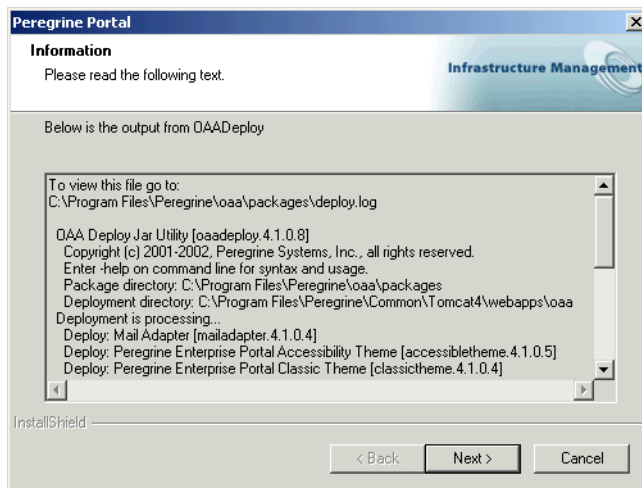
A Setup Status dialog box opens as the installation is completed.

The following warning message appears:
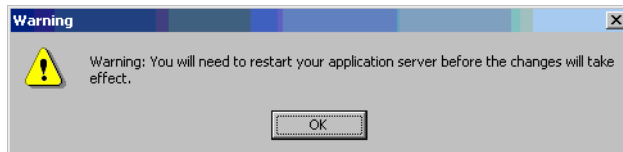
13 Click **OK**.

14 In the InstallShield Wizard Complete screen, click **Finish** to end the Get-Resources installation program.
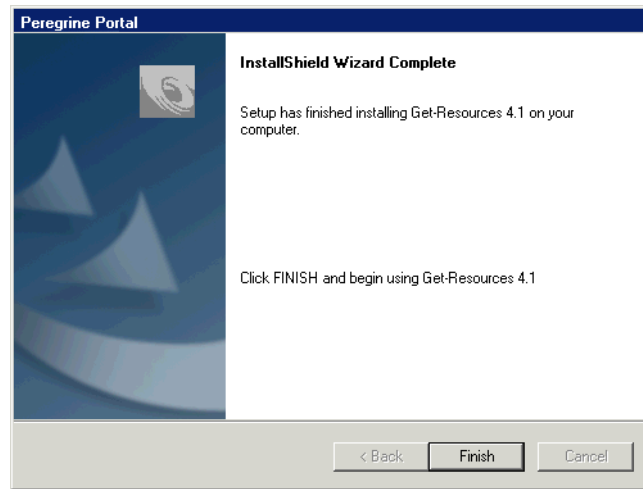
15 After the InstallShield Wizard screen closes, access Windows Services to verify startup of your web server and application server services.

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-Resources on a Windows operating system server.

Uninstalling Get-Resources

Follow these procedures to uninstall Get-Resources from your Windows system.

Warning: These procedures remove all the components that you selected to install. If you chose the Typical installation option, uninstall removes Get-Resources, Peregrine Tomcat, Apache, and JDK. If you chose the Custom installation option, then only those components that you selected to install are removed.

To uninstall Get-Resources:

1 Access the Windows Add/Remove Programs utility.

2 Select **Peregrine Portal 4.1** and click **Change/Remove**.

   A status message indicates that the setup program is preparing the InstallShield wizard to guide you through the process.

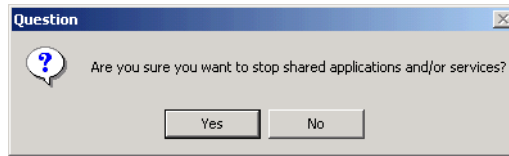3 The Close Programs screen opens if any Get-Resources services or applications are running. Click **Next** to continue.

4 The verification message box opens. Click **Yes** to continue.

   Status messages indicate the termination of the services for Apache and Tomcat.

5 The Confirm Uninstall dialog box opens. Click **OK** to remove Get-Resources.
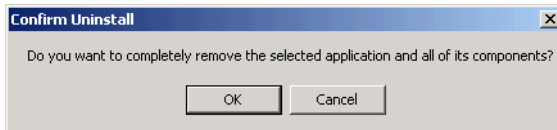
   Important: Back up any data you want to save before continuing.

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 to 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 files 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-Resources.
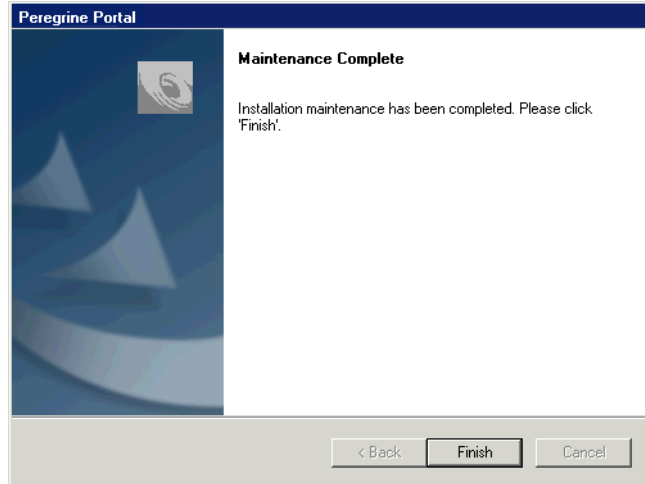
Testing your installation

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

To test your Get-Resources 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-Resources 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-Resources administration login page does not open, see *Troubleshooting* for more information.

3 Installation for UNIX Platforms

CHAPTER

This chapter covers the following topics:

Choosing an installation environment

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

- Development environment
- Production environment

The Get-Resources 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-Resources on one computer system.

You have two choices of development environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Resources deployed on Tomcat 4.1.24 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Resources

The Get-Resources 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-Resources on different servers to maximize performance.

You have two choices of production environment:

- Typical installation
  - Apache 2.0 Web server
  - Get-Resources deployed on multiple instances of Tomcat 4.1.24 application server
- Custom installation
  - Choice of Web server
  - Choice of application server where you deploy Get-Resources

Development Environment

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

To install Get-Resources in a typical development environment:

Step 1 Acquire all necessary hardware and software.

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

Step 3 Run the Get-Resources installer and select the Typical installation option. See *Typical Installation Option* on page 142.

Step 4 Configure the back-end databases and create Get-Resources users. See the *ServiceCenter and AssetCenter Administration* chapter of this guide.

To install Get-Resources in a custom development environment:

Step 1 Acquire all necessary hardware and software.

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

Step 3 Install alternate application and Web servers.

Step 4 Configure the alternate application server for Get-Resources. See *Configuring alternate application servers* on page 103.

Step 5 Run the Get-Resources installer and select the Custom installation option. See *Custom Installation Option* on page 150.

Step 6 Configure the back-end databases and create Get-Resources users. See the *ServiceCenter and AssetCenter Administration* chapter of this guide.

Production Environment

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

To install Get-Resources in a typical production environment:

Step 1 Acquire all necessary hardware and software.

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

Step 3 Run the Get-Resources installer and select the Typical installation option. See *Typical Installation Option* on page 142.

Step 4 Configure multiple instances of Tomcat for load balancing on the Apache Web server. See the *Load Balancing* chapter of this guide.

Step 5 Configure the back-end databases and create Get-Resources users. See the *ServiceCenter and AssetCenter Administration* chapter of this guide.

To install Get-Resources in a custom production environment:

Step 1 Acquire all necessary hardware and software.

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

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

Step 4 Configure the alternate application server for Get-Resources. See *Configuring alternate application servers* on page 103.

Step 5 Run the Get-Resources installer and select the Custom installation option. See *Custom Installation Option* on page 150.

Step 6 Configure the Web servers and application servers for load balancing. See the *Load Balancing* chapter of this guide.

Step 7 Configure the back-end databases and create Get-Resources users. See the *ServiceCenter and AssetCenter Administration* chapter of this guide.

Migrating Get-Resources from previous versions

To migrate older versions of Get-It or Get-Resources to Get-Resources 4.1 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-Resources 4.1:

Step 1 Review the customizations of previous version and determine which customizations need to be recreated in Get-Resources 4.1. See *Recreating customizations in Get-Resources 4.1* on page 99.

Step 2 Install Get-Resources 4.1 on a new system. See *Choosing an installation environment* on page 96.

Step 3 Apply any required configuration changes to the back-end database you want to migrate to Get-Resources 4.1. See *Configuring an existing back-end database for Get-Resources 4.1* on page 102.

Recreating customizations in Get-Resources 4.1

You cannot directly migrate customizations implemented before version 4.0.1 to Get-Resources 4.1. Instead, you must recreate your changes using the new features and methods available in Get-Resources 4.1.

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

No customizations

If you have made no customizations to Get-Resources, you can simply install Get-Resources 4.1 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-Resources 4.1 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-Resources 4.1 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-Resources 4.1 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-Resources 4.1, 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-Resources 4.1 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-Resources 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-Resources 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-Resources 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-Resources tailoring kit.

Configuring an existing back-end database for Get-Resources 4.1

The following table lists the options available for data migration.

Get-Resources 2.5 to Get-Resources 4.1

Back-end version	Migration required
AssetCenter 3.6	Apply Get-Resources 4.1 scripts to existing AssetCenter 3.6.
AssetCenter 4.x	Apply Get-Resources 4.1 scripts to existing AssetCenter 4.x.
ServiceCenter 3.0	Upgrade to ServiceCenter 4.x or 5.0.x.
ServiceCenter 4.x	Apply Get-Resources 4.1 unload files to the existing ServiceCenter 4.x and apply the unload files under the upgrade folder.

Get-Resources 4.0.1 to Get-Resources 4.1

Back-end version	Migration required
ServiceCenter 4.x	Apply all unload files delivered with Get-Resources 4.1, including the file under the upgrade folder. **Note:** Delete the old unload files (scgr.unl, scgrdata.unl, scrmdemo.unl, 201patch6.unl).
ServiceCenter 5.0.x	Apply all unload files delivered with Get-Resources 4.1 for ServiceCenter 5.0.

Upgrading from Get-Resources 4.0.1 to 4.1

To upgrade from Get-Resources 4.0.1 to Get-Resources 4.1, you simply need to run the 4.1 installer. See either the *Typical Installation Option* on page 142 or *Custom Installation Option* on page 150 for more information.

> **Note:** Note that the **<appsrv>/WEB-INF/web.xml** file has been renamed to **<appsrv>/WEB-INF/web.xml.XXX.bak**, where *<appsrv>* is the path to your application server, and **XXX** represents a unique sequence of characters generated by File.createTempFile(). This will preserve any customizations that you might have (see *Preserving customized web.xml file settings*, next).

After upgrading Get-Resources from 4.0.1 to 4.1, a new file called **web.xml.xxx.bak** is created in the *<appsrv>*/WEB-INF directory. You must reconcile the contents of this file with the contents of the new **web.xml** file. Any portion of **web.xml.xxx.bak** file that does not exist in the new **web.xml** file needs to be added to the **web.xml** file.

Preserving customized web.xml file settings

You can open the **web.xml**.xxx.bak file, copy its customized configuration settings into the new **web.xml** file, and save the new file to preserve your customizations.

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:

- *Existing Tomcat and Apache servers*
- *WebSphere Application Server 4.0.2*
- *WebSphere Application Server 5.0*
- *WebLogic 6.1 SP3 or SP4*
- *JRun 3.1*

The Get-Resources typical installation option installs Tomcat 4.1.24 and connects it to an Apache 2.0 web server. You can also install Tomcat 4.1.24 using the custom installation option.

Important: If you want to use an application server other than Tomcat 4.1.24, then you must configure your application and Web servers *prior* to running the Get-Resources installer.

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

Existing Tomcat and Apache servers

If you use the typical installation option, the Get-Resources installer configures Tomcat to connect to a new instance of the Apache Web server. If you have existing instances of Tomcat or Apache Web Server installed, you can configure Get-Resources to use these existing instances by copying the necessary files from a typical installation.

To configure an existing Tomcat server to connect to an Apache server:

1 Copy the following files from the installation CD **\SupportFiles**… directory to the directories indicated below.

Copy this file	To the following location
■ mod_jk.conf	The **/conf** directory of your existing Tomcat installation. The default source file path is: /usr/local/peregrine/common/Tomcat 4/conf
■ workers.properties	The **/conf** directory of your existing Tomcat installation. The default source file path is: /usr/local/peregrine/common/Tomcat 4/conf
■ mod_jk.dll	The **/modules** directory of your existing Apache installation. The default source file path is: /usr/local/peregrine/common/apache2/modules

Note: The mod_jk.dll included with this release is compatible with Apache 2.0.43 and Tomcat 4.1.25. If you are using other versions, refer to the jakarta.apache.org/builds/jakarta-tomcat-connectors/jk/doc site to download the compatible version.

2 Using a text editor, open the files mod_jk.conf and workers.properties. These files are located in the **/conf** directory of your Tomcat installation.

   a Find all instances where the path to Tomcat appears and edit these to reflect your current Tomcat 4.1 installation path.

   b Find all instances where the path to JDK appears and edit these to reflect your current JDK installation path.

3 Using a text editor, open the httpd.conf file. This file is located in the **/conf** directory of your Apache installation.

   a Add the path to your existing Tomcat installation to the include statement in the Global Environment section:

```
### Section 1: Global Environment
...
include "<Tomcat_path>/conf/mod_jk.conf"
```

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

b Add login.jsp to the list of files in the DirectoryIndex section:

```
# DirectoryIndex: Name of the file or files to use as a pre-written
# HTML directory index.  Separate multiple entries with spaces.
#
<IfModule mod_dir.c>
 DirectoryIndex index.html login.jsp
</IfModule>
```

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

Alias <Tomcat>/webapps/oaa

where *<Tomcat>* is the path to your Tomcat installation.

4 Install Get-Resources using the **Custom** option. See *Custom Installation Option* on page 150.

5 Restart Tomcat and Apache.

6 Browse to the Get-Resources login URL and verify that you can successfully connect.

> **Note:** Depending on your Web server configuration, if you browse to http://servername/oaa, the Web server may display a list of all the OAA files instead of the login page.
>
> If your server displays this behavior, follow these steps to configure your Web server to display the OAA login page instead of a directory listing.

To configure Apache to display login.jsp by default:

1 Open Apache's conf/httpd.conf file in a text editor.

2 Find the existing line that reads DirectoryIndex index.html.

3 Add login.jsp to the end:

DirectoryIndex index.html login.jsp

4 Save httpd.conf.

5 Restart the Apache Web server.

WebSphere Application Server 4.0.2

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

Note: The Get-Resources installer creates duplicate alias entries in the IBM HTTP Server when you install more than one Peregrine OAA Platform application on WebSphere.

Duplicate entries can also occur if you reinstall Get-Resources or install another Peregrine OAA Platform application on a system that formerly had Get-Resources installed on it.

Remove any duplicate alias entries from the IBM HTTP Server httpd.conf file.

To configure WebSphere Application Server 4.02:

Step 1 Install WebSphere 4.02. Your version of WebSphere 4.0.2 includes the IBM HTTP Server. *Installing WebSphere 4.02* on page 107.

Step 2 Deploy the Portal WAR file to WebSphere to create the necessary folder structure for Get-Resources. See *Deploying the Portal WAR file to WebSphere* on page 108.

Step 3 Set the JVM Java heap size for each WebSphere instance running Get-Resources. See *Setting the Java heap size* on page 110.

Step 4 Create the virtual directory you want to use for Get-Resources in your Web server. See *Configuring a virtual directory for IBM HTTP Server* on page 112.

Step 5 Run the Get-Resources installer. See *Running the Get-Resources installer* on page 112.

Step 6 Regenerate and configure. See *Regenerating the plug-in configuration* on page 113.

If you plan on setting up a WebSphere Portal Server or a WebSphere Translation Server, see *Installing WebSphere Portal Server* on page 116 or *Configuring WebSphere Translation Server for Get-Resources* on page 126.

Installing WebSphere 4.02

Purchase and install IBM WebSphere 4.0.2. Your version of WebSphere 4.0.2 includes the IBM HTTP Server.

Verify that you install fix pack 2. To check this, go to the default_server_Stdout.log file under /Websphere/AppServer/logs.

Deploying the Portal WAR file to WebSphere

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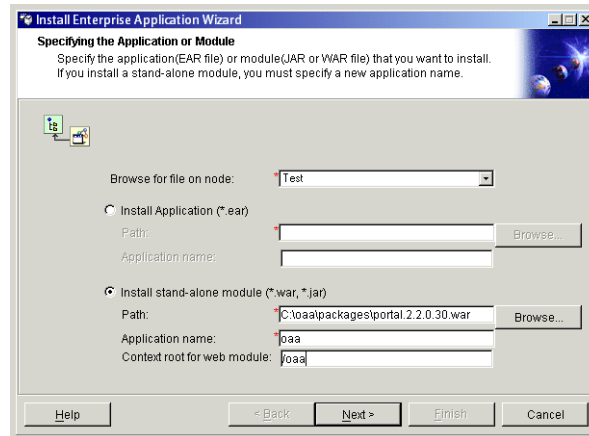
To deploy the Portal WAR file to WebSphere:

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-Resources 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.

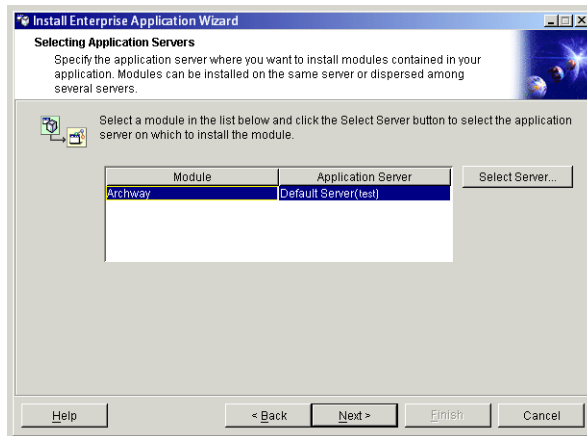The following screen shows the completed form.

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-Resources 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-Resources:

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.

Running the Get-Resources installer

Run the Get-Resources installer and select the Custom installation option. See *Custom Installation Option* on page 150.

If you plan on setting up a WebSphere Portal Server or a WebSphere Translation Server, see *Installing WebSphere Portal Server* on page 116 or *Configuring WebSphere Translation Server for Get-Resources* on page 126.
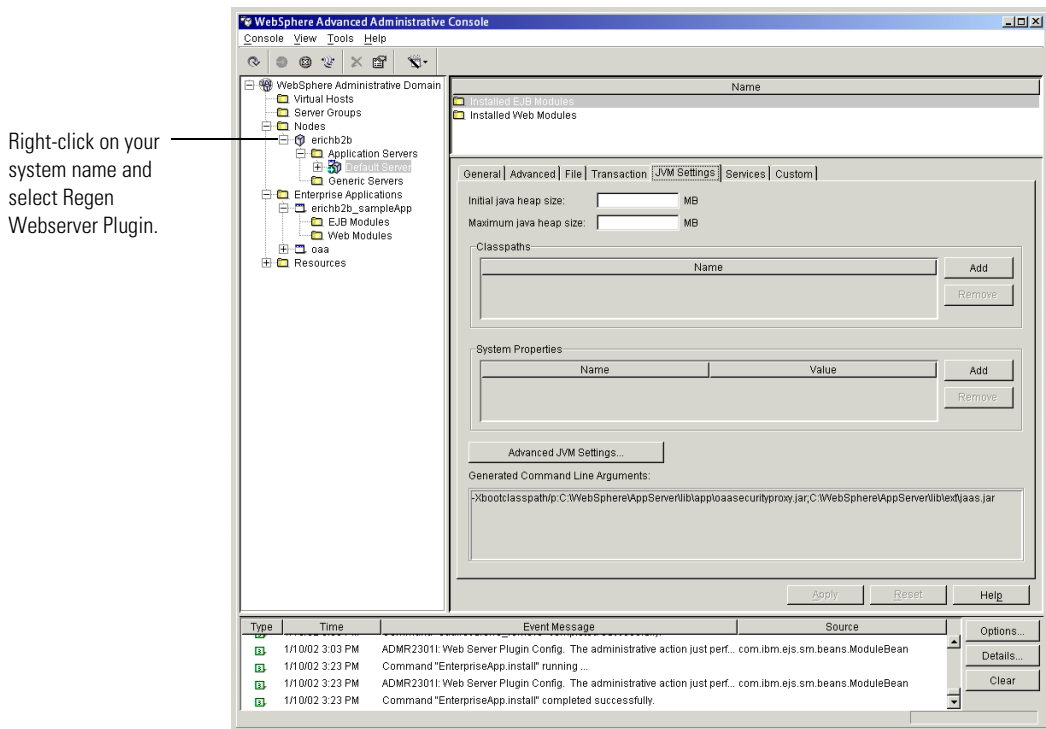
Regenerating the plug-in configuration

You must regenerate the plug-in configuration using the WebSphere Admin console after running the Get-Resources installer.

To regenerate the plug-in configuration:

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

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

The server settings page opens.

Right-click on your system name and select Regen Webserver Plugin.

3 Right-click on the <*System Name*>, then select **Regen Webserver Plugin**.

4 Copy the following lines from the <settings> section of **<appserver>/WEB-INF/default\archway.xml** to the <settings> section in your **<appserver>/WEB-INF/local.xml** file, where <*appserver*> is the location of your application server:

<SSLProvider>com.ibm.jsse.JSSEProvider</SSLProvider>

```
<HTTPSHandlerPkg>com.ibm.net.ssl.internal.www.protocol</HTTPSHandlerPkg>
<CryptoProvider>com.ibm.crypto.provider.IBMJCE</CryptoProvider>
```

5 Restart your application server.

WebSphere Application Server 5.0

Use the following procedures to configure WebSphere 5.0 to run
Get-Resources on UNIX. WebSphere Application Server 5.0 is referred to as
WAS5.

Note: The Get-Resources installer creates duplicate alias entries in the IBM
HTTP Server when you install more than one Peregrine OAA
Platform application on WebSphere.

Duplicate entries can also occur if you reinstall Get-Resources or
install another Peregrine OAA Platform application on a system that
formerly had Get-Resources installed on it.

Remove any duplicate alias entries from the IBM HTTP Server
httpd.conf file.

To run Get-Resources 4.x on WebSphere Application Server 5.0:

1 Set the WAS_HOME environment variable to your WAS5 home directory.

The default location is: opt/WebSphere/AppServer.

2 With WAS5 running, log in to the Admin console and create a new
Enterprise Application using portal.war from the packages directory on the
Get-Resources 4.1 CD.

Note: The important option to specify is the context root, typically /oaa or
/getit.

  a Specify the context root.
  b Keep all the other settings at their defaults.
  c Save the server configuration.

3 Run setup from the CD.
  a Choose the **Custom** install.
  b Clear the Tomcat, JDK, and Apache options.

4 When prompted for the deployment directory, browse to it under the WAS5 installed applications directory; typically, opt/WebSphere/AppServer/installedApps/[hostname]/oaa.ear/portal.war.

5 When the installation is complete, delete pop3.jar from opt/WebSphere/AppServer/java/jre/lib/ext.

It is no longer needed because WAS5 includes it in mail.jar.

6 Copy js.jar from …portal.war/WEB-INF/lib to opt/WebSphere/AppServer/java/jre/lib/ext.

7 From the IBMHttpServer\conf\httpd.conf file:

a Add the following lines.

```
LoadModule ibm_app_server_http_module
"<AppServerPath>/bin/mod_ibm_app_server_http.dll"
WebSpherePluginConfig
"<AppServerPath>/config/cells/plugin-cfg.xml"
```

b Verify that the following line is in the file; if not, add it.

```
Alias /oaa/ "opt/WebSphere/AppServer/installedApps/[hostname]
/oaa.ear/portal.war/"
```

The installer fails to do this because earlier versions were named differently.

Note: This alias needs to match the context root specified in step 2. The portal.war deployment folder has the same name as the portal.war file you deployed in step 2.

8 Using a text editor:

    a Add the following UriGroup mappings to the **plugin-cfg.xml** file located in **WebSphere\AppServer\config\cells**.

```
 <UriGroup Name="default_host_server1_sys_Cluster_URIs">
 :
 :
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/archway"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/rpcrouter"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/messagerouter"/
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/download/*"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/servlet/*"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.do"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsp"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsv"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/*.jsw"/>
  <Uri AffinityCookie="JSESSIONID" Name="/oaa/j_security_check"/>
 </UriGroup>
```

    b Save and close the file.

9 Restart WAS5.

10 Log in to the Admin console again.

    a From Environment on the left side, click **Update Web Server Plugin**.

    b Click **OK**.

11 Restart the IBM HTTP Server.

12 Log in to **admin.jsp** and continue configuring the system as usual.

Installing WebSphere Portal Server

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

- All Get-Resources and WebSphere components running on a single system. See *Recommended WebSphere Portal Server configuration* on page 117.
- Get-Resources components running on one system and WebSphere components running on another. See *Alternate WebSphere Portal Server configuration* on page 118.

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

Note: The OAA interface to the WebSphere Translation Server requires a mouse to use. The translation interface will be made 508 accessible in a future release.
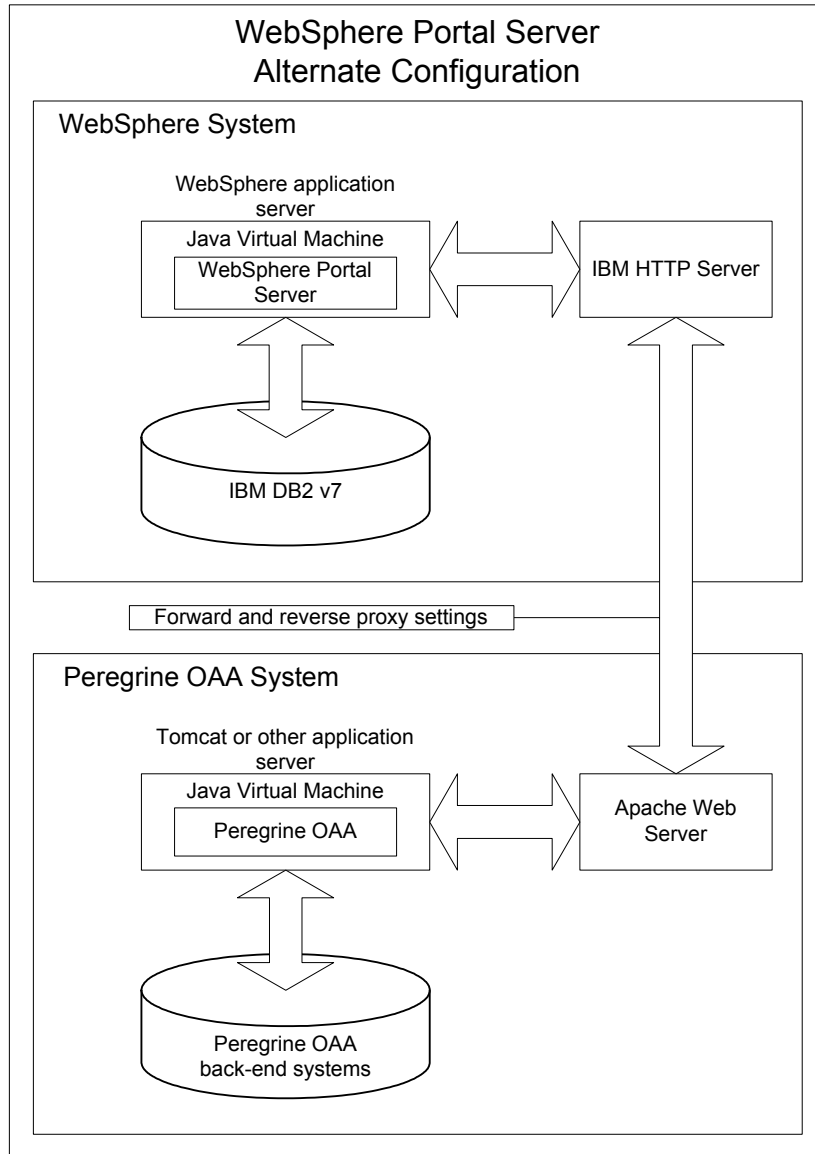
Recommended WebSphere Portal Server configuration

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

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 120.

Step 2 Generate a Get-Resources WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Resources WAR file* on page 121.

Step 3 Login to the Get-Resources server and stop the WebSphere application server. See *Stopping the WebSphere application server* on page 122.

Step 4 Modify the local.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the local.xml file* on page 122.

Step 5 Modify the web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 122.

Step 6 Modify the ibm-web-ext.xmi file to set the fileServingEnabled parameter. See *Modifying the ibm-web-ext.xmi file* on page 123.

Step 7 Start the WebSphere application server. See *Starting the WebSphere application server* on page 124.

Step 8 Deploy the Get-Resources WAR file to WebSphere Portal Server. See *Deploying the Get-Resources WAR file to WebSphere Portal Server* on page 124.

Step 9 Create places and pages in WebSphere Portal Server to display Get-Resources portlets. See *Configuring WebSphere Portal Server places and pages* on page 124.

Step 10 Enable edit rights for Get-Resources portlets. See *Enabling edit rights for Get-Resources portlets* on page 125.

When complete, your installation will have the following configuration:

Alternate WebSphere Portal Server configuration

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

Step 1 Review the WebSphere Portal Server installation requirements. See *WebSphere Portal Server installation requirements* on page 120.

Step 2 Generate a Get-Resources WAR file containing the portal components WebSphere Portal Server can display. See *Generating a Get-Resources WAR file* on page 121.

Step 3 Login to the Get-Resources server and stop the WebSphere application server. See *Stopping the WebSphere application server* on page 122.

Step 4 Modify local.xml to change the HTTP authentication method used from Basic to Alternate. See *Modifying the local.xml file* on page 122.

Step 5 Modify web.xml to enable the AuthController servlet. See *Modifying the web.xml file* on page 122.

Step 6 Modify the ibm-web-ext.xmi file to set the fileServingEnabled parameter. See *Modifying the ibm-web-ext.xmi file* on page 123.

Step 7 Modify setDomain.js to call the SetDomain function. See *Modifying the setDomain.js file* on page 123.

Step 8 Start the WebSphere application server. See *Starting the WebSphere application server* on page 124.

Step 9 Deploy the Get-Resources WAR file to WebSphere Portal Server. See *Deploying the Get-Resources WAR file to WebSphere Portal Server* on page 124.

Step 10 Create places and pages in WebSphere Portal Server to display Get-Resources portlets. See *Configuring WebSphere Portal Server places and pages* on page 124.

Step 11 Enable edit rights for Get-Resources portlets. See *Enabling edit rights for Get-Resources portlets* on page 125.

Step 12 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 126.

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-Resources 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-Resources compatible application server
    - Web server
    - Back-end database for Get-Resources
    - An installation of Get-Resources

Generating a Get-Resources WAR file

In order to display Get-Resources in WebSphere Portal Server, you must first export the Get-Resources 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-Resources WAR file:

1 Login to the Get-Resources 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-Resources `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-Resources WAR file.

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

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

4 Click **Generate WAR file.**

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

Stopping the WebSphere application server

To continue configuring, you must log in to the Get-Resources server and stop the WebSphere application server.

To stop the WebSphere application server:

1 Login to the Get-Resources server.

2 Stop your WebSphere application server.

Modifying the local.xml file

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

To modify the local.xml file:

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

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

2 Add the following on a separate line anywhere between <settings> and <\settings>:

```
<httpauthclass>HttpAlternateAuthenticationManager</httpauthclass>
```

3 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>\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 ibm-web-ext.xmi file
You need to set the fileServingEnabled parameter to true to handle static content.

To modify the ibm-web-ext.xmi file:

1 Using a text editor, open the ibm-web-ext.xmi file. The default file path is:

`c:\WebSphere\AppServer\installedApps\getit.ear\getit.war\WEB-INF`

2 Find the fileServingEnabled parameter and set it to **true**.

`fileServingEnabled="true"`

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-Resources server.

2 Stop your application server.

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

   <*application server*>**//oaa/js.**

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

   ```
   setDomain();
   ```

5 Save the file.

Starting the WebSphere application server

To continue configuring, you must restart the WebSphere application server.

Deploying the Get-Resources WAR file to WebSphere Portal Server

After you deploy the Get-Resources 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-Resources 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-Resources WAR file.

5 Click Next to load the Get-Resources 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

Note: Refer to your WebSphere Portal documentation for details on the following.

You can deploy Get-Resources 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-Resources portlets that you display in a page must grant "all authenticated users" the minimum edit permission.

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

To enable edit rights for Get-Resources 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-Resources 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-Resources.

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-Resources 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-Resources is installed.

5 Save the file.

Configuring WebSphere Translation Server for Get-Resources

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

To configure WebSphere Translation Server for Get-Resources:

Step 1 Copy the file wts.jar to the Get-Resources deployment folder. See *Copying wts.jar to the Get-Resources deployment folder* on page 126.

Step 2 Configure Get-Resources to use the WebSphere Translation Server. See *Configuring WebSphere Translation Server for Get-Resources* on page 126.

Copying wts.jar to the Get-Resources deployment folder

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

To copy wts.jar to the Get-Resources 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-Resources deployment folder located at:

<Application server install>/WEB-INF/lib

5 Restart your application server.

Configuring Get-Resources to use the WebSphere Translation Server

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

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

1 Login to the Get-Resources 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.

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-Resources data in a mixture of languages, you can configure Get-Resources 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-Resources 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-Resources* on page 126.

The translate button appears in the upper right tool bar.

The Translation button.

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

Click on the text you want to translate.

3 Click the translate button.

The Translation window opens.

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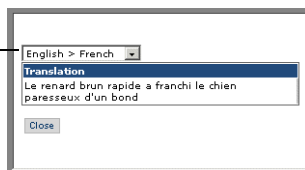
4 Select the target language to which you want to translate from the drop down select box.

The translation of your selection displays in the Translation box.

WebLogic 6.1 SP3 or SP4

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

To configure WebLogic 6.1 SP3 or SP4:

Step 1 Stop both WebLogic and your Web server. See *Stopping WebLogic* on page 129.

Step 2 Edit the startWebLogic.cmd file to set the system password, memory settings, and start mode. See *Editing startWebLogic.cmd* on page 129.

Step 3 Run the Get-Resources installer. See *Custom Installation Option* on page 150

Step 4 Create a virtual directory for Get-Resources in your Web server. See *Creating a virtual directory for Get-Resources* on page 130.

Step 5 Restart WebLogic and your Web server. See *Restarting the servers* on page 131.

Stopping WebLogic

Before you begin to configure WebLogic, you must stop both 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.

Running the Get-Resources installer

Run the Get-Resources installer and select the Custom installation option. See *Custom Installation Option* on page 150.

Creating a virtual directory for Get-Resources

To run Get-Resources, 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-Resources virtual directory

Requirement	Setting
Create virtual directory	`<oaa>`
Map to physical path	`<WebLogic>/applications/oaa`
Directory access rights	Read, Run scripts, Execute

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

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

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

Restarting the servers

Restart WebLogic and your Web server for your new settings to take effect.

JRun 3.1

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

To configure JRun 3.1:

Step 1 Install a Java run-time environment. See *Installing a Java run-time environment* on page 132.

Step 2 Install JRun from the Macromedia Web site to the root of your hard drive. See *Installing JRun* on page 132.

Step 3 Apply the latest JRun update. See *Applying the latest JRun update* on page 133.

Step 4 Deploy the Portal WAR file to JRun to create the necessary folder structure for Get-Resources. See *Deploying the Portal WAR file to JRun* on page 133.

Step 5 Run the Get-Resources installer. *Running the installer* on page 136.

Step 6 Move js.jar to the Java development kit ext folder. See *Moving js.jar to the Java development kit* on page 136.

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 137.

Step 8 Configure your JRun Java settings. See *Configuring Java settings* on page 137.

Step 9 Define any library path environmental variables to your back-end databases. See *Defining library path environment variables* on page 140.

Step 10 Create a virtual directory for Get-Resources in your Web server. See *Creating a virtual directory for Get-Resources* on page 141.

Step 11 Restart JRun and your Web server. See *Restarting JRun* on page 142.

Installing a Java run-time environment

The Get-Resources 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 150.

Installing JRun

Before you install Get-Resources, you must install JRun to your root directory.

To install JRun:

1 Browse to the following URL:

http://www.macromedia.com/software/jrun/

2 Click the link **JRun 3.1 Available for Purchase**.

3 Follow the installation instructions provided.

Applying the latest JRun update

Before you install Get-Resources, 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-Resources in your application server. After you have deployed this file to WebLogic you will be ready to run the Get-Resources installer.

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

1 Open the JRun Management Console and log in.

2 Select JRun Default Server > Web Applications.
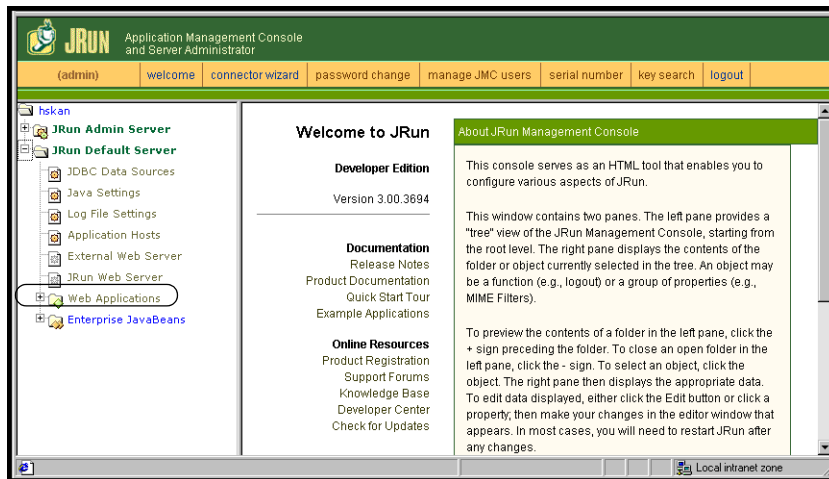
The Edit / Create / Deploy and Remove Applications page opens.
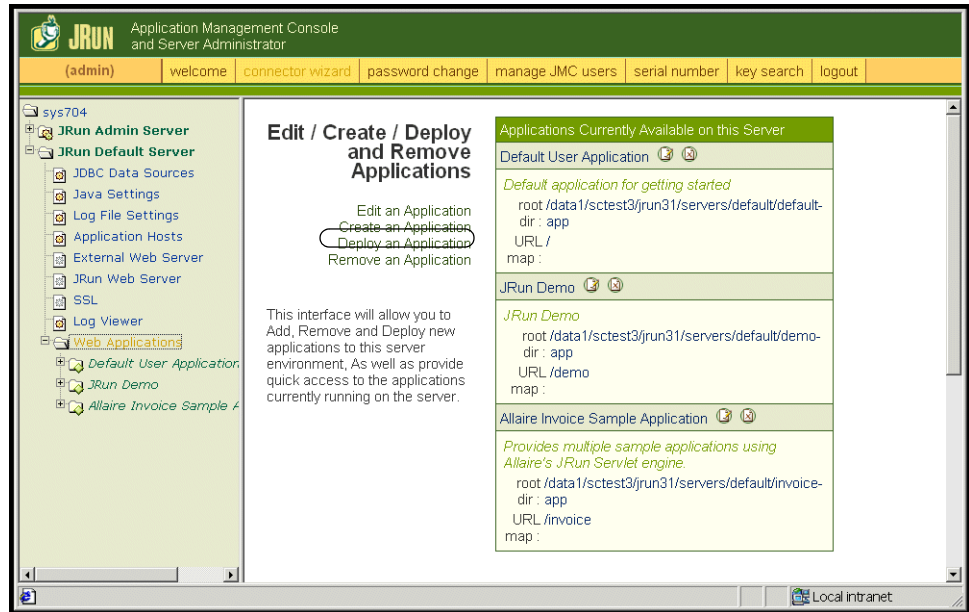
3 Click the **Deploy an Application** link.

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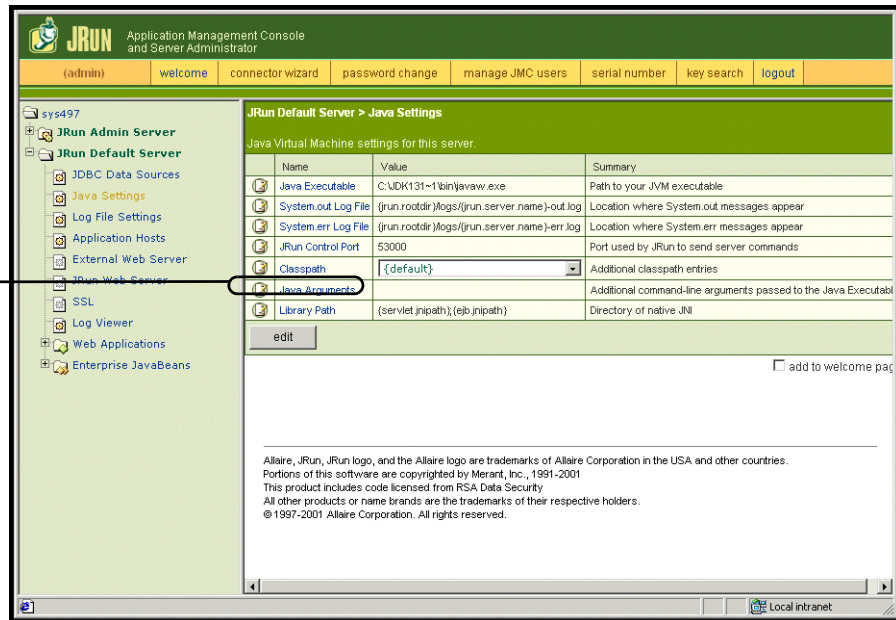
7 In the edit window that opens, use the format example to enter an -Xmx value. This defines the maximum amount of heap memory allocated for your system. It is recommended that you set this value to at least 225 MB, but not more than 512 MB.

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.

Running the installer

Run the Get-Resources installer using the Custom install option. See *Custom Installation Option* on page 150.

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-Resources, 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**.

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 using AssetCenter, add:
  - AssetCenter /**bin** directory
  - Oracle /**lib** directory to the path.
- If you are running Get-Resources on Solaris, add:
  - /usr/bin
  - /usr/ucblib

5 Click **update**.

JRun displays the updated library paths.

The updated path appears here.
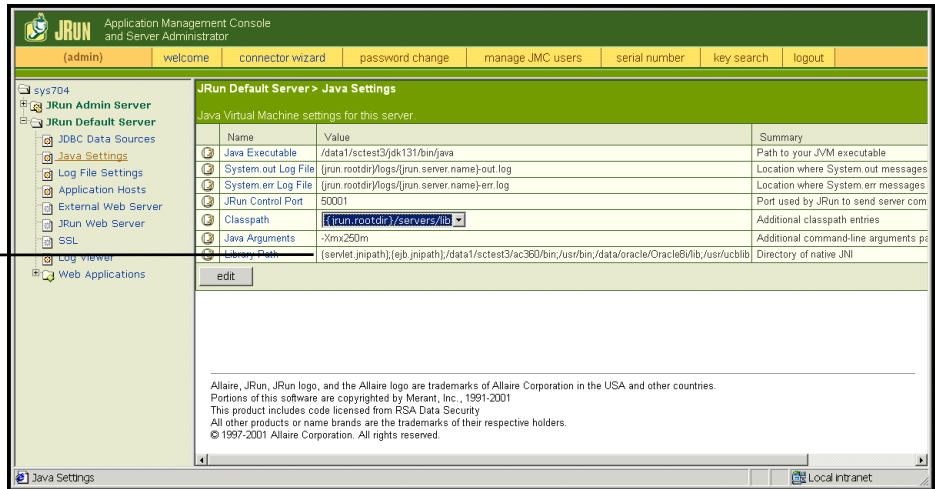
6 Logout of the Management Console.

Creating a virtual directory for Get-Resources

To run Get-Resources, 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-Resources virtual directory

Requirement	Setting
Create virtual directory	<oaa>
Map to physical path	<JRun>/oaa
Directory access rights	Read, Run scripts, Execute

For <*oaa*>, enter the name of the virtual directory you want to use for Get-Resources. 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

Restarting JRun

Restart JRun and your Web server for your new settings to take effect.

Typical Installation Option

A typical installation of Get-Resources 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-Resources:

Applications and File Locations

Get-Resources 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-Resources	/usr/local/peregrine/oaa

Communications Ports

Get-Resources uses the following communications ports in a typical installation. After installation, you can configure Get-Resources 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

Default Port	Component used by	Alternate Port
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-Resources.

Typical Installation Procedures

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

Note: If you cancel the installation before completing all the steps, you must run Uninstall to remove all the files.

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

1 Log into your server with an account that has root privileges.

Important: Verify that your temp directory has a minimum of 300 MB of available space. On Solaris, for example, the system-wide temp directory is /tmp.

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

Exit the automatic launch and 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 2.6, 7, 8, 9	`./setupsolaris`

The installer welcome page opens.
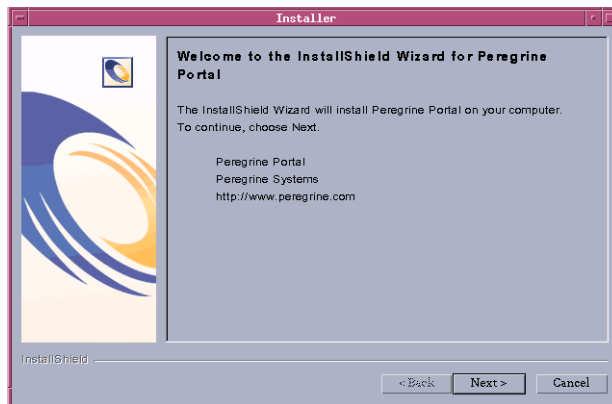
3 Click **Next** to continue to the next page of the wizard.

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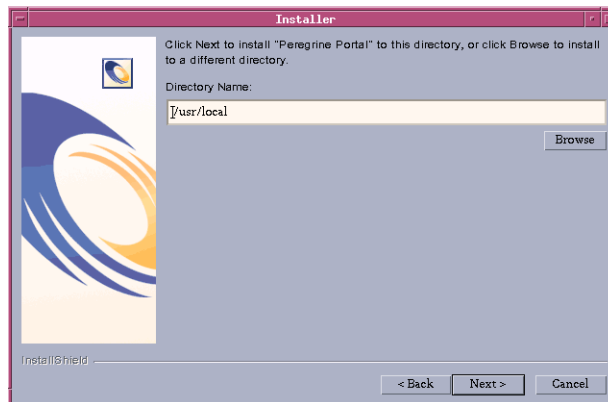
4 Click **Browse** to change the default installation location of **/usr/local**.

5 Click **Next** to open the next page of the wizard that instructs you to stop your application server and Web server.
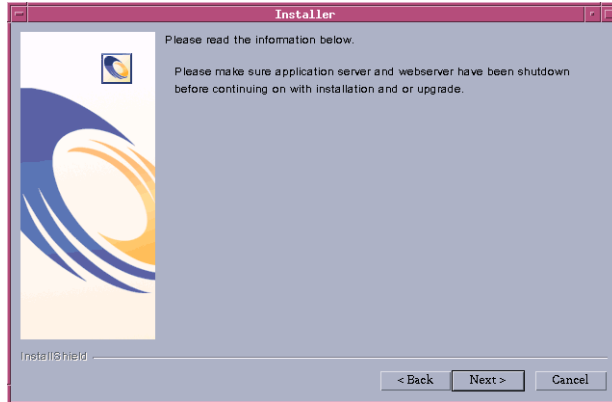
6 Click **Next** to continue to the next page of the wizard.

The setup type page opens.
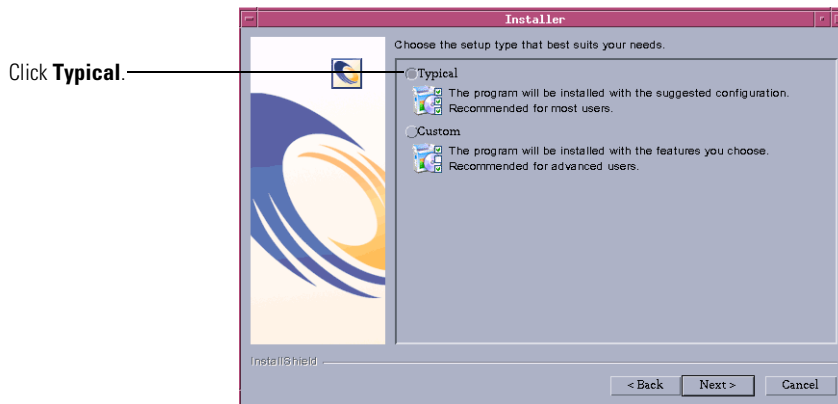
Click **Typical**.

7 Select **Typical**.

8 Click **Next** to open the list of components that will be installed.
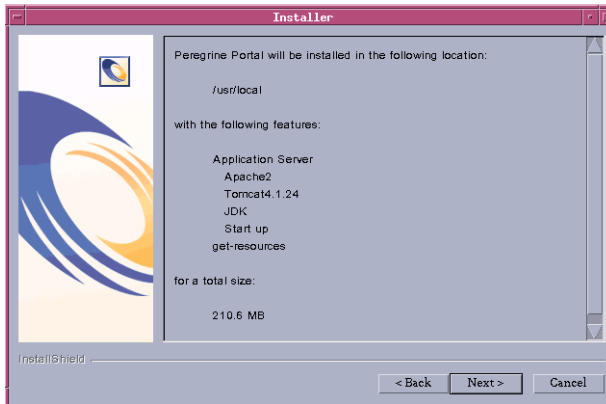
Note: The list depends on the application that you install.

9 Click **Next** to continue installing Get-Resources components.

The installation progress page opens.
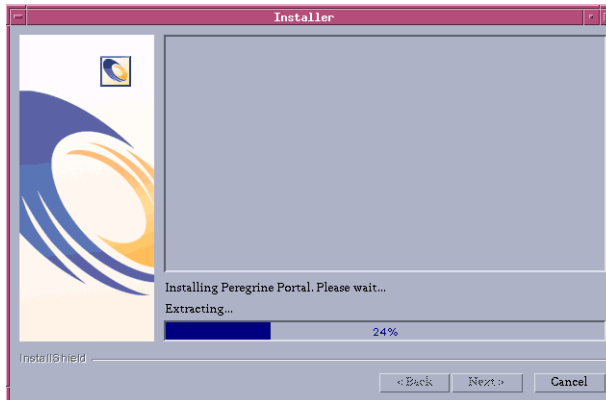
The installer verifies the availability of port 80 for the Apache Web server. If the installer finds a port conflict on port 80, the WebServer Port page opens.
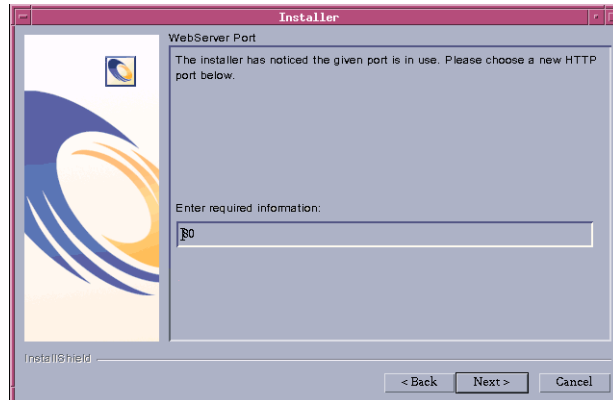
10 If required, enter the new Web server communications port, then click **Next**.

11 Change the Tomcat memory settings as needed, and click **Next**.
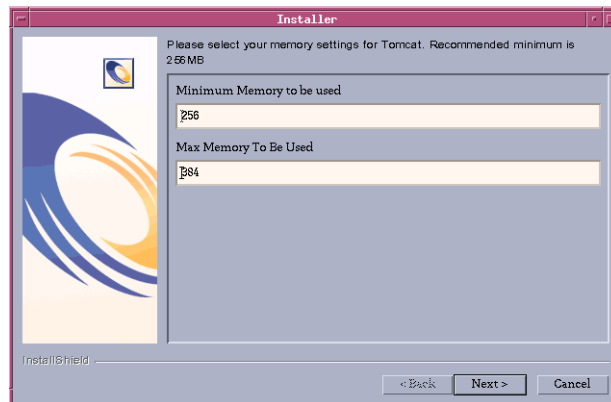
12 Click **Next** to open the deployment utility page that starts deploying Get-Resources components.
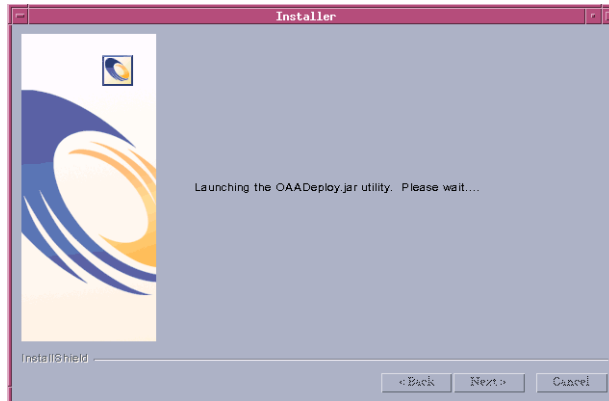
13 Click **Next** to view the list of all deployed packages.
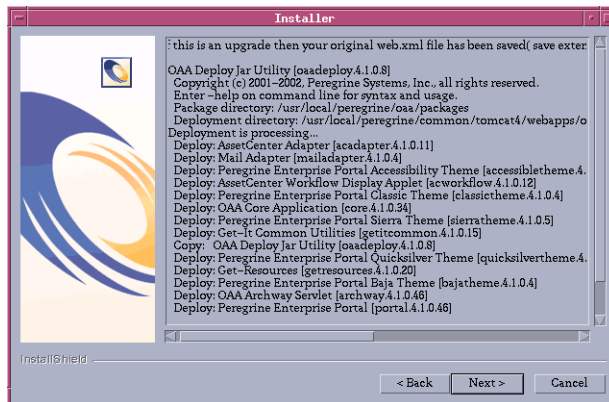
Note: The list of deployed packages depends on the application that you install. Click **Next** to continue.

The start OAA page opens.

Select **Yes**
to start
Get-Resources
immediately.
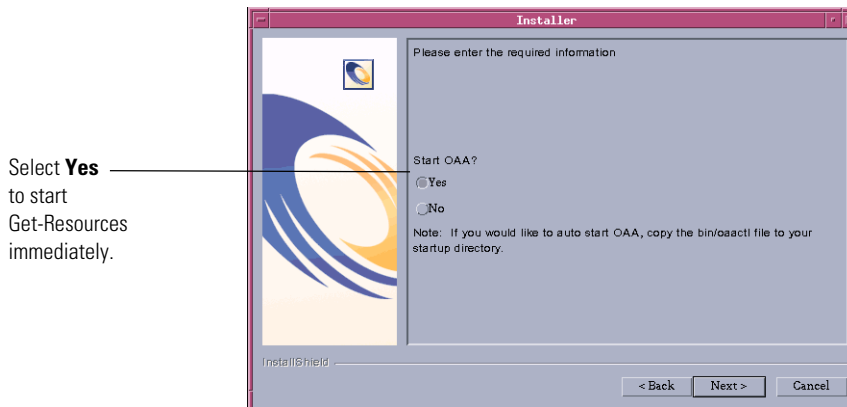
14 Click **Yes** to start Get-Resources immediately or select **No** to manually start Get-Resources after installation is complete.

If you want Get-Resources to start every time the server is started, then copy the file **oaactl** into your startup directory. The default file location is: /usr/local/peregrine/bin/.
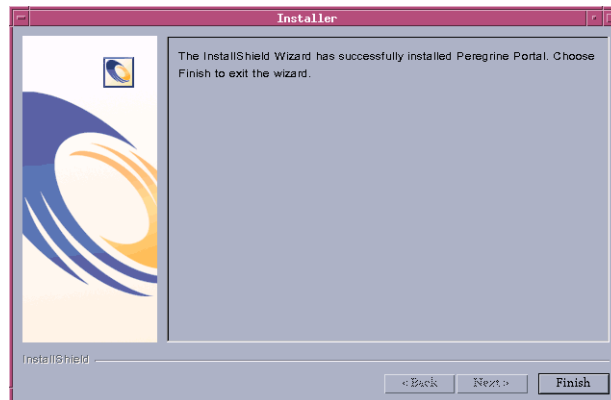
15 Click **Finish** to complete the Get-Resources installation.

If you have not already done so, you 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.

Custom Installation Option

The following section describes how to perform a custom installation of Get-Resources 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-Resources:

Application options

Get-Resources Component	Options
Web Server	■ Apache 2.0.43 ■ IBM HTTP Server 1.3.19 ■ Microsoft IIS 5.0 for Win 2000
Application Server	■ Tomcat 4.1.24 ■ WebSphere 4.02, 5.0 ■ WebLogic 6.1 SP3 ■ JRun 3.1
Java Development Kit	Java 2 SDK

Communications Ports

The communications ports used by a custom installation of Get-Resources 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-Resources to use alternate communications ports if your local network already uses particular communications ports.

Get-Resources on servers running Oracle 9.2.0.1

If you are running Get-Resources 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

Note: If you cancel the installation before completing all the steps, you must run Uninstall to remove all the files.

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

1 Log into your server with an account that has root privileges.

Important: Verify that your temp directory has a minimum of 300 MB of available space. On Solaris, for example, the system-wide temp directory is /tmp.

---

2 Insert the Get-Resources 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.
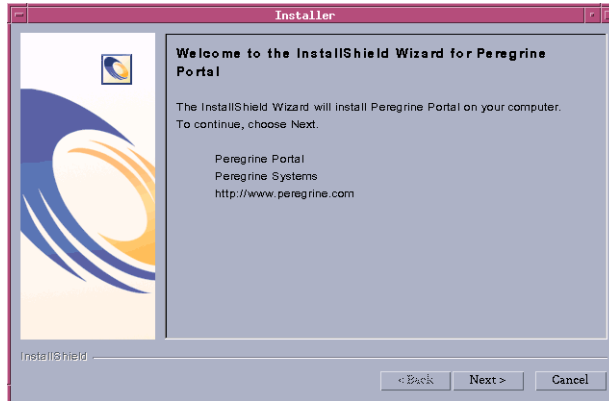
3 Click **Next** to continue to the next page of the wizard.

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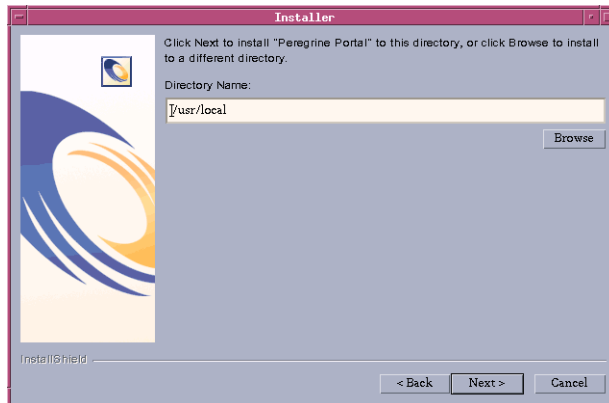
4 Click **Browse** to change the default installation location of **/usr/local**.

5 Click **Next** to read the information about closing the servers before continuing with the installation.
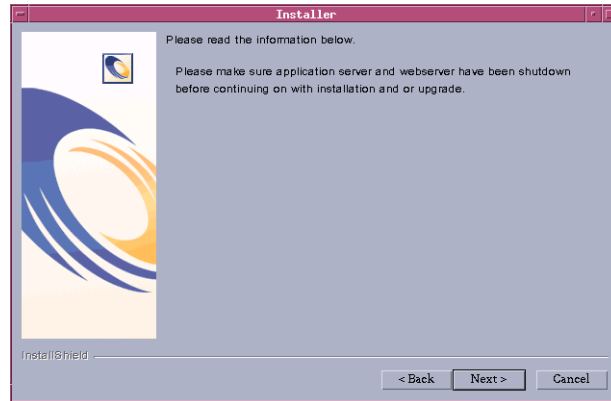
6 Click **Next** to open the setup type page.
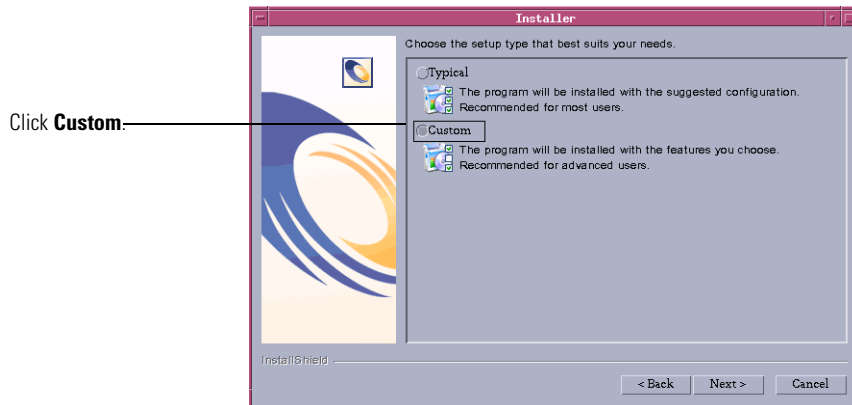
Click **Custom**

7 Select **Custom**.

8 Click **Next** to select the Peregrine Portal features that you want to install.

Select the desired components from the list provided.
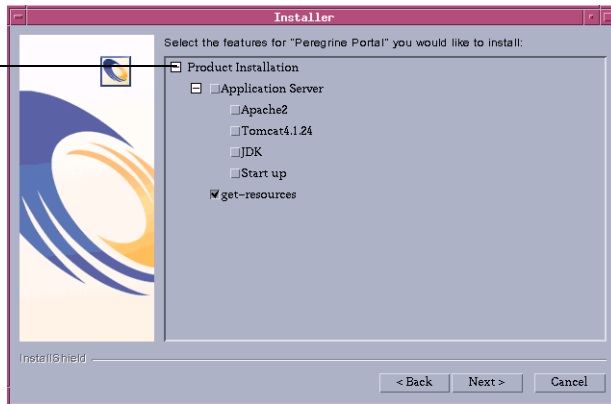
The review components page opens.
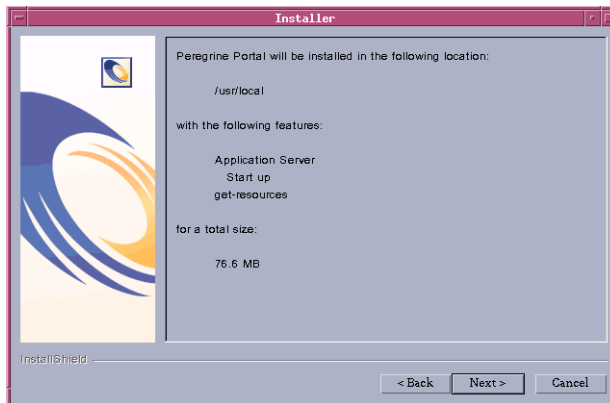
Note: The list of features on this page depends on what you actually install.

9 Click **Next** to start installing Get-Resources components.

The installation progress page opens.
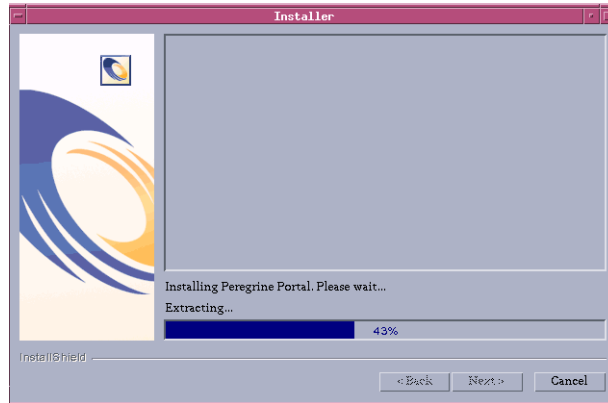
Note: If you are installing multiple applications, you see the following
message. Click **Yes to All**.
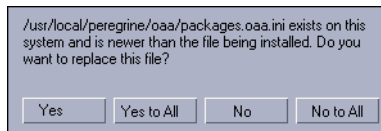
10 Click **Next** to select the Application Server you want to configure.

Select your application
server from the list provided.
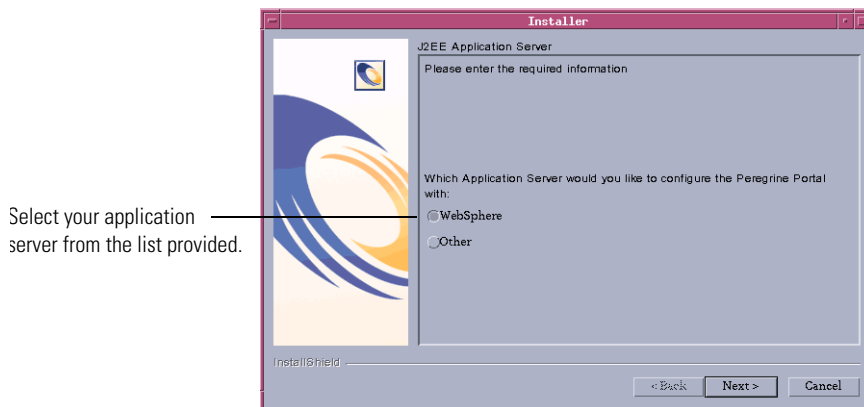
11 Click **WebSphere** to configure a WebSphere application, or click **Other** to
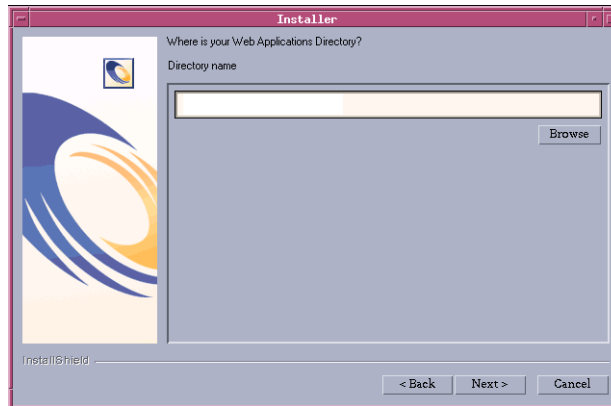configure another application server. Click **Next** to continue to the next page
of the wizard.

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

The WebSphere AppServer installation location page opens.
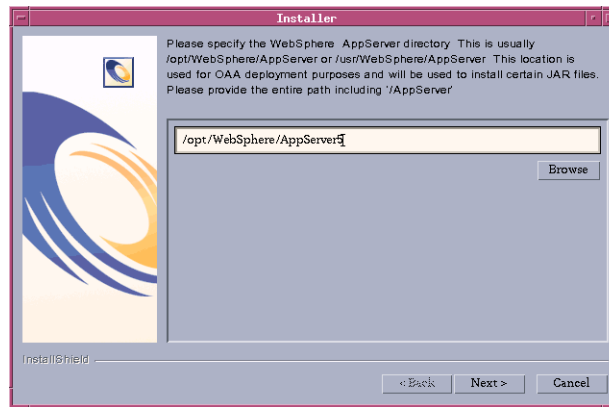
a Click **Browse** to locate the directory where you installed the WebSphere AppServer. Click **Next** to continue.
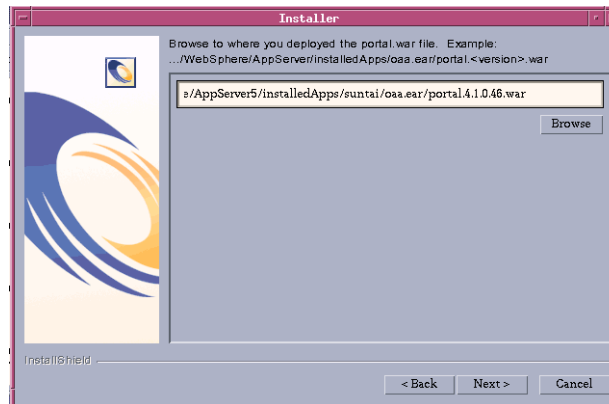
b Click **Browse** to locate the directory where you deployed the portal.war file. Click **Next** to continue.

WebSphere automatically created this directory when you deployed the Get-Resources portal.war as an enterprise application. See *WebSphere Application Server 4.0.2* on page 106 or *WebSphere Application Server 5.0* on page 114 for more information on deploying a WAR file.

The WebSphere JDK installation location page opens.
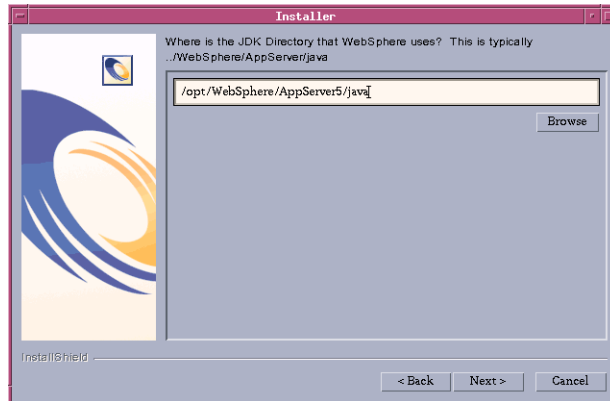
c Click **Browse** to locate the directory where you installed the Java development kit used by WebSphere. Click **Next** to read the information on the screen.
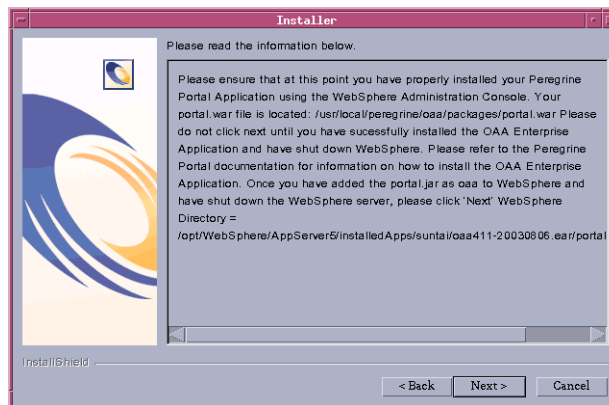
d Click **Next** to continue.

The Get-Resources deployment utility page opens.
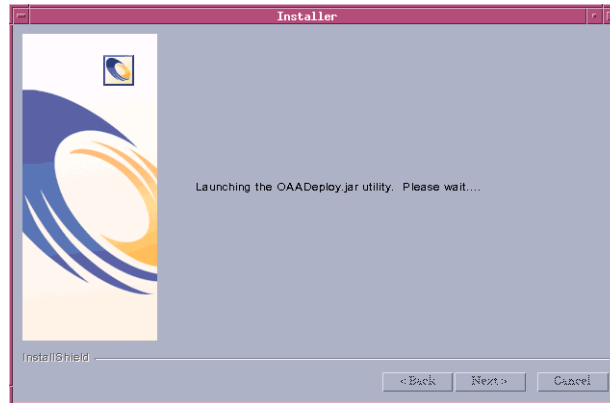
e Click **Next** to view the list of all deployed packages.
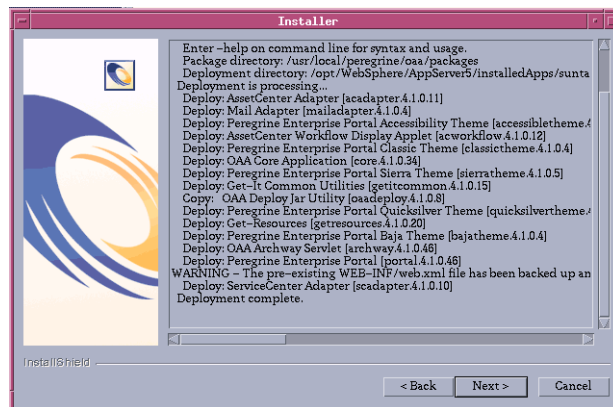
Note: The list of deployed packages depends on what you actually install. Click **Next** to continue.

f Click next to open the IBM HTTP Server location page.
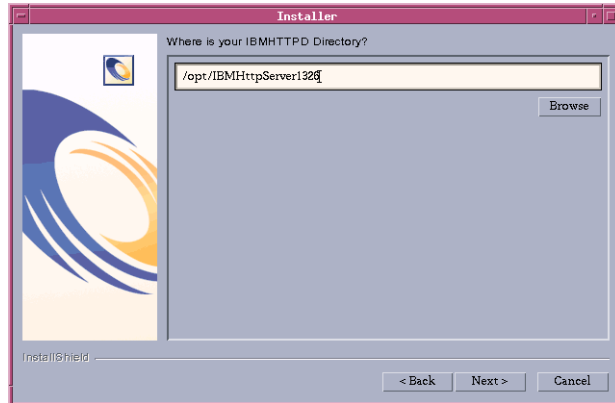
g Click Browse to locate where you installed the IBM HTTP Server. Click **Next** to continue.

The Get-Resources installer will automatically configure a Web server virtual directory called **oaa**. If you want to define a different Web server virtual directory, see *WebSphere Application Server 4.0.2* on page 106 for a list requirements.

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

The other application server information page opens.
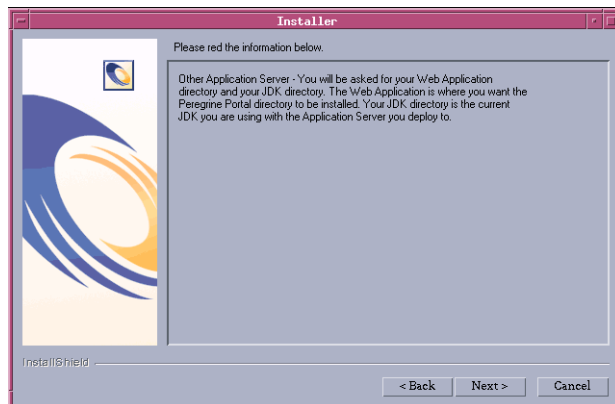
h Click **Next** to configure your application server.

The Web applications directory page opens.

i 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 103.

The application server JDK installation location page opens.
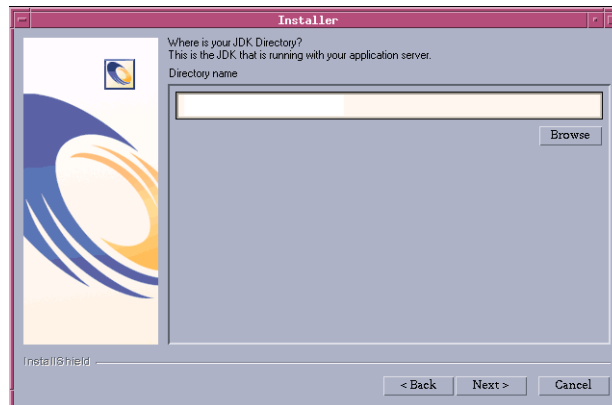
j Click **Browse** to locate the directory where you installed the Java development kit used by your application server. Click **Next** to continue.

12 Click Finish to close the installer.
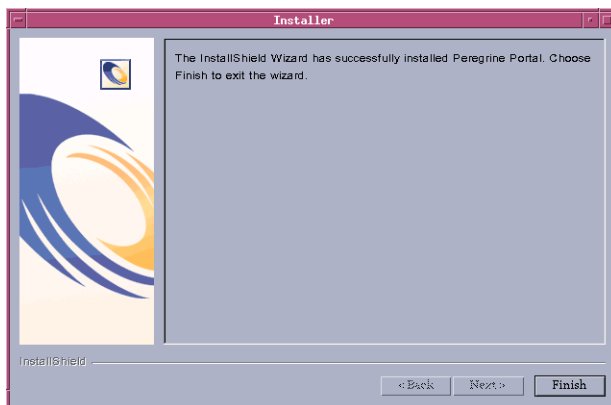
13 Enter the library path environment variables required by your operating system and application server.

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-Resources.

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-Resources.

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 path 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/oaa.ear/portal.4.
0.0.55.war/WEB-INF/lib/AIX:/WebSphere/AppServer/installedApps/answer.
ear/portal.4.0.0.55.war/WEB-INF/lib/AIX/ServiceCenter4
export LIBPATH
```

3 Save the file.

See the *Get-Resources Configuration* chapter in this guide for installing and configuring the ACAdapter on UNIX.

Uninstall—AIX, Linux, or Solaris

Use the following instructions to uninstall Get-Resources.

Warning: These procedures remove all the components that you selected to install. If you chose the Typical installation option, uninstall removes Get-Resources, Peregrine Tomcat, Apache, and JDK. If you chose the Custom installation option, then only those components that you selected to install are removed.

To uninstall Get-Resources 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-Resources:

./uninstall.bin

Note: You must stop the servers before proceeding. Type the command: ./ooactl stop.

4 Follow the on-screen instructions to complete the uninstall.

Testing your installation

Use the following steps to confirm that you have properly installed Get-Resources on AIX or Solaris.

To test your Get-Resources 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-Resources 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-Resources administration login page does not open, see *Troubleshooting* for more information.

4 Load Balancing

CHAPTER

This chapter covers the following topics:

- *Load balancing application servers* on page 168
- *Creating multiple instances of Tomcat for Apache* on page 170
- *Creating multiple instances of Tomcat for IIS* on page 181

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-Resources 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:
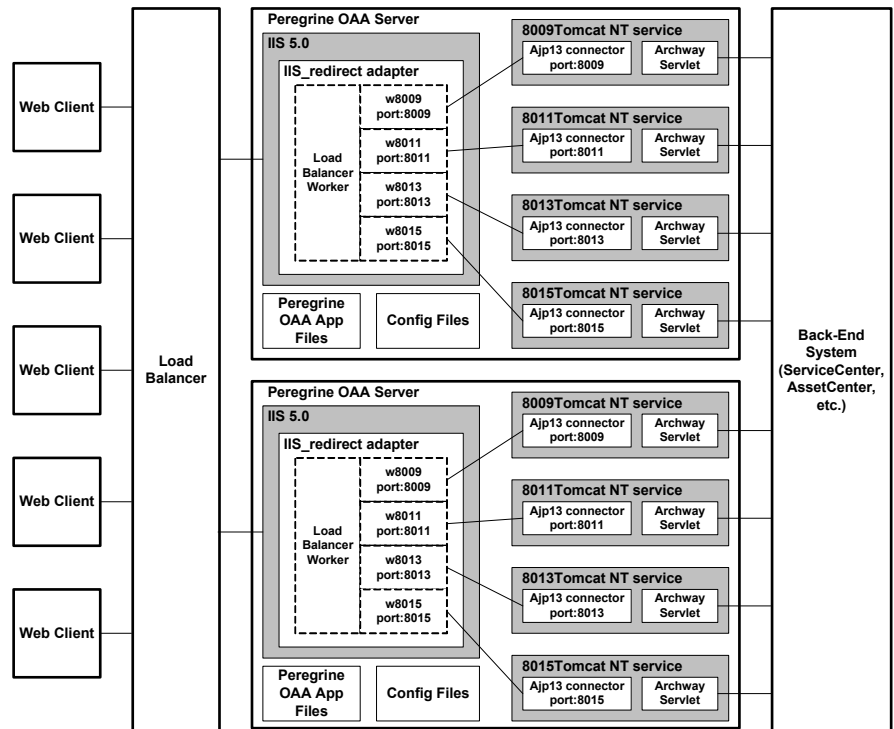
Note: A white paper on Peregrine OAA architecture and optimization is available on the Customer Support Web site at http://support.peregrine.com.

Creating multiple instances of Tomcat for Apache

You can create multiple instance of Tomcat to load balance requests to Get-Resources. You can configure each instance of Tomcat as a service. 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.

Note: The following procedures assume that you have already installed Get-Resources. Refer to either the Windows or UNIX installation chapter for more information on installing Get-Resources.

For systems using IIS, see *Creating multiple instances of Tomcat for IIS* on page 181.

To create multiple Tomcat instance for Apache:

Step 1 Log in to the Get-Resources admininstration page and disable the script pollers setting. See *Disabling script pollers on the primary Tomcat instance* on page 171.

Step 2 Create copies of the Tomcat directory, then delete the \webapps\oaa directory from the newly copied instances of Tomcat. See *Copying the Tomcat directory* on page 171.

Step 3 Edit the workers.properties file of the first or primary Tomcat instance to set the values for each additional Tomcat instance. See *Editing the workers.properties file* on page 172.

Step 4 Edit the mod_jk.conf file of the first or primary Tomcat instance to establish a connection between Tomcat and Apache. See *Editing the mod_jk.conf file* on page 174.

Step 5 Edit the httpd.conf file to define the Tomcat workers available for Apache. See *Editing the httpd.conf file* on page 175.

Step 6 Edit the server.xml files for each Tomcat instance. See *Editing the server.xml files for Apache* on page 175.

Step 7 Edit the jk2.properties files for each Tomcat instance. See *Editing the jk2.properties files for Apache* on page 177.

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 178.

Step 9 Log in to the Get-Resources admininstration page for the primary Tomcat instance and enable the script pollers setting. See *Enabling script pollers on the primary Tomcat instance* on page 179.

Step 10 Testing the configuration. See *Testing load balancing on Apache* on page 179.

Disabling script pollers on the primary Tomcat instance

You only need one Tomcat instance running script pollers. Before you copy your primary Tomcat instance, you should login to the Get-Resources administration page and turn off script polling. This will disable script polling on all of the Tomcat instances you create by copying the primary Tomcat instance.

To disable script pollers on the primary Tomcat instance:

1 Log in to the Get-Resources administration page. The default URL is:

http://*<server_name>*/oaa/admin.jsp

2 Click Settings.

Get-Resources displays the common settings page.

3 Scroll down to the Server-Side Scripts section, and select No for the Enable scipt pollers option.

4 Scroll down to the bottom of the form and click Save.

Get-Resources displays the Control Panel page.

5 Click Reset Server to commit your changes.

6 Log out of the Get-Resources administration page.

7 Stop the Peregrine Tomcat service to temporarily disable Get-Resources.

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. The default file path is:

C:\Program Files\Peregrine\Common\Tomcat4

2 Paste a copy into the same root path. The default file path is:

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 can create 3 copies of the Tomcat folder called \Tomcat4_8011, \Tomcat4_8013, and \Tomcat4_8015. The primary instance uses port 8009.

Warning: If you are using more than four Tomcat instances, change the port numbers to avoid conflicts.

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, ajp13, w8011, w8013, w8015
```

Find the worker.loadbalancer.type=lb code and make changes to the line that follows as shown below.

```
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=ajp13, 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.w8011.port=8011
worker.w8011.host=localhost
worker.w8011.type=ajp13
worker.w8011.cachesize=40
worker.w8011.lbfactor=10
```

Note: All Tomcat instances share this workers.properties file; therefore, all additional lines must be in the file for the primary Tomcat instance.

4 Update the last two lines in the Default ajp13 Worker Definition section.

The first three lines are already in the file.

```
worker.ajp13.port=8009
worker.ajp13.host=localhost
worker.ajp13.type=ajp13
.
.
.
worker.ajp13.lbfactor=10
worker.ajp13.cachesize=40
```

Find the `worker.ajp13.lbfactor=10` code and make changes as shown below

Change lbfactor from =1 to =10
Change cachesize from =10 to =40

5 Save the file.

Editing the mod_jk.conf file

The mod_jk.conf 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-Resources files, otherwise the Get-Resources mount points, file locations, and directories will not be included in the mod_jk.conf file, and you will have to manually add them.

To edit the mod_jk.conf file:

1 Make a copy of the mod_jk.conf file and rename the copy to mod_jk.conf-local.

The mod_jk.conf file is located in the Tomcat conf directory.

Note: This is done only on the primary Tomcat instance.

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:\Program Files\Peregrine\Common\Tomcat4
\conf\worker.properties"
```

4 Change all JkMounts to use *loadbalancer* instead of *default worker ajp13*.

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.

The default location is:
C:\Program Files\Peregrine\Common\Apache2\conf.

2 Update the following line to include -local:

```
include "<Tomcat>/conf/mod_jk.conf-local"
```

For <*Tomcat*>, enter the path to your Tomcat installation. The default file path is: C:\Program Files\Peregrine\Common\Tomcat4

3 Save the file.

Editing the server.xml files for Apache

You will need to modify the server.xml file for each Tomcat instance. 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.

Tip: Make a back up copy of the server.xml file before editing.

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 Verify that the port number attribute of the <Server> element is a unique value that does not conflict with other port numbers used by Tomcat. It is recommended that the port numbers 8005-8008 be used for the shutdown port when configuring four Tomcat instances.

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

Note: This is not the worker communications port number. The worker port number is defined in step 4 on page 176.

> **Warning:** If you are using more than four Tomcat instances, change the port numbers to avoid conflicts.

3 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:

```
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<!--
<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" />
-->
```

4 Update the port number used by the Coyote Connector to a unique, non-conflicting value. If you are configuring four Tomcat instances, the values 8009 (as the primary port), 8011, 8013, and 8015 are recommended.

Example:

```
<!-- Define a Coyote/JK2 AJP 1.3 Connector on port 8009 -->
<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" />
```

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

Example:

```
<!-- Define the top level container in our container hierarchy -->
<Engine jvmRoute="localhost:8009" name="Standalone"
defaultHost="localhost" debug="0">
```

The port number should follow the convention used elsewhere in the configuration (8009, 8011, and so on). These entries must be the same as the Tomcat ID entries you added to the **workers.properties** file in the primary Tomcat instance.

6 Update the appBase attribute of the <Host> element with the absolute path to the **webapps** directory of the primary Tomcat instance.

Example:

```
<!-- Define the default virtual host -->
<Host name="localhost" debug="0"
appBase="C:\Program Files\Peregrine\Common\Tomcat4\webapps"
unpackWARs="true" autoDeploy="true">
```

7 Create a <Context> element entry in the first or primary Tomcat instance and copy it to the other Tomcat instances, changing the OAA context so that it is not reloadable.

This prevents Tomcat from reloading the servlet without restarting the service. It improves performance and helps keep the JSP code that the Tomcat instances are serving in sync during an update.

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 primary Tomcat instance.

8 Save the file.

9 Repeat step 2 through step 7 for each **server.xml** file in each Tomcat instance you made.

Editing the jk2.properties files for Apache

You will need to modify the **jk2.properties** file for each Tomcat instance. 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 instance 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 assign to 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.

Note: The <service name> cannot have a space in it.

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

Warning: The command to create Tomcat instances cannot accept spaces in the file path.

Example:

```
installservice Tomcat8009 C:\Progra~1\Peregrine\Common\Tomcat4_8009
C:\Progra~!\Peregrine\Common\jdk1.3.1_05\jre\bin\server\jvm.dll
```

Note: Use the Windows naming convention to avoid problems of spaces in the file path name. For example, replace Program Files with Progra~1.

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

> **Tip:** You can easily remove a service. From the DOS command prompt, change directories to your Tomcat `bin` directory, then enter the following command: `Tomcat -uninstall <service name>`. The command is case-sensitive.

4 Start each Tomcat service that you install.

Enabling script pollers on the primary Tomcat instance

You only need one Tomcat instance running script pollers. Before you test your load balancing configuration, you should login to the Get-Resources administration page of the primary Tomcat instance and turn on script polling.

To enable script pollers on the primary Tomcat instance:

1 Log in to the Get-Resources administration page of the Tomcat instance. The default URL is:

http://*<server_name>*:*<port_number>*/**oaa/admin.jsp**

For *<port_number>*, enter the port number you have defined for your primary Tomcat instance. This is typically port 8009.

2 Click Settings.

Get-Resources displays the common settings page.

3 Scroll down to the Server-Side Scripts section, and select Yes for the Enable script pollers option.

4 Scroll down to the bottom of the form and click Save.

Get-Resources displays the Control Panel page.

5 Click Reset Server to commit your changes.

6 Log out of the Get-Resources administration page.

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-Resources.

3 Perform an action in Get-Resources. For example, perform a search.

4 Logout of Get-Resources.

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 create multiple Tomcat instance for IIS:

Step 1 Log in to the Get-Resources admininstration page and disable the script pollers setting. See *Disabling script pollers on the primary Tomcat instance* on page 182.

Step 2 Create copies of the Tomcat directory, then delete the \webapps\oaa directory from the newly copied instances of Tomcat. See *Copying the Tomcat directory* on page 182.

Step 3 Configure the ISAPI Plugin for IIS. See *Configuring the ISAPI Plugin for IIS* on page 183.

Step 4 Create and configure a jakarta virtual directory in IIS. See *Creating and configuring a jakarta virtual directory in IIS* on page 184.

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 184.

Step 6 Create and configure an oaa virtual directory in IIS. See *Creating and configuring an oaa virtual directory in IIS* on page 185.

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 186.

Step 8 Edit the server.xml files for each Tomcat instance. See *Editing the server.xml files for IIS* on page 187.

Step 9 Edit the jk2.properties files for each Tomcat instance. See *Editing the jk2.properties files for IIS* on page 189.

Step 10 Install multiple instances of Tomcat as a service using installservice.bat. This file is in the Tomcat\bin directory. See *Installing Tomcat instances as services for IIS* on page 189.

Step 11 Log in to the Get-Resources admininstration page for the primary Tomcat instance and enable the script pollers setting. See *Enabling script pollers on the primary Tomcat instance* on page 190.

Step 12 Testing the configuration. See *Testing load balancing on IIS* on page 191.

Disabling script pollers on the primary Tomcat instance

You only need one Tomcat instance runningscript pollers. Before you copy your primary Tomcat instance, you should login to the Get-Resources administration page and turn off script polling. This will disable script polling on all of the Tomcat instances you create by copying the primary Tomcat instance.

To disable script pollers on the primary Tomcat instance:

1 Log in to the Get-Resources administration page. The default URL is:

   http://*<server_name>*/oaa/admin.jsp

2 Click Settings.

   Get-Resources displays the common settings page.

3 Scroll down to the Server-Side Scripts section, and select No for the Enable scipt pollers option.

4 Scroll down to the bottom of the form and click Save.

   Get-Resources displays the Control Panel page.

5 Click Reset Server to commit your changes.

6 Log out of the Get-Resources administration page.

7 Stop the Peregrine Tomcat service to temporarily disable Get-Resources.

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. The default file path is: C:\Program Files\Peregrine\Common\Tomcat4

2 Paste a copy into the same root path. The default file path is: 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.

Warning: If you are using more than four Tomcat instances, change the port numbers to avoid conflicts.

---

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-Resources 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. The default file path is:
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
Map to physical path	<Tomcat>\bin
Directory access rights	Read, Run scripts, Execute

For *<Tomcat>*, enter the path to your Tomcat installation. The default file path is: C:\Program Files\Peregrine\Common\Tomcat4\bin. This path must contain the isapi_redirector2.dll file.

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 From Windows Control Panel > Administrative Tools, 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. The default file path is: C:\Program Files\Peregrine\Common\Tomcat4\bin\isapi_redirector2.dll

6 Click **OK**.

7 From the Internet Services management console, right-click the **Default Web Site** node, then select **Properties**>**Isapi Filters** again.

The ISAPI filter in IIS displays a green status arrow to indicate that it is running.

8 Close the Internet Services management console.

Note: You must stop and then start the IIS service for changes to take effect. You must also restart Tomcat.

Creating and configuring an oaa virtual directory in IIS

To run Get-Resources 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>`
Map to physical path	`<Tomcat>\webapps\oaa`
Directory access rights	Read, Run scripts

For <*oaa*>, enter the name of the virtual directory you want to use for Get-Resources. The recommended virtual directory name is oaa. If you choose to use another virtual directory name, you must enter the new name in the following places:

- Rename the folder `<Tomcat>\webapps\oaa` to `<Tomcat>\webapps\<new name>`
- Rename the [uri] mappings in workers2.properties from oaa to the new virtual directory name.
- Rename all the oaa context entries in mod_jk2.conf from oaa to the new virtual directory name.
- Rename the <Context> path and docBase attributes in server.xml from oaa to the new virtual directory name.

Important: The virtual directory name you choose will become part of the URL users enter to connect to Get-Resources. For example: http://server_name/<new name>/login.jsp

---

For <*Tomcat*>, enter the path to your Tomcat installation. The default file 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 is shared by all other Tomcat instances on that particular server.

The workers2.properties file specifies the worker threads that the Web server connector creates 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.

To edit the worker2.properties file:

1 Open the workers2.properties file, located in the conf directory of your primary 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 where the Tomcat instance is located.

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

The 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 is oaa.

If you defined a different virtual directory other than **oaa** to run Get-Resources, 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.

Tip: Make a back up copy of the **server.xml** file before editing.

To edit the server.xml files:

1 Each Tomcat instance has a **server.xml** file located in the conf directory. 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.

Peregrine Systems recommends that you use the port numbers 8005-8008 when configuring four Tomcat instances.

Warning: If you are using more than four Tomcat instances, change the port numbers to avoid conflicts.

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.

Peregrine Systems recommends that you use port numbers 8009, 8011, 8013, and 8015 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 primary Tomcat instance and copy it to the other Tomcat instances.

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 primary Tomcat instance.

5 Update the jvmRoute attribute of the <Engine> element with the server name and communications port used by each Tomcat instance.

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

6 Update the <Host> element with the **webapps** directory used by the first or primary 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 Comment out port 8080 in the non-SSL Coyote HTTP... statement.

Example:

```
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
port="8080" minProcessors="5" maxProcessors="75"
acceptCount="10" debug="0" connectionTimeout="20000"
useURIValidationHack="false" />
-->
```

8 Save the file server.xml.

9 Repeat step 2 through step 8 for each copy of the server.xml file you made.

Editing the jk2.properties files for IIS

You will need to modify the jk2.properties file for each Tomcat instance. 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 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 instance 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 the bin directory of your Tomcat instance.

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 assign to 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.

Note: The <service name> cannot have a space in it.

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

Warning: The command to create Tomcat instances cannot accept spaces in the file path.

Example:

```
installservice Tomcat8009 C:\Progra~1\Peregrine\Common\Tomcat4
C:\Progra~1\Peregrine\Common\jdk1.3.1_05\jre\bin\server\jvm.dll
```

Note: Use the Windows naming convention to avoid problems of spaces in the file path name. For example, replace Program Files with Progra~1.

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

Tip: You can easily remove a service. From the DOS command prompt, change directories to the bin directory of your Tomcat instance, then enter the following command: Tomcat -Uninstall <service name>.

4 Start each Tomcat instance that you install.

Enabling script pollers on the primary Tomcat instance

You only need one Tomcat instance running script pollers. Before you test your load balancing configuration, you should login to the Get-Resources administration page of the primary Tomcat instance and turn on script polling.

To enable script pollers on the primary Tomcat instance:

1 Log in to the Get-Resources administration page of the Tomcat instance. The default URL is:

http://*<server_name>*:*<port_number>*/oaa/admin.jsp

For *<port_number>*, enter the port number you have defined for your primary Tomcat instance. This is typically port 8009.

2 Click Settings.

Get-Resources displays the common settings page.

3 Scroll down to the Server-Side Scripts section, and select Yes for the Enable scipt pollers option.

4 Scroll down to the bottom of the form and click Save.

Get-Resources displays the Control Panel page.

5 Click Reset Server to commit your changes.

6 Log out of the Get-Resources administration page.

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-Resources.

3 Perform an action in Get-Resources. For example, perform a search.

4 Logout of Get-Resources.

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 Back-end System Configuration

CHAPTER

Get-Resources can be configured for use with AssetCenter or ServiceCenter. The back-end authenticates user access rights and stores personalization changes.

- With AssetCenter as the back-end, Get-Resources accesses the AssetCenter Procurement module. A sample AssetCenter database is included with the installation of Get-Resources. There are also import scripts for setting up an existing AssetCenter database.

Important: With ServiceCenter as the back-end, Get-Resources accesses ServiceCenter Request Management. To continue the installation for Get-Resources, an administrator must perform the procedures in this chapter within the back-end system. For detailed information on how to perform any of the steps listed in the following sections, refer to the *ServiceCenter System Administrator's Guide or the AssetCenter Administration Guide*.

This chapter includes the following sections:

- *Configuring AssetCenter for Get-Resources* on page 194
- *Configuring ServiceCenter for Get-Resources on page 201*

Configuring AssetCenter for Get-Resources

As requests are opened and processed through Get-Resources, data is stored and updated in the AssetCenter database.

Users are authenticated using AssetCenter profiles, with special user rights provided for use with Get-Resources.

Using the Get-Resources AssetCenter demo database

The AssetCenter Demo database included with the installation already contains the necessary updated tables, workflows and user rights that will enable AssetCenter to perform as a back-end system for Get-Resources.

Important: Although the Get-Resources installation includes an AssetCenter demo database, AssetCenter must be installed separately. If not already installed, during either a full or custom AssetCenter installation, select the AssetCenter API option. Refer to your AssetCenter documentation for more detailed instructions for this procedure.

Follow these steps to configure the AssetCenter Demo database installed with Get-Resources as a back-end system:

Step 1 Create a new connection (User DSN only) to the demo database installed with Get-Resources for the version of AssetCenter that you are running.

The default path to the demo database is:

C:\Program Files\Peregrine\Common\<appsrv>\WEB-INF\etc\en

where *<appsrv>* is the name of the application server you are using with Get-Resources (that is, Tomcat 4.1).

Back-end System	Default Folder	Database Filename
AssetCenter 3.60	ac3	am360.db
AssetCenter 4.10	ac4	am410.db
AssetCenter 4.11	ac4	am411.db

Back-end System	Default Folder	Database Filename
AssetCenter 4.2x[1]	ac4	am411.db
AssetCenter 4.3	ac4	am411.db

> 1When using the am411.db demo database with AssetCenter 4.2x, you may receive a warning message stating that the database version is compatible with the software version, but the database is not quite up to date. Refer to the *AssetCenter Migration Guide* for instructions on updating the database.

Step 2 If your application server is configured to run as a service, set up a System DSN.

Step 3 Test the connection by logging in to the database from AssetCenter.

Step 4 Configure settings within Get-Resources for connectivity to the AssetCenter back-end system.

Using an existing AssetCenter database

If you intend to use an existing AssetCenter database as a back-end system, you will need to import scripts which will update the necessary tables, workflows and user rights that AssetCenter needs to work with Get-Resources.

The type of scripts that you need to import will vary depending on whether or not your existing AssetCenter database has been configured to work with a previous version of Get-Resources.

This section includes instructions on how to import the scripts necessary for your version of AssetCenter in addition to a list of script locations and a description of script functions.

Configuring an AssetCenter database that has never been used with Get-Resources

Follow these steps to import the scripts appropriate for your version of AssetCenter:

Step 1 Login to AssetCenter and connect to the existing database.

Step 2 Import the scripts provided with the Get-Resources installation.

By default, these files are located in:

C:\Program Files\Peregrine\Common\<appsrv>\WEB-INF\etc\<lang>

...ac3 folder for AssetCenter 3.6

...ac4 folder for AssetCenter 4.x

where *<appsrv>* is the name of the application server you are using with Get-Resources (that is, Tomcat 4.1) and *<lang>* is the ID for the localized script (that is, en for English).

Important: The scripts must be imported in the order outlined here, according to the version of AssetCenter that you are running.

AssetCenter 3.x:

- From the getit import folder, import the following in the order listed:
  - feat.scr
  - calcflds.scr
- From the getit import folder, import the following scripts in the order listed:
  - profile.scr
  - workflow.scr
  - empldept.scr

AssetCenter 4.x:

- From the essential folder, import the following:
  - essential.scr

- From the sample folder, import the following:
  - profile.scr
  - workflow.scr
- If you are using the Procurement module, from the procurement subfolder under the sample folder, import the following:
  - profile.scr
  - workflow.scr
- From the demo folder, import the following:
  - empldept.scr

- reqtypes.scr

Configuring an AssetCenter database that has been used with a previous version of Get-Resources

Login to AssetCenter and import the scripts necessary for your AssetCenter version from the following path:

C:\Program Files\Peregrine\Common\<appsrv>\WEB-INF\etc\<lang>\

where *<appsrv>* is the name of the application server you are using with Get-Resources (for example, Tomcat 4.1) and *<lang>* is the ID for the localized script (for example, en for English).

Version	Configured for	Path and script filename
AssetCenter 3.6	Get-Resources 2.0.1 or 2.5	ac3/getit import/profile.scr
AssetCenter 4.x	Get-Resources 2.5	ac4/sample/profile.scr

Script Locations

The following table lists the default folder and filename for each script:

Back-end System	Default Folder	Script
AssetCenter 3.6	getit import	feat.scr calcflds.scr profile.scr workflow.scr empldept.scr
AssetCenter 4.x	essential	essential.scr (including calcflds.scr and feat.scr). This file also imports the itemized list values that Get-Resources uses for the request status.
	sample	profile.scr workflow.scr
	demo	empldept.scr reqtypes.scr
AssetCenter 4.x Procurement module	sample/procurement	profile.scr workflow.scr

Script Descriptions

The following table lists the contents and functions of each script:

Script	Description
feat.scr	Features for table amEmployee; contains DelegatedApproverId and DelegatedApprExpiration scripts. These features delegate request approvals to a person given by DelegatedApproverId, until the date specified by DelegatedApprExpiration. Features for table amProduct include ManufacturerURL, a text-based feature containing the manufacturer's URL. Others include Architecture, Processor, ProcSpeed, RAM, and HDCapacity, used by the Description calculated field to enhance the product description, none of which are used directly by Get-Resources.
calcflds.scr	Calculated field Description for the Product table. Gives a technical description for the product.
profile.scr	Imports user rights which correspond to Get-Resources capability words to control access to the screens. These include: getit.requester (access to the request) getit.advancedrequester getit.approver getit.reserve The final user right, getit.receiver, imports three predefined user profiles. The user profiles and their user rights are: **getit.admin** = getit.admin + getit.requester + getit.advancedrequester + getit.approver + getit.reserve **getit.default** = getit.requester **getit.full** = getit.requester + getit.advancedrequester + getit.approver + getit.reserve This script is provided as an example of how profiles can be defined. You can modify this as needed.

Script	Description
workflow.scr	Imports necessary workflows to make Get-Resources work out-of-box. Listed in order of Reference, Name, and Description.
	BUNDLPO; Bundle Ordering; Workflow necessary to work with the bundles
	REQROUTE; Routing Request; Sample workflow that manages the full qualification of an off catalog request before it enters the approval cycle
	REQAPPR_GR; RequestApproval; Sample approval workflow. It is important to note that only the workflow with the **REQAPPR_GR** reference will be displayed in Get-Resources.
	REQSTATUS; Request Status; Workflow that, on fulfillment of the request, changes its status to "Received" for better Get-Resources visibility.
	The **REQSTATUS** script also imports three employee workgroups. These include:
	■ **IT**: Technical approvers group (any employee in this group can give a technical approval for a request, provided one has the getit.approver user right).
	■ **Finance**: Financial approvers group (any employee in this group can give a financial approval for a request, provided one has the getit.approver user right).
	■ **Getit**: Employees group that will qualify the off catalog request.
	Note: The off catalog request qualification must be performed in AssetCenter. There is no module for this task in Get-Resources.
	This script is provided as an example of how workflows can be defined. You can modify the workflows as needed. Refer to the *Get-Resources Administration Guide* for more information about the AssetCenter workflows provided with Get-Resources.

Script	Description
empldept.scr	This script is optional and is considered demonstration data. It imports one employee per profile, as defined in the following:
	Valentine: Michael Valentine is a getit.admin.
	Hartke: Richard Hartke is a getit.full.
	Tossi: Michaela Tossi is a getit.default.
reqtypes.scr	This script enables the AssetCenter 4.x demo database to work with out-of-box menus. It imports the new request types: Hardware, Software, Paid Time Off, Employee Promotion.

AssetCenter Procurement module script descriptions

The following table lists the contents and functions of each script for the AssetCenter Procurement module:

Script	Description
profile.scr	Imports user rights that correspond to Get-Resources capability words to control access to the screens.
	These include:
	getit.requester (access to the request)
	getit.advancedrequester
	getit.approver
	getit.receiver
	getit.pcardmanager
	getit.pcarduser
	getit.buyer
	getit.reserve

Script	Description
profile.scr (cont)	The final user right, getit.receiver, imports three predefined user profiles.
	The user profiles and their user rights are:
	getit.admin = getit.admin + getit.requester + getit.advancedrequester + getit.approver + getit.receiver + getit.pcardmanager + getit.pcarduser + getit.buyer + getit.reserve
	getit.default = getit.requester
	getit.full = getit.requester + getit.advancedrequester + getit.approver + getit.receiver + getit.pcardmanager + getit.pcarduser + getit.buyer + getit.reserve
	This script is an example of how profiles can be defined. You can modify this as needed.
workflow.scr	Imports necessary workflows to make Get-Resources work out-of-box. Listed in order of Reference, Name, and Description.
	GRCREATEMISSINGREQ; Create Missing Requests; Creates a request and request lines from orphan order lines of the current purchase order
	POGEN; Automatic PO Generation; Automatically creates POs from approved requests
	This script is an example of how workflows can be defined. You can modify the workflows as needed. Refer to the *Get-Resources 4.1 Administration Guide* for more information about the AssetCenter workflows provided with Get-Resources 4.1.

Configuring ServiceCenter for Get-Resources

Get-Resources accesses the catalog in the Request Management module of ServiceCenter. Requests opened in Get-Resources are stored as quotes in the ServiceCenter database.

Users are authenticated using ServiceCenter Operator records, with special capability words provided for use with Get-Resources. Refer to the Security chapter in the *Get-Resources Administration Guide* for more information about user authentication.

Applying the unload files to ServiceCenter

The next step in the Get-Resources installation involves installing updated RAD codes, event register records, event maps, and processes that are contained in unload files. You apply different unload files to ServiceCenter 4.x, ServiceCenter 5.0.x, and ServiceCenter 5.1.

Warning: This procedure overwrites existing ServiceCenter records when you apply the new unload files. If you tailored ServiceCenter, you will lose changes made to the original records.

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.
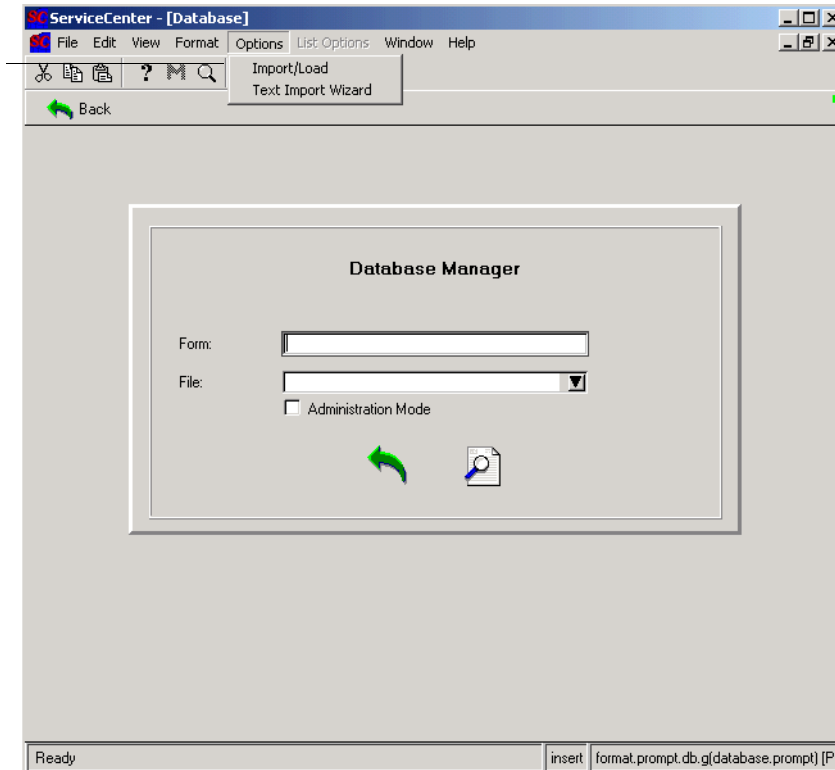
2 From **Options**, choose Import/Load to open the **File Load/Import** dialog box.
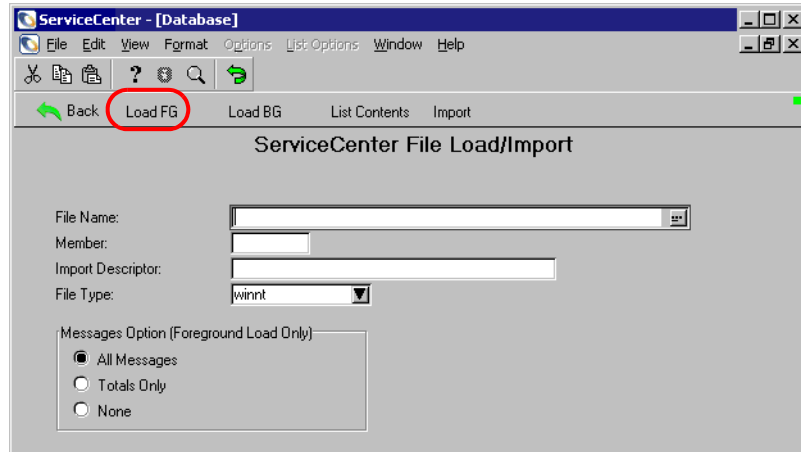
3 In the **Filename** text box, browse to, or type, the path for the unload file that pertains to your particular integration.

The default location of the unload file is:
<appsrv>\web-inf\etc\sc*<version number>* where *<appsrv>* is the name of the application server you are using with Get-Resources.

4 Click **Load FG** and note any errors.

5 Repeat step 3 and step 4 for each unload file that pertains to your integration.

Using ServiceCenter 4.x with Get-Resources 4.1

In order to perform as a back-end system for Get-Resources, ServiceCenter 4.x requires the following files. Apply these files in the order listed:

- etc/sc4/esapproval.unl
- etc/sc4/sc4_scr23463.unl
- etc/sc4/sc4_scr26028.unl
- etc/sc4/scr27148.unl

The default location is: *<appsrv>*\WEB-INF\etc\sc4, where *<appsrv>* is the name of the application server you are using with Get-Resources.

Upgrading from Get-Resources 2.0.1 through 4.0.1 to Get-Resources 4.1

To upgrade from Get-Resources 2.01 through Get-Resources 4.0.1 to Get-Resources 4.1 using ServiceCenter 4.x as the back-end system, apply these files in the order listed:

- etc/sc4/upgrade/es_lineitem.unl
- etc/sc4/esapproval.unl
- etc/sc4/sc4_scr23463.unl
- etc/sc4/sc4_scr26028.unl
- etc/sc4/scr27148.unl

The default location is: *<appsrv>*\WEB-INF\etc\sc4, where *<appsrv>* is the name of the application server you are using with Get-Resources.

Using ServiceCenter 5.0x with Get-Resources 4.1

You must apply the following load files to configure ServiceCenter 5.0x for Get-Resources. Follow the instructions in *Applying the unload files to ServiceCenter* on page 202 to load the files:

- etc/sc5/esapproval.unl
- etc/sc5/sc5_scr23463.unl
- etc/sc5/sc5_scr26028.unl
- etc/sc5/scr27148.unl

The default location of these files is: <appsrv>\web-inf\etc\sc5, where *<appsrv>* is the name of the application server you are using with Get-Resources.

Using ServiceCenter 5.1 with Get-Resources 4.1

You must apply the following load files to configure ServiceCenter 5.1 for Get-Resources. Follow the instructions in *Applying the unload files to ServiceCenter* on page 202 to load the files:

- etc/sc51/sc51_scr23463.unl
- etc/sc51/sc51_scr26028.unl
- etc/sc51/scr27148.unl

The default location of these files is: `<appsrv>\web-inf\etc\sc51`, where *<appsrv>* is the name of the application server you are using with Get-Resources.

6 Get-Resources Configuration

CHAPTER

Final configuration settings necessary to complete the installation of Get-Resources 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-Resources.

A more detailed description of additional features necessary for administration and maintenance using the Peregrine Portal Administration utility is available in the *Get-Resources Administration Guide*.

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.

A default administrator, System, 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>\WEB-INF** directory to store these parameters. If you reinstall Get-Resources, make a copy of this file and store it outside your Get-Resources 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.
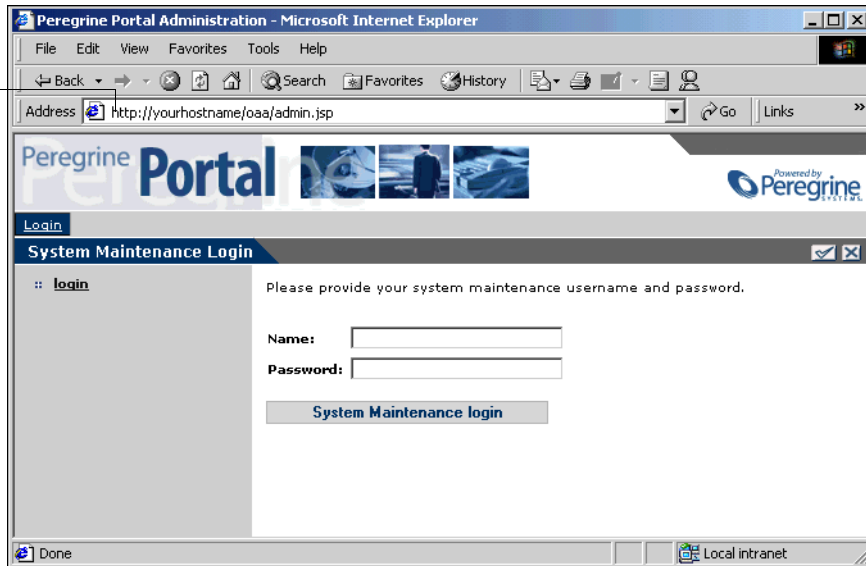
System is the default administrator name.

4 In the Name field, type **System**.

No password is required on initial login. Refer to the *Get-Resources Administration Guide* for detailed instructions on how to change the Administrator login name and password.

5 Click **System Maintenance login** to open the **Control Panel** page.

Use the Admin module to define settings to Get-Resources.

Activity menu

The left pane Activity menu provides access to a number of different administrative utilities for Get-Resources. You need to access some of these during this installation process. The utilities and their functions are described in detail in the *Get-Resources 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-Resources 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.

All changes involving the Control Panel require the following steps:

Step 1 Log in to the Admin page with the username **System**.

Step 2 Go to the Settings page.

Step 3 Make your changes.

Step 4 Reset the server.

Using the Settings page

From the Admin module 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, click **Settings**.

The Settings page opens, providing access to the various tabs used to configure settings for Get-Resources.

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** [**Limited**] link and click **Save**.

The parameter returns to the default state.

Resetting the server

After making any configuration changes in the Peregrine Portal Administration module, the system returns to the Admin Control Panel and the following informational message appears at the top of the page.

The settings have been saved. To ensure all changes take effect, you must reset the Peregrine Portal server.

The following instructions explain how to reset the server and apply your configuration changes to Get-Resources.

To reset the connection between the Archway servlet and back-end system:

1 From Control Panel, click **Reset Server.**

When the operation is complete, 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.

Changes include the addition, removal, or connectivity of the target adapters.

Configuring connections to AssetCenter

This section describes how to configure and verify settings for the following:

- Connection settings for AssetCenter as a back-end.
- ACAdapter on UNIX.

Connection settings

This section describes how to configure and verify back-end system settings.

To verify settings for AssetCenter as a back-end system:

1 From the Peregrine Portal Admin module, click **Settings**.

This displays the Settings page which provides access to the various tabs used to configure settings for Get-Resources.

The Common settings page opens by default.

2 At the top of the page, click the **Get-Resources** tab to open the **Connections Settings** page for Get-Resources:

3 Type **GRRequestDB** to identify the target back-end alias.

Field Name	Value Description	Default Value
Get-Resources Targets	Contains the target aliases for the back-end.	(none)
Procurement Available	Determines AssetCenter Procurement availability	No

To use AssetCenter Procurement, select Yes and click **Save**.

4 If you make any changes, click **Save** to return to the Admin Control Panel page, then click **Reset Server** to apply the configuration settings.

Setting the GRRequestDB adapter

This section describes how to specify the alias target configuration.

To set the GRRequestDB adapter:

1 From the Peregrine Portal Admin module, click **Settings**, then click the **GRRequestDB** tab to open the GRRequestDB settings page.

2 Type **ac** to specify AssetCenter as the alias target configuration.

3 Click **Save** to return to the Admin Control Panel page, then click **Reset Server** to apply the configuration settings.

AssetCenter database settings

This section describes how to connect to the AssetCenter database.

To connect to the AssetCenter database:

1 From the Peregrine Portal Admin module, click **Settings**, then click the **AssetCenter** tab to open the **AssetCenter** settings page.

| AssetCenter | Common | E-mail | Get-Resources | GRRequestDB | Logging | Portal | Portal DB | Themes |
| Web Application | XSL |

Database: ACDemo42en	Name of the AssetCenter database. Click for default: []
Anonymous name: Admin	Anonymous user name used when an unknown user attempts to communicate with AssetCenter.
Anonymous password:	Anonymous user password.
Admin name: Admin	Administration user used by the Peregrine Portal when performing tasks such as user authentication and registration.
Admin password:	Administration password.
AC Shared Library Name:	AssetCenter API shared libarary name. This setting is unused on Windows systems.
AC Shared Library Path:	Path to the AC API shared library. This setting is unused on Windows systems.
Default Capability Words:	Semicolon separated list of access rights that all users should have regardless of their profile. An example would be ac(getit.requester)
Adapter: com.peregrine.oaa.adapter.ac.ACAdapter	Full class path for adapter associated with this target.
Enum Source: WEB-INF/bizdoc/Enum/SysEnums.xml	Specifies the xml file which provides the values for enumeration data types. Leave this blank if the enum values are stored in backend database (i.e. bizdoc is present).

Save

2 Update the following fields:

Field Name	Value Description	Default Value
Database	Name of the AssetCenter database you are using as a back-end for Get-Resources.	(none)
Anonymous name	Anonymous user name used when an unknown user attempts to communicate with AssetCenter.	Admin
Anonymous password	Password for the anonymous user name.	(none)

Field Name	Value Description	Default Value
Admin name	Administration login name used by the Peregrine Portal when performing tasks such as user authentication and registration.	Admin
Admin password	Administrator password.	(none)
AC Shared Library Name	AssetCenter API shared library name.	(none)
AC Shared Library Path	Path to the AC API shared library.	(none)
Default Capability Words	Access rights for all users.	(none)
Adapter	Full class path for adapter associated with this target.	com.peregrine.oaa. adapter.ac.ACAdapter
WEB-INF/bizdoc/Enum /SysEnums.xml	The xml file that provides the values for enumeration data types.	(none)

3 When you finish updating these fields, click **Save** to return to the Admin Control Panel page.

4 Click **Reset Server** to apply the configuration settings.

Setting the Portal DB adapter

Get-Resources enables personalization of portal application screens without the need to manually change and compile code. To enable this feature, Get-Resources requires a database adapter connection to store settings and customizations to the portal interface in the back-end database. Until a database adapter is defined for the portal page, users cannot see or make personalization changes to the Peregrine portal home page.

To configure Get-Resources to save personalization settings in the AssetCenter back-end database:

1 From the Peregrine Portal Admin module, click **Settings**.

2 At the top of the Settings page, click the **Portal DB** tab to open the Portal Database settings page.

3 In the **Alias for** field, type **ac**, then click **Save** to return to the Admin Control Panel page.

4 Click **Reset Server** to apply the configuration changes.

Setting the Web Application database adapter

Get-Resources 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 configure Get-Resources to save personalization to the Web application in the AssetCenter back-end database:

1 From the Peregrine Portal Admin module, click **Settings**.

2 At the top of the Settings page, click the **Web Application** tab to open the **Web Application** settings page.

3 In the **Alias for:** field type **ac**, then click **Save** to return to the Admin Control Panel page.

4 Click **Reset Server** to apply the configuration changes.

> **Important:** Verify that the **Web Application**, **PortalDB** and **ac** targets use the com.peregrine.oaa.adapter.ac.ACAdapter and are *connected* in the Connection Status table.

Troubleshooting the AssetCenter database connection

If you are having trouble making a connection between Get-Resources and the AssetCenter back-end database (that is, the Connection Status displays *disconnected* for the target adapters in the Admin Control Panel), follow these instruction to verify the AssetCenter Database Settings.

To troubleshoot the AssetCenter database connection:

1 From the AssetCenter tab, verify that the parameter defined for the **Database** field is the same as the database name displayed in the **Connect to database** dialog box when you log into AssetCenter.

2 Verify that the correct Admin name and password is defined for Get-Resources to login to AssetCenter.

3 Check the ODBC connections to AssetCenter.

Depending on the way you run the application server, it looks for either an ODBC User DSN or an ODBC System DSN.

- If you start your application server as a service (the default method), it references the System DSN for the ODBC connection to the AssetCenter database.

- If you start your application server as an application, it references the User DSN to determine the ODBC connection.

Refer to *AssetCenter database settings* on page 214 for detailed instructions on how to perform these functions.

Installing and configuring ACAdapter on UNIX

By default, Get-Resources is installed with all of the connectivity required to interface with AssetCenter through a special adapter, the ACAdapter. Use the following procedures to install and configure the ACAdapter on UNIX.

To install and configure the ACAdapter on UNIX:

1 If necessary, follow the AssetCenter documentation to install AssetCenter on the AIX, Linux, or Solaris system where you run Get-Resources.

Note: AssetCenter 4.2 on AIX requires the OpenSSL package from http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html. Click the AIX Toolbox Cryptographic Content link.

2 Copy amdb.ini from a Windows AssetCenter installation that is configured to use the same AssetCenter database.

Note: Peregrine recommends that you copy it to the AssetCenter installation directory. You can copy the file to any location of your choosing where the application server user can read it.

Depending on your AC and Windows versions, this file may be in the WINDOWS or WINNT directory or your user's home directory under Documents and Settings.

3 Set the LD_LIBRARY_PATH environment variable (LIBPATH on AIX) for the user running your application server to include the AssetCenter/bin directory and the directory containing the shared libraries for the database client that AssetCenter is using.

For example: /usr/local/ac421/bin:/usr/local/oracle/product/8.1.6/lib

4 Log in to **admin.jsp**, then click **Settings** and the **AssetCenter** tab.

5 Set the AssetCenter database name to match the database name defined in amdb.ini.

6 Set the AssetCenter library to match the version in your AssetCenter/bin directory.

For example, "aamapi42" for libaamapi42.so

Java will add the prefix and suffix back on when it looks for the file.

7 Optionally, set the AssetCenter library path to your AssetCenter/bin directory, including a trailing slash.

8 Click **Save**.

Configuring connections to ServiceCenter

Get-Resources connects to the ServiceCenter database through a special adapter, the SCAdapter. This section describes how to configure and verify the following settings:

- ServiceCenter adapter
- GRRequestDB adapter
- Portal DB adapter
- Web Application database adapter

Connection settings

This section describes how to configure and verify back-end system settings.

To define ServiceCenter as the back-end system for Get-Resources:

1 From the Peregrine Portal Admin module, click **Settings**.

This displays the Settings page which provides access to the various tabs that configure settings for Get-Resources.

The Common settings page opens by default.

2 Update the fields as required.

The Common tab includes settings for the following:

- Language Translation
- Personalization
- Attachments
- Server-Side Scripts
- Encoding, Locales, and Sessions
- Performance Tuning

Tip: You can reload forms without having to reset your Peregrine OAA server. Set the **Reload Forms** option to **Yes** under Performance Tuning. This works best in a production environment.

Set the Reload Forms option to yes to automatically reload the forms if their sources have been modified after the server has started.

3 At the top of the page, click the **Get-Resources** tab to open the **Connections Settings** page for Get-Resources:

4 Update the following fields as follows:

Field Name	Enter Value	Description
Get-Resources Targets	GRRequestDB	This is an alias that you can set to another alias such as **ac** or **sc**.
Procurement Available	no	The default **no** indicates that AssetCenter Procurement is not available.

5 Click **Save** to return to the Admin Control Panel page.

6 Click **Reset Server** at the bottom of the page to apply your changes to the system.

7 When the operation is complete, verify that the **sc** target adapter, com.peregrine.oaa.adapter.sc.SCAdapter, is displayed in the **Connection Status** table.

> **Note:** Unless the ServiceCenter system you are using as a back-end is configured with default login information, the status for the ServiceCenter target adapter displays *disconnected* until you follow instructions to connect Get-Resources to the ServiceCenter database.

ServiceCenter database settings

Get-Resources requires the ServiceCenter adapter to connect to the ServiceCenter database.

To connect to the ServiceCenter database:

1 From the Peregrine Portal Admin module, click **Settings**.

2 At the top of the Settings page, click the **ServiceCenter** tab to open the ServiceCenter settings page.

3 Update the following fields as necessary:

Field Name	Value Description	Default Value
Host	Host name of the ServiceCenter server.	localhost
Port	Port number of the ServiceCenter server.	12670
Log	The path to SC logging that the ServiceCenter client connection uses	(none)
Admin user	Administration login name that the Peregrine Portal uses when performing tasks such as user authentication and registration	falcon
Admin password	Administrator password	(none)
Anonymous user	Name to use when unknown user attempts to connect with ServiceCenter	falcon
Anonymous password	Anonymous user password for ServiceCenter	(none)
Default capabilities	Access rights for all users.	(none)
Adapter	Full class path for adapter associated with this target	com.peregrine.oaa. adapter.sc.SCAdapter
WEB-INF/bizdoc/Enum /SysEnums.xml	The xml file that provides the values for enumeration data types	(none)

4 When finished, click **Save** at the bottom of the page to return to the Admin Control Panel page.

5 Click **Reset Server** at the bottom of the page to apply your changes to the system.

6 When the operation is complete, verify that the **sc** target adapter, com.peregrine.oaa.adapter.sc.SCAdapter, is displayed in the **Connection Status** table.

Note: Unless the ServiceCenter system you are using as a back-end is configured with default login information, the status for the ServiceCenter target adapter displays *disconnected* until you follow instructions to connect Get-Resources to the ServiceCenter database.

Setting the GRRequestDB adapter

This section describes how to specify the alias target configuration.

To set the GRRequestDB adapter:

1 From the Peregrine Portal Admin module, click **Settings**, then click the **GRRequestDB** tab to open the GRRequestDB settings page.

2 Type **sc** to specify ServiceCenter as the alias target configuration.

Note: AssetCenter (ac) is the default alias.

3 Click **Save** to return to the Admin Control Panel page, then click **Reset Server** to apply the configuration settings.

Setting the PortalDB adapter

Get-Resources enables personalization of portal application screens without the need to manually change and compile code. To enable this feature, Get-Resources requires a database adapter connection to store settings and customizations to the portal interface in the back-end database. Until a database adapter is defined for the portal page, users will not be able to see or make personalization changes to the Peregrine portal home page.

To configure Get-Resources to save personalization settings in the ServiceCenter back-end database:

1 From the Peregrine Portal Admin module, click **Settings**.

2 At the top of the Settings page, click the **Portal DB** tab.

This displays the Portal Database settings page.

3   In the **Alias for** field, type **sc**, then click **Save** to return to the Admin Control Panel page.

Note: AssetCenter (ac) is the default alias.

4   Click **Reset Server** at the bottom of the page to apply your changes to the system.

5   When the operation completes, verify that the adapter used for the **portalDB** target is com.peregrine.oaa.adapter.sc.SCAdapter and displays *connected* in the Connection Status table.

Setting Web Application database adapter

Get-Resources 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 configure Get-Resources to save personalization to the Web application in the ServiceCenter back-end database:

1   From the Peregrine Portal Admin module, click **Settings**.

2   At the top of the Settings page, click the **Web Application** tab to open the
    Web Application settings page.

3   In the **Alias for** field type **sc**, then click **Save** to return to the Admin Control
    Panel page.

    Note: AssetCenter (ac) is the default alias.

4   Click **Reset Server** at the bottom of the page to apply your changes to the
    system.

5   When the operation completes, verify that the adapter used for the
    weblication target is com.peregrine.oaa.adapter.sc.SCAdapter and displays
    connected in the Connection Status table.

Troubleshooting the ServiceCenter database connection

If the Connection Status table in the Admin Control Panel displays
disconnected for any of the target adapters, perform the following procedures
to troubleshoot possible configuration issues:

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-Resources 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.

In the ServiceCenter settings page of the Admin module in Get-Resources:

- Verify that the parameters entered for the **Host:** and **Port:** fields are correct.

- Verify that the Admin user name and password defined for Get-Resources are the same login values used when logging directly into the ServiceCenter back-end system as an Administrator.

Refer to *ServiceCenter database settings* on page 221 for detailed instructions on how to perform these functions.
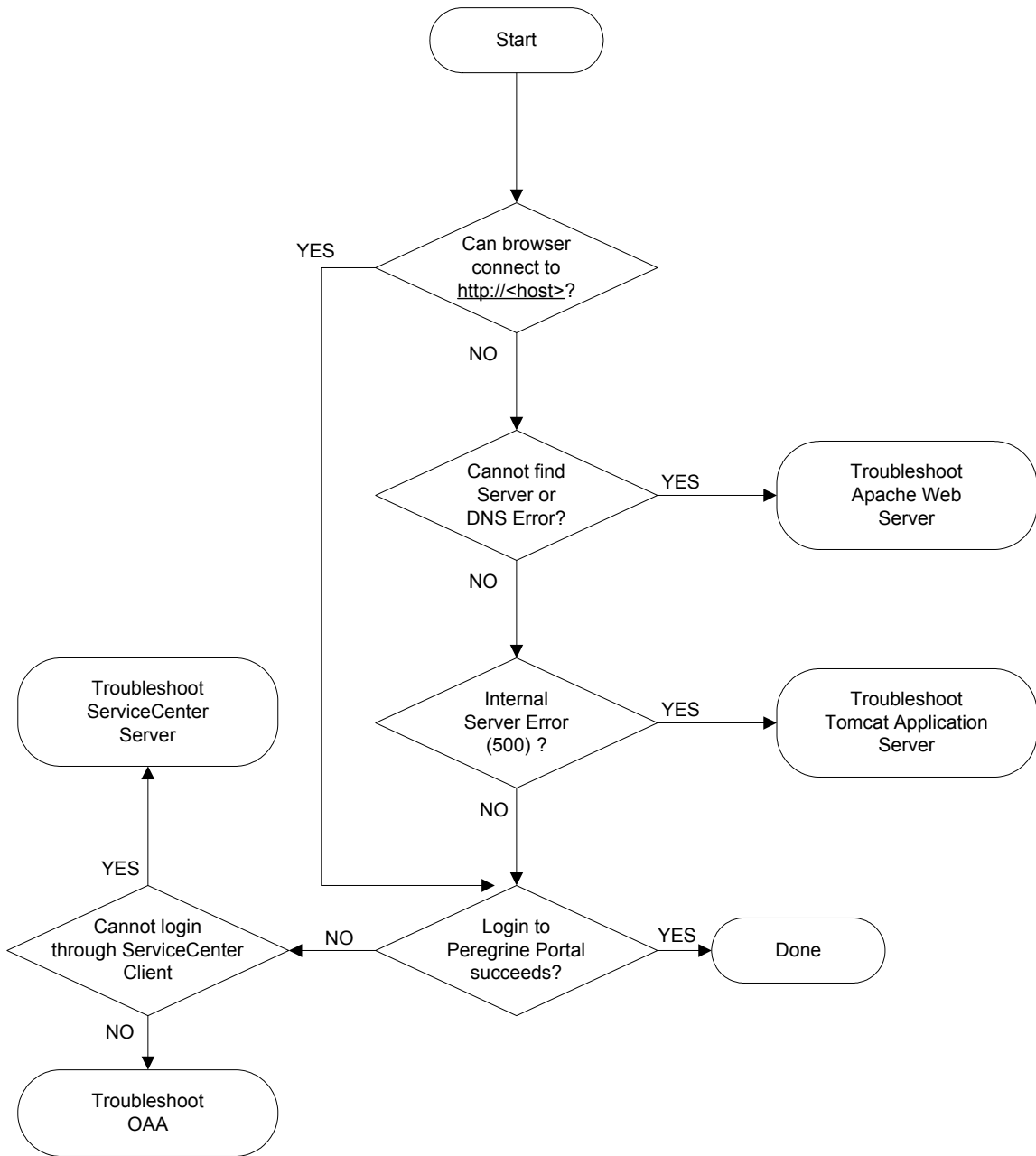
7 Troubleshooting

CHAPTER

This chapter covers the following topics:

The following troubleshooting diagram shows areas to consider when trying to resolve installation problems.

```
                            ┌──────────┐
                            │  Start   │
                            └────┬─────┘
                                 │
                                 ▼
                          ╱ Can browser ╲
                  YES    ╱   connect to   ╲
              ◄─────────◄  http://<host>?  ►
              │          ╲                ╱
              │           ╲      │NO     ╱
              │                  ▼
              │            ╱ Cannot find ╲          ┌──────────────┐
              │           ╱   Server or    ╲  YES   │ Troubleshoot │
              │          ◄   DNS Error?      ►─────►│  Apache Web  │
              │           ╲                ╱         │   Server     │
              │            ╲     │NO      ╱          └──────────────┘
              │                  ▼
  ┌───────────────┐         ╱  Internal  ╲           ┌──────────────────┐
  │ Troubleshoot  │        ╱ Server Error  ╲   YES   │   Troubleshoot   │
  │ ServiceCenter │       ◄    (500) ?       ►──────►│ Tomcat Application│
  │    Server     │        ╲                ╱         │     Server       │
  └───────▲───────┘         ╲     │NO      ╱          └──────────────────┘
          │                        ▼
       YES│
  ╱ Cannot login ╲   NO   ╱  Login to   ╲   YES   ┌──────┐
 ╱ through ServiceCenter◄──◄ Peregrine Portal►───►│ Done │
◄    Client      ►        ╲  succeeds?   ╱         └──────┘
  ╲              ╱          ╲           ╱
   ╲    │NO    ╱
        ▼
  ┌──────────────┐
  │ Troubleshoot │
  │     OAA      │
  └──────────────┘
```

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 the IBM HTTP Server

The Get-Resources installer creates duplicate alias entries in the IBM HTTP Server when you install more than one Peregrine OAA Platform application on WebSphere.

Duplicate entries can also occur if you reinstall Get-Resources or install another Peregrine OAA Platform application on a system that formerly had Get-Resources installed on it.

Remove any duplicate alias entries from the IBM HTTP Server httpd.conf file.

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 Stop the Tomcat service.

2 Use **netstat -a** to list ports being listened on. Check for port conflicts.

3 Make necessary modifications to Tomcat port configuration or disable (or modify) the conflicting service. Additional information about Tomcat is available at http://jakarta.apache.org/tomcat/.

Checking for Port Conflicts: an example

Check for entries where the Proto value is "TCP" and the State is "Listening." For example, the following output from the **netstat -an** command shows that ports 80, 8009, 8025, 12670, and 1585 are in use:

```
Active Connections

Proto   Local Address      Foreign Address  State
TCP     0.0.0.0:8          0 0.0.0.0:0        LISTENING
TCP     0.0.0.0:8009       0.0.0.0:0          LISTENING
TCP     0.0.0.0:8025       0.0.0.0:0          LISTENING
TCP     0.0.0.0:12670      0.0.0.0:0          LISTENING
TCP1    0.2.3.154:1032     66.163.173.77:80 ESTABLISHED
TCP1    0.2.3.154:1342     10.2.3.154:12670 ESTABLISHED
TCP1    0.2.3.154:1585     0.0.0.0:0          LISTENING
TCP1    0.2.3.154:1585     10.2.0.112:139   ESTABLISHED
```

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 the Apache and Tomcat services.

4 Use a browser to connect to the Web server.

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 OAA Web 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 **System** 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 WebSphere

Duplicate alias entries can occur from the IBM HTTP Server **httpd.conf** file during a WebSphere installation. If this happens, the Admin form at **http://hostname/oaa/login.jsp** does not render.

Remove duplicate Alias /oaa lines from the **httpd.conf** file under the **conf** directory of the IBM HTTP or Apache web server. You want only one of the following:

```
Alias /oaa "C:\WebSphere\AppServer\installedApps\oaa.ear\portal.war"
Alias /oaa "C:/WebSphere/AppServer/installedApps/oaa.ear/portal.war"
```

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.

Using ServiceCenter on Oracle

When using ServiceCenter on Oracle as the back-end database, personalization does not display pages correctly. From ServiceCenter, you must map the giComponentUsers table to Oracle.

To map the giComponentUsers table to Oracle:

1 Open the ServiceCenter client.

2 Log in as Administrator.

3 Add a sqlsystemtables record.

   a Click the Toolkit tab to open the Database Manager dialog box.

   b Type **sqlsystemtables** in the File field and click Search.

   c Check the **Map as Blob** flag.

4 Go to the sqlmapping table and delete all records for giComponentUsers table.

5 Map the giComponentUsers table to Oracle.

Contact your ServiceCenter Administrator for more information on updating ServiceCenter table definitions.

Index