Oracle9*i* Application Server Wireless Edition

Configuration Guide

Release 1.1

May 2001

Part No. A86701-02

ORACLE

Oracle9*i* Application Server Wireless Edition Configuration Guide, Release 1.1

Part No. A86701-02

Contents

4 Logging

5 Multiple JVM Support and Load Balancing

6 Oracle Enterprise Manager

List of Tables

Send Us Your Comments

Oracle9*i* Application Server Wireless Edition Configuration Guide, Release 1.1

Part No. A86701-02

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: iasdocs_us@oracle.com
- Postal service:
  Oracle Corporation
  Oracle Mobile and Wireless Products
  500 Oracle Parkway, Mailstop 4OP6
  Redwood Shores, California 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

The Oracle9*i* Application Server Wireless Edition Configuration Guide discusses how to configure Wireless Edition.

This Preface contains these topics:

Audience

Organization

Related Documentation

Documentation Accessibility

Audience

Oracle9*i* Application Server Wireless Edition Configuration Guide is for users who are setting up their environments to administer and / or develop applications for wireless users. This document includes the following topics:

Overview

Setting up Wireless Edition

Setting up Oracle Enterprise Manager

Settings, Utilities, and Properties

To use this document, you need a general knowledge of application servers, Web servers, and database-driven Web sites, wireless technologies, and familiarity with the technologies and programming languages used in middle-tier environments.

Organization

This document contains:

Chapter 1, "Overview"	Overview of post-installation configuration procedures.
Chapter 2, "FastForward"	Configure Mail and LDAP applications.
Chapter 3, "Alerts and SMS Request Listener"	Configuring the SMS Listener to process time-based, scheduled services known as Alerts.
Chapter 4, "Logging"	Configuring the logging capabilities of Wireless Edition.
Chapter 5, "Multiple JVM Support and Load Balancing"	Configuring Wireless Edition to allow multiple Java Virtual Machine instances and load balancing.
Chapter 6, "Oracle Enterprise Manager"	Configuring the Oracle Enterprise Manger for use with Wireless Edition.
Chapter 7, "Multi-byte Character Support"	Multi-byte character support and localization of Wireless Edition.
Chapter 8, "Utilities"	Utilities, such as Password Encryption / Decryption, LoadXML, Upload / Download utilities, Xslt, and CopyObjects.
Chapter 9, "Recommended Settings"	Recommended setting for Mod_JServ and Solaris TCP / IP and Kernel parameters.
Chapter 10, "Property Files"	Properties files and their parameters.

Related Documentation

Wireless Edition-specific Documentation

Oracle9*i* Application Server Wireless Edition Configuration Guide (this Guide)

Oracle9*i* Application Server Wireless Edition Implementation Guide

Oracle9*i* Application Server Wireless Edition Developer's Guide

Oracle9*i* Application Server Wireless Edition Installation Guide

Oracle9*i* Application Server Wireless Edition Release Notes

Additionally, Oracle9*i* Application Server, Database, Networking, Application Development, and other documentation is available on your Documentation CD, and at docs.oracle.com.

Documentation Accessibility

Oracle's goal is to make our products, services, and supporting documentation accessible to the disabled community with good usability. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program web site at http://www.oracle.com/accessibility/.

1

Overview

This document describes the post-installation configuration of Wireless Edition.

Each section of this document presents a different topic. These sections include:

- Section 1.1, "Installation Overview"
- Section 1.2, "Manual Configuration"
- Section 1.3, "Upgrade"
- Section 1.4, "Samples"
- Section 1.5, "Web Integration Server"
- Section 1.6, "Wireless Edition Client"
- Section 1.7, "OCI8 JDBC Driver"
- Section 1.8, "User Provisioning"
- Section 1.9, "Geocoding Setup"
- Section 1.10, "JDBC Connection Pooling"

Important: In this document:

- WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

  ```
  c:\ias10210\panama (Windows)
  /private/ias10210/panama (UNIX)
  ```

  Replace "WE_HOME" with the fully-qualified directory path.

- ORACLE_HOME is the directory in which Oracle9*i* Application Server is installed.

1.1 Installation Overview

The Oracle9*i* Application Server installation includes three options:

- Oracle HTTP Server
- Standard Edition
- Enterprise Edition

Wireless Edition is installed as part of each of these options. The *Oracle9i Application Server Installation Guide* discusses each option, and discusses the installation steps specific to Wireless Edition. The Installation Guide defines Wireless Edition and Oracle HTTP Server hardware and software requirements, preinstallation tasks (setting environment variables), port numbers, starting and stopping the Oracle HTTP Server, and the Wireless Edition client configuration.

For a new installation of Wireless Edition, follow the steps in this document starting with the manual configuration in Section 1.2. If you have a previous release of Wireless Edition installed see Section 1.3.

Important: Wireless Edition is not properly configured until the manual configuration steps are completed in Section 1.2.

1.2 Manual Configuration

After installation, perform the manual configuration steps described in this section:

- Section 1.2.1, "Configure Oracle HTTP Server and JServ"
- Section 1.2.2, "Configure ProxyFirewall.properties"
- Section 1.2.3, "Start the Oracle HTTP Server"
- Section 1.2.4, "Wireless Edition Configuration Verification"

1.2.1 Configure Oracle HTTP Server and JServ

This section describes the configuration of the Oracle HTTP Server and JServ.

1. Configure the **httpd.conf** file. The **httpd.conf file** is in the ORACLE_HOME/Apache/Apache/conf directory.

   a. Create an alias for this release of the Personalization Portal. This is needed so that the application server can find the http://hostname/portal/Login.jsp URL. Include the following in the Alias section:

      ```
      Alias /portal/ "WE_HOME/server/portal/"
      ```

   b. To access a previous release of the Personalization Portal, in addition to this release, create an alias. This allows the application server to find the http://hostname/papz/login.jsp URL. Include the following in the Alias section:

      ```
      Alias /papz/ "WE_HOME/server/papz/"
      ```

   c. Create an alias for the Javadoc directory. Include the following in the Alias section. This step is optional.

      ```
      Alias /ptgdoc/ "WE_HOME/doc/javadoc/"
      ```

2. Configure the **jserv.conf** file. The **jserv.conf** file is in the ORACLE_HOME/Apache/Jserv/etc directory. In the ApJServMount section, add the Wireless Edition specific mount point:

   ```
   ApJServMount /ptg /root
   ```

3. Configure the **jserv.properties** file. The **jserv.properties** file is in the ORACLE_HOME/Apache/Jserv/etc directory.

a. Verify that the Java Virtual Machine is version 1.2.2. For example:

```
wrapper.bin= <ORACLE_HOME>/Apache/jdk/bin/java.exe
```

b. After the other "wrapper.classpath" entries, add all the required Wireless Edition files after the "tools.jar" entry in the classpath.

For Windows (for UNIX replace "\" with "/"):

```
wrapper.classpath=<WE_HOME>\server\classes
wrapper.classpath=<WE_HOME>\lib\panama.zip
wrapper.classpath=<WE_HOME>\lib\panama_core.zip
wrapper.classpath=<WE_HOME>\lib\panama_portal.zip
wrapper.classpath=<WE_HOME>\lib\panama_papz.zip
wrapper.classpath=<WE_HOME>\lib\client.zip
wrapper.classpath=<WE_HOME>\lib\server.zip
wrapper.classpath=<WE_HOME>\lib\caboshare-opt.zip
wrapper.classpath=<WE_HOME>\lib\marlin-opt.zip
wrapper.classpath=<WE_HOME>\lib\ocelot-opt.zip
wrapper.classpath=<WE_HOME>\lib\share-opt-1_1_7.zip
wrapper.classpath=<WE_HOME>\lib\tecate-opt.zip
wrapper.classpath=<WE_HOME>\lib\regexp.jar
wrapper.classpath=<WE_HOME>\lib\sax2.jar
wrapper.classpath=<WE_HOME>\lib\activation.jar
wrapper.classpath=<WE_HOME>\lib\mail.jar
wrapper.classpath=<WE_HOME>\lib\ldap.jar
wrapper.classpath=<WE_HOME>\lib\ldapbp.jar
wrapper.classpath=<WE_HOME>\lib\providerutil.jar
wrapper.classpath=<WE_HOME>\lib\sdoapi101.jar
wrapper.classpath=<WE_HOME>\lib\sdovis.jar
wrapper.classpath=<WE_HOME>\lib\MapQuestX.jar
wrapper.classpath=<WE_HOME>\lib\MQJavaCore.jar
wrapper.classpath=<WE_HOME>\lib\RoutingJServerClient.jar
wrapper.classpath=<WE_HOME>\lib\jai_codec.jar
wrapper.classpath=<WE_HOME>\lib\jai_core.jar
wrapper.classpath=<WE_HOME>\lib\jpeg_codec.jar
wrapper.classpath=<WE_HOME>\lib\ordimimg.jar
```

Note: Perform the following step before the Oracle HTTP Server is started. The Personalization Portal requires this step for generating the correct image.

---

4. For UNIX only, to display the images from the Personalization Portal properly, set the DISPLAY environment variable in the **jserv.properties** file to configure access to the host on which the server runs:

   a. If the X server runs on the same machine as the Wireless Edition server, set the DISPLAY environment variable in the **jserv.properties** file as follows:

      ```
      wrapper.env=DISPLAY=:0.0
      ```

      From the X server host machine (*x_server_host_name* in the following example) run the following command:

      ```
      xhost + <x_server_host_name>
      ```

   b. If the X server runs on a different machine (*x_server_host_name* in the following example) than the Wireless Edition server, set the DISPLAY environment variable in the **jserv.properties** file as follows:

      ```
      wrapper.env=DISPLAY=<x_server_host_name>:0.0
      ```

      From the X server host machine, run the following command using the Wireless Edition host (*wireless_edition_host_name* in the following example):

      ```
      xhost + <wireless_edition_host_name>
      ```

      Note: For both cases, remain logged into the Console while the `xhost` commands are executed. Otherwise, properties set through the xhost command may be lost and images may not be displayed properly in the Personalization Portal.

5. Configure the **zone.properties** file. The **zone.properties** file is in the ORACLE_HOME/Apache/Jserv/etc directory.

   a. In the List of Repositories section, add the Wireless Edition specific repository to the existing repository line with a comma (,) separator:

      For Windows:

      ```
      repositories=<ORACLE_HOME>\Apache\Jserv\servlets,<WE_
      HOME>\serverportal,<WE_HOME>\server\papz
      ```

      For UNIX:

      ```
      repositories=<ORACLE_HOME>/Apache/Jserv/servlets,<WE_
      HOME>/server/portal,<WE_HOME>/server/papz
      ```

b. In the Startup Servlets section, add the Wireless Edition specific servlets:

```
servlets.startup=oracle.panama.ParmImpl
```

c. In the Servlet Aliases section, add the Wireless Edition specific servlets:

```
servlet.rm.code=oracle.panama.ParmImpl
```

1.2.2 Configure ProxyFirewall.properties

If a proxy server is used to access an external web site from Wireless Edition, configure the **ProxyFirewall.properties** file:

1. Set http.use.proxy to true. For example:

   ```
   http.use.proxy=true
   ```

2. Specify the correct host in http.proxy.host. For example:

   ```
   http.proxy.host=www-proxy.us.oracle.com
   ```

3. Specify the correct port in http.proxy.port. For example:

   ```
   http.proxy.port=80
   ```

Note: For a list of property files provided by Wireless Edition, see Chapter 10.

1.2.3 Start the Oracle HTTP Server

To start the Oracle HTTP Server see the instructions in the *Oracle9i Application Server Installation Guide*.

1.2.4 Wireless Edition Configuration Verification

After installation, you can verify that individual Wireless Edition components are properly configured using a desktop browser:

1. Test whether the Personalization Portal is working properly by accessing the following URL:

   ```
   http://host_name.domain:7777/portal/Login.jsp
   ```

The login page should appear. The Personalization Portal prompts you to enter a user name and a password. You can log in using "Administrator" as the user name and "manager" as the password.

2. Run the Wireless Edition Request Manager by accessing the following URL:

   `http://host_name.domain:7777/ptg/rm`

   The login page should appear. The page prompts you to enter a user name and a password.

Important: You can log in using "Administrator" as the user name and "manager" as the password. These are the default settings.

1.3 Upgrade

To upgrade a previous release of Wireless Edition (Portal-to-Go) including release 1.0.2.1.0, 1.0.2.2.0, or 1.0.2.3.0 perform the following steps:

1. Stop the Oracle HTTP Server. See the *Oracle9i Application Server Installation Guide* for instructions.
2. Back up the entire "panama" directory. For example, copy the entire "panama" directory to a different location.
3. Back up the repository using an Oracle backup tool.

Important: The Wireless Edition installation in the following step upgrades the database schema and migrates the existing data.

---

4. Run the Oracle9*i* Application Server installation.
5. Follow the manual configuration steps in Section 1.2.
6. Restart the Oracle HTTP Server. See the *Oracle9i Application Server Installation Guide* for instructions.

1.4 Samples

You can install the sample services and adapters into the repository after completing the steps in Section 1.2 and / or Section 1.3.

1. Change to the sample directory:

   ```
   WE_HOME\sample (Windows)
   WE_HOME/sample (UNIX)
   ```

2. Modify the **upload.bat** or **upload.sh** file using the **-l** option to specify the Wireless Edition user and password for which the data is loaded. For example:

   ```
   -l Administrator/manager
   ```

3. Run the following:

   ```
   upload.bat samples.xml (Windows)
   upload.sh samples.xml (UNIX)
   ```

4. Add the following entry in the **jserv.properties** file. This must be added before the Oracle HTTP Server is started.

   ```
   wrapper.classpath=WE_HOME\sample (Windows)
   wrapper.classpath=WE_HOME/sample (UNIX)
   ```

   Note: All the sample Java source and class files are stored in the "sample" directory. Include the sample directory entry in the **jserv.properties** file so that JServ can locate the sample class files.

5. Restart the Oracle HTTP Server. See the *Oracle 9i Application Server Installation Guide* for instructions.

You should see the sample adapters and sample services in the \master\Samples folder using the Service Designer or the Personalization Portal.

Note: Directory Service in the samples directory does not run unless the Oracle Internet Directory is installed. Contact content providers before running any sample service in the YP and Router directories.

1.5 Web Integration Server

The Wireless Edition Web Integration Server hosts services that applications can use to exchange data and information sources via the Web. The Web Integration Server is installed with Wireless Edition. The following steps guide you through the configuration process of the Web Integration Server:

1. Run the Web Integration Server. From the WE_HOME/panama/WebIntegration/Server/bin directory, typing the following:

   a. For Windows:

   ```
   server.bat
   ```

   b. For UNIX

   ```
   server.sh &
   ```

   The Web Integration Server can also be run as a service on Windows.

2. From a browser, go to the Web Integration Server URL:

   ```
   http://host_name.domain:5555
   ```

3. Log in to the Web Integration Server with the user name *Administrator*, and password *manage*. These are the default settings.
4. Select Settings. The server settings appear. Click Edit.
5. Enter the Proxy (HTTP) and Secure Proxy (HTTPS) settings for your environment.
6. Click Submit.
7. Click Logout.

> **Note:** Please make sure the max heap size has been specified for the Web Integration Server. To increase the max heap size in the Web Integration Server startup script, add the **-mx** option.

The Web Integration Developer, the development environment for creating and testing Web Integration services written in Web Interface Definition Language (WIDL), is installed as part of the Wireless Edition client. For more information, see Section 1.6.

1.6 Wireless Edition Client

The Wireless Edition client runs only on the Windows platform. The client installation consists of the following:

- Service Designer
- Web Integration Developer

1.6.1 Service Designer

The Service Designer is a visual interface for implementing and managing services in Wireless Edition. It creates and modifies Wireless Edition objects, including adapters, transformers, and services. The Service Designer provides a tree view of the Wireless Edition repository. This tree displays Wireless Edition objects classes, such as adapters and transformers, as folders or branch nodes. It shows instances of those classes as objects or leaf nodes.

The Service Designer is installed during the Wireless Edition client installation. See the *Oracle 9i Application Server Wireless Edition Implementation Guide* for information on using this tool.

1.6.2 Web Integration Developer

Web Integration Developer is a development environment for creating and testing Web Integration services written in Web Interface Definition Language (WIDL). The Web Integration Developer includes tools for publishing WIDL services for the Web Integration Server.

The Web Integration Developer is installed during the Wireless Edition client installation. For installation instructions, see the *Oracle 9i Application Server Installation Guide*.

Note: The Web Integration Developer includes its own Java Virtual Machine (JVM). It does not require any Java setup.

To configure the Web Integration Developer:

1. Run the Web Integration Developer from the Windows Programs menu. Select Programs, Oracle for Windows, Wireless Edition, and Web Integration Developer.

2. From the Edit menu, select Preferences, and then Configuration.
3. Enter the Proxy (HTTP) and the Secure Proxy (HTTPS) settings appropriate for your environment.
4. Click OK.

1.7 OCI8 JDBC Driver

The default configuration of Wireless Edition uses the Oracle THIN JDBC driver. For a production environment, the Oracle OCI8 JDBC driver is required, as it provides improved performance, concurrency, scalability, and multi-byte support. Additionally, this driver is required if the Oracle Parallel Server option is used.

To configure Wireless Edition to use the Oracle OCI8 driver:

1. Verify that ORACLE_HOME is the directory under which Wireless Edition has been installed.
   a. On Windows: Modify the **jserv.properties** file to include the PATH of the directory where the OCI library (**ocijdbc8.dll**) is installed. The default is ORACLE_HOME\bin.

      ```
      wrapper.path=<ORACLE_HOME>\bin
      ```

   b. On UNIX: Verify that the LD_LIBRARY_PATH environment variable includes the directory where the OCI libraries (**libocijdbc8.so** and **libocijdbc8_g.so**) are installed. The default is ORACLE_HOME/lib.
2. Modify the **System.properties** file to change the driver and the connect string settings.

   ```
   db.connect.string=%WE_REPOSITORY_USERNAME%/%WE_REPOSITORY_USER_
   PW%@(description=(address=(host=%WE_REPOSITORY_
   HOST%)(protocol=tcp)(port=%WE_REPOSITORY_PORT%))(connect_data=(sid=%WE_
   REPOSITORY_SID%)))
   db.driver=V8
   ```

1.8 User Provisioning

You can integrate your repository with an existing provisioning system using one of the following:

1. You can manage users by creating a service with the Wireless Edition provisioning adapter. The provisioning adapter supports the Wireless Edition services that create, search for, update, and delete users at runtime.
2. Use the `LoadXml` utility to import users from a flat XML file that conforms to the Repository DTD. This utility enables you to create and update user information. For more information on the LoadXML utility, see Section 8.2.
3. Use the Wireless Edition data model API to programmatically create users. A model example is provided in the WE_HOME\sample\model directory. Also, see the Javadoc and *Oracle9i Application Server Wireless Edition Developer's Guide* for more information on the data model API.

1.9 Geocoding Setup

Wireless Edition provides basic geocoding through the provider MapInfo. To expand the geocoding capability to include other providers, see the *Oracle9i Application Server Wireless Edition Implementation Guide*. Additionally, if a proxy server is used to access an external web site (including the MapInfo provider) from Wireless Edition, configure the **ProxyFirewall.properties** file:

1. Set http.use.proxy to true. For example:

   ```
   http.use.proxy=true
   ```

2. Specify the correct host in http.proxy.host. For example:

   ```
   http.proxy.host=www-proxy.us.oracle.com
   ```

3. Specify the correct port in http.proxy.port. For example:

   ```
   http.proxy.port=80
   ```

1.10 JDBC Connection Pooling

Pooling for JDBC connections improves resource utilization and avoids connection establishment overhead. A set of configurable parameters for this connection pool is provided in the **System.properties** file. For more information see Section 10.2.

Connection pooling under Wireless Edition session management supports server environments that include short-lived threads, long-lived threads (thread pool model), or a combination of short-lived and long-lived threads. The following system properties apply to the connection pooling mechanism:

```
#Minimum number of connections for the connection pool
db.connect.minConnections=5

#Maximum number of connections for the connection pool
db.connect.maxConnections=100

#Incremental allocation of new connections to the connection pool
db.connect.incConnections=1

#Time interval (in seconds) for the watchdog to find and release unused
#connections to the connection pool
repository.session.check.interval=1
```

Note: To control the maximum idle time for any database connection, configure it through the Oracle data server.

2

FastForward

This document describes the FastForward Email and Directory applications, their installation, and configuration.

Each section of this document presents a different topic. These sections include:

- Introduction
- Installation
- Configuration

2.1 Introduction

The FastForward Email and Directory applications enables you to provide your end users with access to their corporate email and directory lookup on any mobile device. Mobile email will drive productivity within your enterprise, allowing employees to stay in touch while away from the office.

2.1.1 Features

The FastForward Email and Directory applications include implementation of the following features:

- General mobile device support, including: WAP phones, Palms, PocketPCs, and RIM pagers with browser support.
- Flexible deployment options depending on your business needs.
- Support for IMAP and POP3 protocols (including MS Exchange 5.5 and later and Lotus Domino 4.5 and later).
- Integrated corporate email with LDAP employee lookup.
- Standalone LDAP employee lookup.

2.2 Installation

2.2.1 System Requirements

This section is only applicable if the Mail and Directory applications are not run with Oracle 9*i*AS Wireless Edition.

Common

- Sun's JDK 1.2 or JRE 1.2
- Apache 1.3.9
- JSP engine that supports Servlet API 2.0 and JSP 1.0. The product is tested to run on JServ, ServletExec 2.2, and JRun 3.0.
- Optional: HTTPS or VPN for security purposes.
- Oracle 9*i*AS Wireless Edition for non-hosting customers.

Mail

- JavaMail API 1.2 from Sun Microsystems
- JavaBeans Activation Framework extension (JAF) from Sun Microsystems

Directory

- JNDI 1.2 from Sun Microsystems

2.2.2 Installation

2.2.2.1 On Your Own Web Server and JSP Engine

Ensure you have a working web server and a JSP engine configured to run on that web server. Refer to your web server and JSP engine documentation for more information.

Put all the jar files from the `lib` directory into the JSP engine's classpath.

Ensure that **<JDK 1.2>/jre/lib/rt.jar and <JDK 1.2>/lib/tools.jar** is in the classpath.

Configure the JSP engine to point to the JSP files in the `JSP` directory.

Mail.jsp is the main JSP for the Mail Application, and **LDAP.jsp** is the main JSP for the Directory Application.

2.2.2.2 On PTG 1.1

PTG 1.1 already includes Apache web server and JServ 1.1.1.

Configure JServ so that the jar files are in its classpath. Go to **<JServ1.1.1 directory>/conf/jserv.properties**, then put the following lines before all other classpaths:

- **wrapper.classpath=<ompim properties directory>**
- **wrapper.classpath=<ompim lib directory>/ompim.jar**

Also ensure that the JavaMail and LDAP jar files are in **wrapper.classpath**.

Configure Apache so that the JSP files are on the document root on Apache. **Mail.jsp** is the main JSP for the Mail Application, and **LDAP.jsp** is the main JSP for the Directory Application.

Create a new service in PTG ServiceDesigner to point to the URL of the applications. See the Configuration of URL Adapter in PTG 1.1 ServiceDesigner for more information.

2.3 Configuration

The OracleMobile Mail and Directory applications provide mobile access for industry standard electronic mail systems and Directory servers. Supported backends include:

Mail

- IMAP
- POP3
- MS Exchange 5.5 and 2000
- Lotus 4.6 and Release 5

Directory

- LDAP servers supporting JNDI 1.2

Mail and Directory applications can be two separate applications, or they can be configured separately in such a way that they can interact with each other.

Also, the Mail and Directory applications can be used as both hosted applications or as shipped products.

2.3.0.3 Configuration

You can configure the Mail and Directory Application in one of two ways:

- Send configuration parameters through `request`, or
- Use the Mail application property file.

The Mail and Directory applications will first try to use the parameters sent through `request`. Then they will try to look for other missing parameters. Finally, default values will be used.

2.3.0.3.1 Configuration Through Properties Files

For the Mail application, configure **OracleMobileEmail.properties** (in the `oracle/panama/module/communicate/mail` directory).

For the Directory Application, configure **OracleMobileLDAP.properties** (in the `oracle/panama/module/communicate/ldap` directory).

An example of the files is in the `doc` directory.

IMPORTANT: The root "oracle" directory containing the properties files must be in the classpath of the JSP engine. There is a properties directory in the install; modify the properties files there, and put the `<install>/properties` directories in the classpath of the JSP engine.

Note: Parameters and values in properties file is case sensitive.

WARNING: Ensure you are using the proper format for your UNIX or Windows platforms. For example:

UNIX format `mail.logger=/tmp/mail.log`

Windows format `mail.logger=d:\\tmp\\mail.log`

For the Mail application, you must configure the following parameters:

Table 2–1 Mail parameters

mail.service	Mail service supported by the mail server. Currently supported services are "IMAP" or "POP3". Default is IMAP.
mail.server.name	Full hostname of the mail server.
mail.server.port	Port number of the Mail server on the host. Default for IMAP is 143. Default for POP3 is 110.
mail.smtp.server.name	Full hostname of the SMTP server.
mail.smtp.server.port	Port number of the SMTP server on the host. Default for IMAP is 25.
mail.smtp.server.login	Returns the login name of the SMTP server. If login for SMTP is not required, this method should return a null or empty string (""), otherwise, the value will be used for SMTP login.
mail.folder.inbox	Name of the Inbox folder on the mail server. Default is "INBOX". This property is optional.

mail.folder.sent	The name of the Sent folder on the mail server. Default is "Sent". This property is optional.
mail.default.emaildomain	Default email domain for sending emails. When sending emails, if the email domain is not specified, the default email domain will be appended to the username.
mail.log	Determines whether log in is enabled or not. Value is either "true" or "false". Default is "true".
mail.logger	Full path of the log file.
mail.msgfetch.setsize	Maximum number of messages to fetch for one request. This is the maximum number of messages returned for one request. For example, if a folder has 300 messages, only the first 200 will be returned at the first request.
mail.server.connect.timeout	Connection timeout in milliseconds. Default is 2000.
mail.config.class	The configuration class. Users can provide their own configuration classes, allowing them to do more advanced configuration, for example, select a mail server host name based on username. Default is **oracle.panama.module.communicate.mail.util.Config** which is shipped with the OracleMobile Mail application.
mail.directoryEnabled	Enables / disables directory access from email. If enabled, when users are composing emails, they can go to directories to look for email addresses. Values can be "true" or "false". Default is "false".
ldap.designer.path	Used when directory is enabled. Check the "mail.directoryEnabled" parameter. When the application is running on PTG, but not studio, then the path to the application specified in ServiceDesigner Service Trees is needed. For example, if LDAP is the service defined for the Directory Application, and it is put in a folder called `Apps` under `Service Trees`, then the path should be `/Apps/Ldap`.

mail.designer.path	Path to the application specified in ServiceDesigner Service Trees. Check the "studio" parameter. For example, if Mail is the service defined for the Mail application, and it is put in a folder called `Apps` under `Service Trees`, then the path should be `/Apps/Mail`.

For the Directory application, you must configure the following parameters:

Table 2–2 Directory parameters

ldap.host	LDAP server hostname.
ldap.port	LDAP server port number. The default is 389.
ldap.log	Determines whether logging is turn on or off. Values can be "true" or "false". Default is "true".
ldap.logger	Full pathname of the log file.
ldap.queries	Two default queries are shipped with the OracleMobile Directory Application: One searches for a person on the common name LDAP attribute. The second searches for a person on the first name and the last name LDAP attribute. For details about LDAP query syntax, see the LDAP documentation.
ldap.links	Hotlink is a mechanism that enables you to create hyperlinks on the attribute in the result list. For details about Hotlink, see the LDAP documentation.
ldap.labelMultipleMatch	Sets the values that appear in the results list from the query, when a query returns more than one line. Note that the values must be in the query. The default is givenname, surname, and title.
ldap.labelMultipleMatch	Contains givenname, surname, and title
ldap.maxResultCount	Maximum results returned from a query. Default is 1000.
ldap.enableLogin	Determines if LDAP login is required for a user. The value is either "true" or "false". Default is "false".

ldap.username **ldap.password**	If an LDAP login is required, a default username and password can be specified. Ensure the **ldap.enableLogin** property is true.
ldap.emailEnabled	Enables / disables email access from a directory. If enabled, when a person from LDAP is displayed, users can send an email directly from the email address.
ldap.emailAttribute	Name of the email attribute of a person in the LDAP schema.
ldap.phoneAttribute	Name of the telephone number attribute of a person in the LDAP schema.
ldap.maxRecPage	Maximum results displayed on a screen. Default is 9.
mail.designer.path	Path to the application specified in ServiceDesigner Service Trees. Check the "studio" parameter. For example, if Mail is the service defined for the Mail application, and it is put in a folder called `Apps` under `Service Trees`, then the path should be `/Apps/Mail`.
ldap.designer.path	Used when directory is enabled. Check the "mail.directoryEnabled" parameter. When the application is running on PTG, but not studio, then the path to the application specified in ServiceDesigner Service Trees is needed. For example, if LDAP is the service defined for the Directory Application, and it is put in a folder called `Apps` under `Service Trees`, then the path should be `/Apps/Ldap`.

2.3.0.3.2 Configuring Mail and Directory Parameters through `request`

Parameters defined this way will override the values defined in the properties file.

The following Mail parameters can be passed through `request`:

Table 2–3 Mail parameters passed through `request`

CONFIG_FILE	Specifies the properties file that defines all the other Mail properties. This allows multiple properties files for different configurations. By default, the file **OracleMobileEmail.properties** will be used. For example, if you want to use the properties file **myconfig.properties**, put it into the request CONFIG_FILE=myconfig, and ensure the file is on the classpath.
PTG_VERSION	Definition, possible values, and default values are the same as the "oracle.panama.version" in the properties file.
STUDIO	Definition, possible values, and default values are the same as the "studio" in the properties file.
MAIL_SERVICE	Definition, possible values, and default values are the same as the "mail.service" in the properties file.
MAILHOST	Definition, possible values, and default values are the same as the "mail.server.name" in the properties file.
MAILPORT	Definition, possible values, and default values are the same as the "mail.server.port" in the properties file.
SMTPHOST	Definition, possible values, and default values are the same as the "mail.smtp.server.name" in the properties file.
SMTPPORT	Definition, possible values, and default values are the same as the "mail.smtp.server.port" in the properties file.
SMTP_LOGIN	Definition, possible values, and default values are the same as the "mail.smtp.server.login" in the properties file.
SENDER_DOMAIN	Definition, possible values, and default values are the same as the "mail.smtp.domain" in the properties file.
INBOX	Definition, possible values, and default values are the same as the "mail.folder.inbox" in the properties file.
SENT	Definition, possible values, and default values are the same as the "mail.folder.sent" in the properties file.

DOMAIN	Definition, possible values, and default values are the same as the "mail.default.emaildomain" in the properties file.
LOG	Definition, possible values, and default values are the same as the "mail.log" in the properties file.
LOGFILE	Definition, possible values, and default values are the same as the "mail.logger" in the properties file.
MAIL_FETCH_SIZE	Definition, possible values, and default values are the same as the "mail.msgfetch.setsize" in the properties file.
TIMEOUT	Definition, possible values, and default values are the same as the "mail.server.connect.timeout" in the properties file.
CONFIG	Definition, possible values, and default values are the same as the "mail.config.class" in the properties file.
DIR_ENABLED	Definition, possible values, and default values are the same as the "mail.directoryEnabled" in the properties file.
USERNAME	Username of the user for transparent login.
PASSWORD	Password of the user for transparent login. If USERNAME is not sent at the same time, this password will be ignored, and the user must login. For example, if you want to bypass the login, you can pass the username and password through the request: http://myURL/Mail.jsp?USERNAME=myusername&PASSWORD=mypassword

All the other parameters will be read from the properties file, or the default values will be used. If other parameters are defined property, you can bypass the login. Note that all parameters values are strings. Also, if a parameter is defined through `request`, it will override the corresponding parameter in the properties file.

The following Directory parameters can be passed through `request`:

Table 2–4 Directory parameters passed through `request`

LDAPHOST	Definition, possible values, and default values are the same as the "ldap.host" in the properties file.
LDAPPORT	Definition, possible values, and default values are the same as the "ldap.port" in the properties file.

PTG_VERSION	Definition, possible values, and default values are the same as the "oracle.panama.version" in the properties file.
STUDIO	Definition, possible values, and default values are the same as the "studio" in the properties file

All the other parameters will be read from the properties file, or the default values will be used. If other parameters are defined property, you can bypass the login. Note that all parameters values are string. Also, if a parameter is defined through `request`, it will override the corresponding parameter in the properties file.

2.3.0.3.3 Mail Configuration Class

For Mail configuration, you can define your own configuration class to do more advanced configuration. For example, select mail server host name based on runtime information.

To do this, you must define a class which implements **oracle.panama.module.communicate.mail.util.IConfig**, and in package **oracle.panama.module.communicate.mail.util**. In the class, you can define the following methods:

Table 2–5 Mail configuration class

String getMailService(Object o)	Returns the mail service. The current supported services are IMAP and POP3.
String getDomain(Object o)	Returns the default mail domain.
String getServerName(Object o)	Returns the mail server host name.
int getServerPort(Object o)	Returns the mail server port number. Make sure it is a valid number.
String getSMTPServerName(Object o)	Returns the SMTP server host name.
int getSMTPServerPort(Object o)	Returns the SMTP server port number. Make sure it is a valid number.
String getSMPTLogin(Object o)	Returns the login name of the SMTP server. If login for SMTP is not required, this method should return null or empty string (""), otherwise, the value will be used for SMTP login.

String getXMLVersion (Object o)	Returns the PTG XML version. Ensure it is consistent with the "studio" parameters. Currently, studio is running PTG 1.0.
boolean isStudio(Object o)	Indicates whether it is running in studio or not. Ensure it is consistent with the "oracle.panama.version" parameters. Currently, studio is running PTG 1.0.
String getInboxFolderName (Object o)	Returns the Inbox folder name.
String getSentFolderName (Object o)	Returns the Sent folder name.
boolean isLogging(Object o)	Indicates if logging is enabled.
String getLogfile (Object o)	Returns the full path of the log file.
String getTimeout (Object o)	Returns the connection timeout in milliseconds.

In the **OracleMobileEmail.properties** file, change the **mail.config.class** property to be the class you define.

The input is an object. The default behavior is not to pass anything. You can change it to pass information such as a string (*username* for example), or a hashtable containing more information.

If you do not want to implement all of the methods, you can extend from **oracle.panama.module.communicate.mail.util.Config**. You can choose to override methods, but the rest will be the default implementations. The default is to ignore the username and pick up the information from the properties file.

Refer to the Javadoc **IConfig.html** and the Java interface in the doc directory of the install.

2.3.0.3.4 Configuration of URL Adapter in PTG 1.1 ServiceDesigner

Mail and Directory applications can be run through the URL Adapter in PTG 1.1. To do so, you must create a Master Service and a Service Alias. Here are the steps (for details, refer to the PTG documentation):

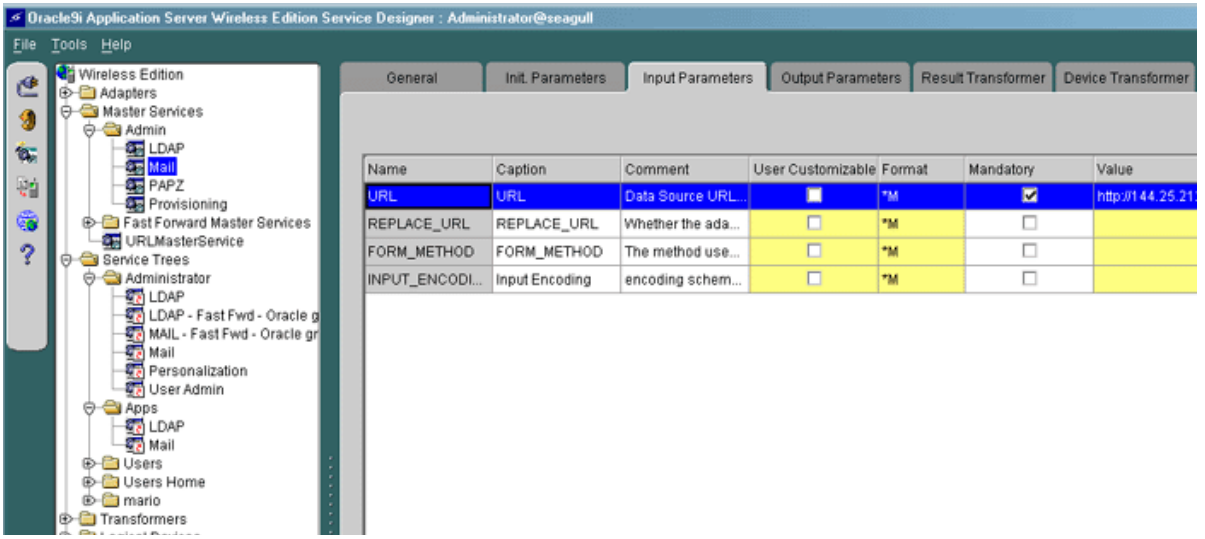
1. Open Service Designer and connect to a PTG 1.1 instance.
2. Open the `Master Services` folder at the top level.

You can create a new Master Service in an existing folder or create a new folder and then create the Master Service in that folder. In the example, the Master Service "Mail" is created under the folder "Admin".

> **IMPORTANT:** Oracle Corporation recommends that the service not to be created at the root level directly under `Master Service`.

3. Choose URL Adapter for the new service.
4. In the "Input Parameters" tab, under "Value", put the URL of the application.
5. Create a service alias (for example: **http://pc1.company.com/mail.jsp**) for the new service under the "Service Trees" folder. The alias should point to the Master Service created. In the example below, the service alias "Mail" is created under the "Apps" folder under "Service Trees". Oracle Corporation also recommends that the alias not be created at the level directly under `Service Trees`. The alias should point to `/master/Admin/Mail`.
6. Create an alias for users that points to the service alias created in the previous step. In the example below, the service alias "Mail" is created under the "Administrator" folder located under "Service Trees". This means the service is available for the user "Administrator".

Figure 2–1 Service Designer Window



Now you can use e-mail.

3

Alerts and SMS Request Listener

Alerts and SMS Request Listener discusses alerts and the SMS Request Listener in Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 3.1, "Overview"
- Section 3.2, "Alerts"
- Section 3.3, "SMS Request Listener"
- Section 3.4, "Request and Notification Queue Pool"

Important: In this document WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

```
c:\ias10210\panama (Windows)
/private/ias10210/panama (UNIX)
```

Replace "WE_HOME" with the fully-qualified directory path.

3.1 Overview

Alerts, in the Wireless Edition, are time-based scheduled services. Alerts are executed when the scheduled time expires. The result, if any, after the service is executed, is delivered to the end user at the default alert address. The Wireless Edition can accept either email or phone numbers as the alert address. If the default alert address is an email address, the result is delivered as an email message. If the default alert address is a phone number, the result is delivered as a SMS. The alert

and alert address can be created by end users through Wireless Edition personalization from the desktop computer or the device.

Services in Wireless Edition can be invoked through the HTTP protocol and through SMS. Users can invoke the services through SMS with the corresponding service name. However, currently only services without input arguments can be invoked through SMS. The result of the service execution is returned as another SMS. The capability is provided through the SMS Request Listener. [[Is this still true?]]

3.2 Alerts

Alerts, which are time-based scheduled services, are placed in the request queue when they are created. The result of the alert execution, if any, is placed in the notification queue. The alert functionality in the Wireless Edition requires the installation of the Oracle8*i* JOB and AQ options.

Important: The aq_tm_processes parameter in the **init.ora** file must be set to at least "2" for the processing of asynchronous alerts in Wireless Edition.

3.2.1 Request Queue

The request queue stores all the alerts to be executed. The Wireless Edition processes an alert as follows:

1. An alert is created or updated with a start date and a specified interval when the user creates a time-based scheduled service. The alert is placed in the request queue.
2. When the specified time expires, the alert is activated and executed. The result, if any, is placed in the notification queue with the alert creator's default alert address. The alert with an updated next execution time is placed back in the request queue.
3. If the alert fails to be executed for any reason, the alert is not placed back in the request queue. It can only be reactivated by updating the start date through the Personalization Portal.

3.2.2 Notification Queue

The result of an alert execution, if any, together with the alert creator's default alert address, is placed in the notification queue. The Wireless Edition dequeues notification messages out of the notification queue. If the alert address is an email address, the result is sent out as an email message. If the alert address is a phone number, the result is sent out as SMS to the specified phone number.

3.2.3 Email-based Alert

If the result of a delivered alert is an email message, set the following entries in the **Notification.properties** file in the WE_HOME\server\classes\oracle\panama\core\admin directory:

```
# The smtp server hostname, mandatory <hostname>
mail.server.name=
# The mail domain name, set if SMTP server requires this <domainname>
mail.domain=
# The from mail address, mandatory <email address>
mail.from=
# The organization, optional <string>
mail.organization=
# The default mail subject, mandatory <string>
mail.subject=
# The Content-Transfer-Encoding, mandatory <string>
mail.content.transfer.encoding=
```

3.2.4 SMS-based Alert

If the result of a delivered alert is a SMS message, configure the **Notification.properties** file in the WE_HOME\server\classes\oracle\panama\core\admin directory.

3.2.4.1 SMS-C Server

Configure the following entries to specify the host on which the SMS-C is running:

```
# the hostname on which the SMS-C is running
sms.server.name=
# the port on the sms.server.name from which the SMS-C is listening the incoming
SMS-message
sms.server.port=
```

Generally, the Wireless Edition uses direct TCP/IP communication to the SMS-C server when sending the SMS message. If the UCP protocol is selected, the Wireless

Edition can also use the URL connection to the SMS-C server when sending the SMS message. In this case, provide the correct URL to the SMS-C server:

```
sms.server.url=
```

3.2.4.2 SMPP Protocol

If the SMPP protocol is used to communicate between the Wireless Edition

and SMS-C when sending the notification message, set:

```
sms.driver.class=oracle.panama.util.sms.SmsSMPPDriver
```

Set the following entries with the appropriate values:

```
# The SMPP system id <string>
#
sms.smpp.system.id=
#
# The SMPP system type <string>
#
sms.smpp.system.type=
#
# The SMPP client password <string>
#
sms.smpp.system.password=
```

3.2.4.3 SMS Protocol

When sending the SMS message, the Wireless Edition can use either the Universal Computer Protocol (either operation 01 or operation 51 command set), or the Short Message Peer-to-Peer (SMPP) protocol to communicate with the SMS-C.

3.2.4.4 UCP Protocol

If the UCP protocol is used to communicate between the Wireless Edition and SMS-C when sending the notification message, set:

```
sms.driver.class=oracle.panama.util.sms.SmsUCPDriver
```

Also, specify which command set to use:

```
sms.ucptype=01 (for operation 01 command set)
sms.ucptype=51 (for operation 51 command set)
```

3.2.4.5 Flow-Control

Configure the following entries for the SMS-control:

```
# The max number of SMS chunks per message. <int>
#
sms.message.maxchunks=
#
# The maximum number of bytes per message. <int>
#
sms.message.maxsize=
```

3.2.4.6 Wireless Edition SMS System Account for SMS-C

If there is a dedicated SMS system account created at SMS-C for the Wireless Edition to send the SMS to, configure the following entries appropriately:

```
# Wireless Edition SMS System Account Id
sms.account.id
# The corresponding password
sms.account.password
```

3.3 SMS Request Listener

Note: The following configuration is not required if the SMS Request Listener will not be provided to the end user.

The SMS Request Listener allows the invocation of any Wireless Edition service using a SMS message. The SMS Request Listener processes the SMS-based service invocation request and returns the result, if any, as a SMS message by performing the following:

1. Authenticate the user by finding the alert address with the actual phone number.
2. Invoke the service.
3. Place the result, if any, after invoking the service, in the notification queue.

To configure the SMS Request Listener:

1. Include oracle.panama.util.sms.SmsServer as another daemon thread in the locator.request.daemon.classes entry in the **System.properties** file in the WE_HOME/server/classes/oracle/panama/core/admin directory.
2. Configure the SMS protocol. See Section 3.2.4.3.
3. If the UCP protocol is used when configuring the SMS protocol, set:

   ```
   sms.receiver.listener.mode=true
   ```

4. If the SMPP protocol is used when configuring the SMS protocol, set:

   ```
   sms.receiver.listener.mode=false
   ```

5. If the UCP protocol is used, set the incoming listener port appropriately:

   ```
   sms.receiver.listener.port=
   ```

3.4 Request and Notification Queue Pool

To improve the throughput for request and notification processing, configure the number of threads that process the corresponding queue.

3.4.1 Request Queue Thread Pool Size

To configure the thread pool size for the request queue, set the following appropriately in the **AsynchRequest.properties** file in the WE_HOME/server/classes/oracle/panama/core/admin directory:

```
# Thread pool size at startup
#
init.pool.size=
#
# Minimum Thread pool size
#
minimum.pool.size=
```

3.4.2 Notification Queue Thread Pool Size

To configure the thread pool size for the notification queue, set the following appropriately in the **AsynchNotification.properties** file in the WE_HOME/server/classes/oracle/panama/core/admin directory:

```
# Thread pool size at startup
```

```
#
init.pool.size=
#
# Minimum Thread pool size
#
minimum.pool.size=
```

4

Logging

The document describes the logging capability of Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 4.1, "Overview"
- Section 4.2, "Database Logging"
- Section 4.3, "Service Designer Logging"
- Section 4.4, "Server Logging"
- Section 4.5, "Transaction Logging"
- Section 4.6, "Configuring the Runtime Monitor and Management"

4.1 Overview

Wireless Edition generates logging information for both the server and development client components.

4.2 Database Logging

Wireless Edition database logging records each authenticated service invocation request and each successful user session. Database logging is required for performance monitoring and measurement using Oracle Enterprise Manager. The logged information includes service access patterns, response times for services, and the number of sessions. This data can be used by remote management tools to display results such as service access patterns, throughput, and average response times. Database logging is handled asynchronously.

4.2.1 Configuring the Database Logger

The database logger of a Wireless Edition instance must point to a database schema to log the information. This configuration is done in the **System.properties** file of each Wireless Edition instance.

```
#
#System Logger related parameters
#

# Number of minutes after which the logger thread wakes up to flush
# the log entries to the database. This parameter is optional.
SystemLog.logger.wakeupFrequency=10

# The maximum number of entries cached by the logger thread.
# This parameter is optional.
SystemLog.logger.maxSize=100

# Whether service logging should be enabled
SystemLog.enableServiceLogging=true

# Whether session logging should be enabled
SystemLog.enableSessionLogging=true

# The request listener class
locator.request.listener.classes=oracle.panama.core.admin.SystemLogger

# The session listener class
locator.session.listener.classes=oracle.panama.core.admin.SystemLogger

# Enable event request logging
event.before.request=true
event.request.end=true
event.request.error=true
event.after.request=true

# Enable event session logging
event.session.begin=true
event.session.end=true

# System Log Database connect string <String>
# Syntax for this property is similar to db.connect.string
# System Logs can be present in a separate database
#
# NOTE: For the OEM cartrige queries, Make sure the user specified below has
```

```
select access to the V$ tables.
#   Usage example: log in as system and say "grant select any table to %PANAMA_
USERNAME%"
#SystemLog.db.connect.string=%PANAMA_USERNAME%/%PANAMA_USER_PW%@%PANAMA_
HOST%:%PANAMA_PORT%:%PANAMA_SID%
```

If the SystemLog.db.connect.string parameter is set, logged information is stored in a separate database. This parameter should not be set if the logged information is to be stored in the same repository.

Note: The database user name used here (the one replacing %PANAMA_USERNAME%) must have select access to all tables. In this example "PANAMA" is Wireless Edition.

4.2.2 Database Table Contents

Data logged to the database is stored in two tables, ptg_service_log and ptg_session_log.

Table 4–1 Contents of the ptg_session_log Table

Column Name	Description
session_id	The identifier of the session.
user_id	The Object Identifier (OID) of the user of the session.
user_name	The user name.
ptg_instance_id	A unique identifier, for the database instance.
logical_device	The logical device which invoked the service.
login_time	The timestamp when the session was created.
login_hour	The hour when the session was created.
expiry_time	The timestamp when the session expired.
timestamp	The time when the data is logged.

Table 4–2 Contents of the ptg_service_log Table

Column Name	Description
service_id	Object Identifier (OID) for the invoked service.
service_name	Name of the invoked service.
ptg_instance_id	Unique identifier, for the database instance.
final_service_id	OID of the final service, for example the master service or folder.
final_service_name	Name of the final service.
session_id	Session identifier for which the service was invoked.
service_arguments	Arguments of the service, in XML format.
bookmark	External link, if the invoked service is a bookmark.
service_type	Type of service.
invocation_hour	Hour when the service was invoked.
invocation_time	Date when the service was invoked.
response_time	Response time for the service.
request_status	Status of the request, a non-zero value indicates an error number.
error_description	Error message, if one occurred while invoking the service.
user_id	OID for the user.
user_name	Name of the user.
remote_address	Host IP address and name.
logical_device	Logical device which invoked the service.
external_user_id	External user ID.
external_user_name	External user name.
adapter_type	Type of adapter.
adapter_home	Time taken by the adapter.
transformation_time	Time taken by the transformer.
timestamp	Time when the data is logged.

4.3 Service Designer Logging

By default, the Service Designer writes error information to the log directory of the Service Designer home directory. You can modify this setting, and the log naming pattern, in the **preferences.xml** file. The file is in the WE_HOME / tools / ServiceDesigner / resources directory.

4.4 Server Logging

The Wireless Edition writes server error information to the files and directory specified in the **System.properties** file. You can open the log files directly, or view them from the Wireless Edition Runtime Information and Management interface (the Probe interface) at port 8090. Click the "Files" link.

The Wireless Edition specifies these types of errors:

Table 4–3 Server Error Information

Error	Value
ERR_SERVICE_UNAVAILABLE	PTG-001
ERR_SERVICE_ERROR	PTG-002
ERR_SERVICE_NOT_FOUND	PTG-003
ERR_CONFIGURATION_ERROR	PTG-004
ERR_INTERNAL_ERROR	PTG-005
ERR_DATABASE_ERROR	PTG-006

In addition to error messages, the Wireless Edition provides extensive runtime exception logging. When fatal exceptions occur, the Wireless Edition logs the exceptions and stack traces in the global log file. If you need to contact Oracle Support Services, you should have the log information available.

4.5 Transaction Logging

The transaction log is a log file on the Wireless Edition server that provides information regarding user access. To enable transaction logging, you must include transaction logging as a log level in the **System.properties** file. For example:

```
log.level=Warning, Error, Notify, Transaction
```

You can use the **System.properties** file to set the directory to which the Wireless Edition writes transaction logs, and to specify a file name pattern for the log files. You can also configure the transaction record pattern.

If you enable transaction logging when a user accesses a service that has a cost value set, the transaction log generates data. You can set a cost for a master service using the Wireless Edition Service Designer. You can access the transaction log data programmatically, to generate billing information or to integrate the Wireless Edition transaction information with an external billing management system.

The following sample shows the log results of four stock quote queries by a single user. The sample uses the default logging pattern, which you can modify in the **System.properties** file. It lists the time of access, the user name (Sample), the service, and the price per access (25):

```
[11/14/99 4:03:26 PM] Sample   /master/OnlineQuoteOracle 25
[11/14/99 4:03:29 PM] Sample   /master/OnlineQuoteOracle 25
[11/14/99 4:03:35 PM] Sample   /master/OnlineQuoteOracle 25
```

4.6 Configuring the Runtime Monitor and Management

The probe is a daemon that displays runtime information about:

- Log files
- Active objects
- Active sessions
- Active threads

The probe performs simple runtime management including:

- Shutting down an active object instance.
- Refreshing persistent attributes on an active object.
- Setting the global debug flag to on or off.
- Reloading properties for device recognition when adding a new device to a running system.

By default, installation of Wireless Edition installs and initiates the probe runtime monitor with the listening port 8090. To access the probe from a browser enter the following URL:

http://*host_name*:8090

You can configure the probe component by editing the **www-server.properties** file in the WE_HOME/server/classes/oracle/panama/core/probe directory. The following daemon manages the implementation:

`oracle.panama.core.probe.WebServer`

5

Multiple JVM Support and Load Balancing

Multiple JVM Support and Load Balancing describes multiple Java Virtual Machine (JVM) support in Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 5.1, "Overview"
- Section 5.2, "Multiple Instances on Same Machine as Oracle HTTP Server"
- Section 5.3, "Multiple Instances on the Same Machine / Oracle HTTP Server on Different Machine"
- Section 5.4, "Multiple Instances and Oracle HTTP Server All on Different Machines"
- Section 5.5, "Load Balancing"
- Section 5.6, "Configuring the Personalization Portal for Load-Balancing"

Note: Oracle recommends that each Java Virtual Machine (JVM) not support more than 200 concurrent user sessions.

Important: In this document:

- WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

  ```
  c:\ias10210\panama (Windows)
  /private/ias10210/panama (UNIX)
  ```

  Replace "WE_HOME" with the fully-qualified directory path.

- ORACLE_HOME is the directory in which Oracle9*i* Application Server is installed.

5.1 Overview

The following sections describe three possible deployment scenarios for multiple Java Virtual Machine (JVM) support in Wireless Edition:

- Section 5.2, "Multiple Instances on Same Machine as Oracle HTTP Server"
- Section 5.3, "Multiple Instances on the Same Machine / Oracle HTTP Server on Different Machine"
- Section 5.4, "Multiple Instances and Oracle HTTP Server All on Different Machines"

This document describes:

- The Cache synchronization setup (to ensure that the object caches of each instance are synchronous).
- Multiple JVM environment without JServ session cookies.
- Personalization Portal setup.

Note: The steps in this document assume an initial configuration of single JVM support. Verify that Wireless Edition is initially set up to run as a single instance communicating with a single Oracle HTTP Server instance.

5.2 Multiple Instances on Same Machine as Oracle HTTP Server

To configure multiple JVM instances on the same machine:

1. Modify the **jserv.conf** file so the Oracle HTTP Server is aware of the multiple JServ instances that are running. Start each of these JServ instances manually. Insert the following statements after any existing statements in the section that begins with <IFModule> and ends with </IFModule> in the **jserv.conf** file. The **jserv.conf** file is in the ORACLE_HOME\Apache\JServ\conf directory.

   a. Modify the following:

   ```
   <IfModule mod_jserv.c>
   ApJServManual on
   ```

   b. Running multiple instances of JServ requires a manual start for each instance:

   ```
   ApJServMount /ptg balance://set1/root
   ```

   c. W1, W2,...WN are integers that determine the load-balancing weight assigned to each of the instances. To balance equally among all instances; do not specify W*x*; the weight value must be an integer. Weight can be decided on the relative load you want each JServ to handle. Requests for /ptg are load-balanced on set "set1":

   ```
   ApJServBalance set1 PC1 W1
   ApJServBalance set1 PC2 W2
   .
   .
   ApJServBalance set1 PCN WN
   ```

   d. Port numbers can be any valid port on the system, however they should not conflict with any other application using the same port.

   ```
   ApJServHost PC1 ajpv12://localhost:7771
   ApJServHost PC2 ajpv12://localhost:7772
   .
   .
   ApJServHost PCN ajpv11://localhost:777N
   ```

   e. JS1 to JSN are routing cookies. These can be set to any unique string for each entry.

   ```
   ApJServRoute JS1 PC1
   ```

```
ApJServRoute JS2 PC2
.
.
ApJServRoute JSN PCN
```

f. This file is used internally by Apache for implementing fault tolerance:

```
ApJServShmFile ORACLE_HOME\Apache\Jserv\jserv_shm
```

g. Close the section using:

```
</IfModule>
```

2. Start the Oracle HTTP Server.

```
ORACLE_HOME\Apache\Apache\Apache.exe -k start -d
ORACLE_HOME\Apache\Apache
```

3. Create and modify the **jserv.properties** files for each of the instances. Make copies of the default **jserv.properties** files for each of the instances, for example, **jserv1.properties**, **jserv2.properties**. The **jserv.properties** file is in the ORACLE_HOME\Apache\JServ\conf directory. Modify the files:

a. Port which the instance uses to communicate with Apache using the AJP protocol. *x* corresponds to each of the JServ instances 1...N.

```
port=777x
```

b. The log file. For Unix, ensure that the JServ process owner has write permissions for this file:

```
log.file=ORACLE_HOME\Apache\Jserv\logs\jservx.log
```

c. This allows only clients from the local machine to access the JServ instances:

```
security.allowedAddresses=127.0.0.1
```

d. Change the location of class files specific to the JServ instance from:

```
wrapper.classpath=<WE_HOME>\server\classes
```

to:

```
wrapper.classpath=<WE_HOME>\server\classesx
```

4. To enable cache synchronization among the database instances, modify the Wireless Edition specific properties files. The default location for these files is in the <WE_HOME>\server\classes directory. Create copies of this directory

structure and modify the structure so that each of the JServ instances have unique port numbers and other resources. For example, these directories should appear as <WE_HOME>\server\classesx, where *x* corresponds to each of the JServ instances 1...N. Modify the **System.properties** file in the <WE_HOME>\server\classesx\oracle\panama\core\admin directory:

a. Set the unique identifier for the instance. *x* corresponds to each of the JServ instances 1...N.

```
instance.identifier=instancex
```

b. Set the following parameter to true for every instance to ensure that the instance participates in cache synchronization:

```
participateInCacheSynchronization=true
```

c. This value should point to the machine and port where the Master Instance for Cache synchronization is running. It should have the same value for all instances. Ensure that the port number (2007 in this case) is the same as the one specified by the rmi.registry.port parameter in the **Master.properties** file:

```
masterInstance.url=//localhost:2007
```

d. Modify the properties files in Table 5–1. These files are in the <WE_HOME>\server\classesx\oracle\panama\core directory. The sub-directories are specified.

e. Run the following SQL scripts by connecting to the repository database as the Wireless Edition database user. The scripts are provided in the <WE_HOME>\server\classesx\oracle\panama\sql directory.

```
create_cachesync_schema.sql
enable-cachesynch_triggers.sql
```

f. The master server must be started by running the script <WE_HOME>\server\classesx\oracle\panama\sample\runMasterServer.bat or runMasterServer.sh.

Table 5–1 Property File Settings

Property File	Setting
\admin\AsynchNotification.properties	server.listen.port=5500x
\core\admin\AsynchRequest.properties	server.listen.port=6500x
\core\admin\Ftp.properties	ftp.server.port=910x
\core\admin\Notification.properties	sms.server.port=5000x
\core\admin\Rmi.properties	rmi.server.port=201x
	The port number on which the RMI server runs.
\probe\www-server.properties	port=809x
\magent\config\MAgent.properties	magent.server.port=201x
	Same as rmi.server.port in **Rmi.properties**.
\spatial\spatial.properties	Change path to class files to reflect the per instance classes.
\Master\Master.properties	Modify the db.connect parameter in this file to be the same as that in **System.properties**.
	Modify only the **Master.properties** file in the classpath of the master instance.
	ptgInstances.length=N
	ptgInstance1.rmi.host=localhost
	ptgInstance1.rmi.port=2011 (This is the value of rmi.server.port in the **Rmi.properties** file.)
	ptgInstance1.rmi.objectName=CacheSyncClient (Do not change CacheSyncClient to any other value.)
	ptgInstance2.rmi.host=localhost
	ptgInstance2.rmi.port=2012
	ptgInstance2.rmi.objectName=CacheSyncClient
	ptgInstanceN.rmi.host=localhost
	ptgInstanceN.rmi.port=201N
	ptgInstanceN.rmi.objectName=CacheSyncClient

5. Start each of the JServ instances manually after the Oracle HTTP Server is started. Create a batch script to supply all the classes specified in the

jserv.properties file to Java as part of the classpath. See Section 5.6.3 for a sample batch script.

5.3 Multiple Instances on the Same Machine/Oracle HTTP Server on Different Machine

The steps are the same as in Section 5.2 except for the following:

1. While configuring **jserv.conf** change the ApJServHost directive so that the Oracle HTTP Server points to the machine running the JServ instances, by specifying its IP address. For example:

   ```
   ApJServHost PC1 ajpv12://jserv-machine:7771
   ApJServHost PC2 ajpv11://jserv-machine:7772
   .
   .
   ApJServHost PCN ajpv11://jserv-machine:777N
   ```

   These port numbers can be any valid port on the machine running JServ instances, only that they must not conflict with any other application using the same port.

2. In each of the **jserv.propertiesx** files, change the bindaddress parameter to the IP address of the machine running the JServ instance, and the security.allowedAddresses parameter to point to the IP address of the machine running the Oracle HTTP Server. For example:

   ```
   bindaddress=ip_address_of_machine_running_jserv_instance
   security.allowedAddresses=192.168.1.2
   ```

5.4 Multiple Instances and Oracle HTTP Server All on Different Machines

The steps are similar to Section 5.2, in addition:

1. Modify the **jserv.conf** file on the machine running the Oracle HTTP Server. Ensure that the ApJServHost directive includes the IP addresses of each of the machines running the JServ instances. For example:

   ```
   ApJServHost PC1 ajpv12://jserv-machine1:8007
   ApJServHost PC2 ajpv11://jserve-machine2:8007
   .
   .
   ApJServHost PCN ajpv11://jserv-machineN:8007
   ```

2. Start the Oracle HTTP Server.
3. On each machine running the JServ instances, modify the **jserv.properties** file:

   ```
   bindaddress=ip_address_of_machine_running_jserv_instance
   security.allowedAddresses=oraclehttpserver_ip_address
   ```

4. On the machine running the Master Instance, modify the **Master.properties** file. See Table 5–1 for more information.
5. Modify the **system.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin directory on all machines:
   a. Set the unique identifier for the instance:

      ```
      instance.identifier=instancex
      ```

   b. Set the following parameter to true to ensure that the instance participates in cache synchronization:

      ```
      participateInCacheSynchronization=true
      ```

   c. Set the following parameter to point to the machine and port where the Master Instance for Cache synchronization is running. It should have the same value for all instances. Ensure that the port number (2007 in this case) is the same as the one specified by the rmi.registry.port parameter in the **Master.properties** file.

      ```
      masterInstance.url=//machine_running_master_instance:2007
      ```

   d. Run the SQL script **enable_cachesync_schema.sql** by connecting to the repository database as the Wireless Edition database user. The script resides, by default, in the <WE_HOME>\sql directory.
   e. Start the master server on the machine running the master instance by running the script **runMasterServer.bat** in the <WE_HOME>\sample directory.
6. Start each of the JServ instances manually using the batch script. See Section 5.6.3 for a sample script.

5.5 Load Balancing

When multiple instances are running, load balancing can be performed by a separate machine. For example, there could be several Oracle HTTP Server instances, each with a set of JServ instances balanced on a one-to-many basis by this machine. The Oracle HTTP Server instances can perform load balancing by using a hardware load balancer in front of the Oracle HTTP Server instances.

The main purpose of multiple JVM support in Wireless Edition is to provide scalability through load-balancing. There are three implementations when setting up load-balancing.

- Load-balancing among Oracle HTTP servers through hardware load-balancing.
- Load-balancing among Jserv instances with cookies.
- Load-balancing among Jserv instances with URL rewriting.

5.5.1 Load-Balancing with Cookies

When intermediate gateways or proxies and the clients support cookies, load balancing can be achieved using cookies:

1. Set the enable.http.session.binding parameter in all of the **System.properties** files (on all machines, if more than one machine is involved) to true.
2. In the ORACLE_HOME\Apache\JServ\conf\zone.properties file set the session.useCookies parameter to true.

5.5.2 Load Balancing Through URL Rewriting

When intermediate gateways or proxies do not support cookies, load balancing can be achieved through URL rewriting:

1. Set the enable.http.session.binding parameter in all of the **System.properties** files (on all machines, if more than one machine is involved) to true.
2. In the ORACLE_HOME\Apache\JServ\conf\zone.properties file set the session.useCookies parameter to false.

5.6 Configuring the Personalization Portal for Load-Balancing

This section describes the configuration of the Personalization Portal in two environments:

- Section 5.6.1, "Clients / Gateways Support Cookies"
- Section 5.6.2, "Clients / Gateways Do Not Support Cookies"

5.6.1 Clients/Gateways Support Cookies

To set up the Personalization Portal when clients and / or intermediate gateways support cookies, enable one of the load-balanced Wireless Edition instances to service Personalization Portal requests. Modify the **jserv.conf** file to change the value of the ApJServDefaultPort parameter to the port number of any of the load-balanced instances (ApJServDefaultPort 7771).

5.6.2 Clients/Gateways Do Not Support Cookies

To set up the Personalization Portal when clients and / or intermediate gateways do not support cookies, run a separate Wireless Edition instance to service Personalization Portal requests.

1. Modify the **jserv.conf** file to change the value of the ApJServDefaultPort parameter to the port number of any unoccupied port (ApJServDefaultPort 7774).
2. Create a copy of the **jserv.properties** file, for example, **jserv4.properties** (assuming that there are three load-balanced instances running) for each of the instances and modify the following:
   a. Set the port which the instance uses to communicate with Oracle HTTP Server using the AJP protocol. This should be the same as the value set for the ApJServDefaultPort parameter in the **jserv.conf** file.

      ```
      port=7774
      ```

   b. Set the log file. For UNIX ensure that the JServ process owner has write permissions for this file.

      ```
      log.file=ORACLE_HOME\Apache\Jserv\logs\jserv4.log
      ```

   c. Ensure that only the Oracle HTTP Server talks to this JServ instance:

      ```
      security.allowedAddresses=oraclehttpserver_ip_address
      ```

   d. Change wrapper.classpath=<WE_HOME>\server\classes to wrapper.classpath=<WE_HOME>\server\classes*x*. This is the location of class files specific to the JServ instance.

3. Copy the default **zone.properties** file in ORACLE_HOME\Apache\Jserv\conf to **zone_pp.properties**. In this file, ensure that the session.useCookies parameter has a value of true.

   ```
   root.properties= ORACLE_HOME\Apache\Jserv\conf\zone_pp.properties
   ```

4. Modify the various Wireless Edition specific properties files. The default location for these files is in the <WE_HOME>\server\classes directory. Create a copy of this directory and modify the properties files so that each of the JServ instances has a unique port number and other resources. For example, the directory should appear as <WE_HOME>\server\classes. In the <WE_HOME>\server\classes4\oracle\panama\core\admin\ directory, modify the **System.properties** file.

   a. Set the unique identifier for this instance:

      ```
      instance.identifier=instance
      ```

   b. Set the following parameter to true to ensure that the instance participates in cache synchronization:

      ```
      participateInCacheSynchronization=true
      ```

   c. Set the enable.http.session.binding to false.

   d. Set the following to point to the machine and port where the Master Instance for Cache synchronization is running. It should have the same value for all instances. Ensure that the port number (2007 in this case) is the same as the one specified by the rmi.registry.port parameter in the **Master.properties** file on the machine running the Master instance.

      ```
      masterInstance.url=//localhost:2007
      ```

   e. Set the following in the **AsynchNotification.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin\ directory to:

      ```
      server.listen.port=5500
      ```

   f. Set the following in the **AsynchRequest.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin\ directory to:

      ```
      server.listen.port=6500
      ```

   g. Set the following in the **Ftp.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin\ directory to:

      ```
      ftp.server.port=910
      ```

h. Set the following in the **Notification.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin\ directory to:

```
sms.server.port=5000
```

i. Set the following in the **Rmi.properties** file in the <WE_HOME>\server\classes\oracle\panama\core\admin\ directory. This is the port number on which the RMI server runs:

```
rmi.server.port=201
```

j. Set the following in the **www-server.properties** file in the <WE_HOME>\server\classes\oracle\panama\core directory:

```
port=809
```

k. Set the following in the **MAgent.properties** file in the <WE_HOME>\server\classes\oracle\panama\magent\config directory. This is the same as the rmi.server.port setting in **Rmi.properties** in the <WE_HOME>\server\classes\oracle\panama\core\admin directory.

```
magent.server.port=201
```

l. Set the path to the class files to reflect the per instance classes in the **Spatial.properties** file in the <WE_HOME>\server\classes\oracle\panama\spatial\ directory.

Now the instance can be started using the batch file in Section 5.6.3.

5.6.3 Sample Batch File

The following is a sample batch file for manually starting and stopping Oracle HTTP Server. Classes for hooks or other features should be added at the end of the file.

```
@ECHO OFF
set classpath=%classpath%;ORACLE_HOME\java\lib;PORACLE_HOME\Apache\jdk\lib
set classpath=%classpath%;ORACLE_HOME\Apache\Jserv\ApacheJserv.jar
set classpath=%classpath%;ORACLE_HOME\Apache\Jsdk\lib\jsdk.jar
set classpath=%classpath%;<WE_HOME>\server\classes%1
set classpath=%classpath%;<WE_HOME>\lib\panama.zip
set classpath=%classpath%;<WE_HOME>\lib\panama_core.zip
set classpath=%classpath%;<WE_HOME>\lib\panama_portal.zip
set classpath=%classpath%;<WE_HOME>\lib\panama_papz.zip
set classpath=%classpath%;ORACLE_HOME\jdbc\lib\classes12.zip
```

```
set classpath=%classpath%;ORACLE_HOME\jsp\lib\ojsp.jar
set classpath=%classpath%;ORACLE_HOME\lib\servlet.jar
set classpath=%classpath%;ORACLE_HOME\Apache\BC4J\lib\jndi.jar
set classpath=%classpath%;ORACLE_HOME\lib\xmlparserv2.jar
set classpath=%classpath%;<WE_HOME>\lib\client.zip
set classpath=%classpath%;<WE_HOME>\lib\server.zip
set classpath=%classpath%;<WE_HOME>\lib\caboshare-opt-1_0_0.zip
set classpath=%classpath%;<WE_HOME>\lib\marlin-opt-1_0_0.zip
set classpath=%classpath%;<WE_HOME>\lib\ocelot-opt-1_0_0.zip
set classpath=%classpath%;<WE_HOME>\lib\share-opt-1_1_7.zip
set classpath=%classpath%;<<WE_HOME>>\lib\tecate-opt-1_0_0.zip
set classpath=%classpath%;<WE_HOME>\lib\regexp.jar
set classpath=%classpath%;<WE_HOME>\lib\sax2.jar
set classpath=%classpath%;<WE_HOME>\lib\activation.jar
set classpath=%classpath%;<WE_HOME>\lib\mail.jar
set classpath=%classpath%;<WE_HOME>\lib\ldap.jar
set classpath=%classpath%;<WE_HOME>\lib\ldapbp.jar
set classpath=%classpath%;<WE_HOME>\lib\providerutil.jar
set classpath=%classpath%;<WE_HOME>\lib\sdoapi.jar
set classpath=%classpath%;<WE_HOME>\lib\sdovis.jar
set classpath=%classpath%;<WE_HOME>\lib\MapQuestX.jar
set classpath=%classpath%;<WE_HOME>\lib\MQJavaCore.jar
set classpath=%classpath%;<WE_HOME>\lib\RoutingJServerClient.jar
set classpath=%classpath%;<WE_HOME>\lib\jai_codec.jar
set classpath=%classpath%;WE_HOME\lib\classes%1
REM  Add other classpath entries here for specific features or hooks implemented
REM  by the customer.
java -c %org.apache.jserv.JServ ORACLE_HOME\Apache\JServ\conf\jserv%1.properties
```

This script (in the example, **jservmanual.sh**) can be used in the following manner to start and stop a Jserv instance:

- To start instance 1, use:

  ```
  jservmanual.sh 1
  ```

- To stop an instance, use:

  ```
  jservmanual.sh 1 -s
  ```

6

Oracle Enterprise Manager

Oracle Enterprise Manager describes the use of Oracle Enterprise Manager with Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 6.1, "Overview"
- Section 6.2, "Prerequisites"
- Section 6.3, "Configuring the Server, Client, and Middle-tier"
- Section 6.4, "Setting up the ptg_services.tcl File"
- Section 6.5, "Configuration for Another Wireless Edition Instance"
- Section 6.6, "Configuring the Database Logger"
- Section 6.7, "Asynchronous Notification of Error Information"
- Section 6.8, "Launching the Oracle Enterprise Manager Tools"

Important: In this document:

- WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

  ```
  c:\ias10210\panama (Windows)
  /private/ias10210/panama (UNIX)
  ```

Replace "WE_HOME" with the fully-qualified directory path.

- ORACLE_HOME is the Oracle 8.1.7 Oracle home.

6.1 Overview

The use of Oracle Enterprise Manager with Wireless Edition provides:

- Performance monitoring
- Fault management
- Configuration management

Performance Monitoring

The Oracle Enterprise Manager Diagnostic Pack includes Performance Manager and Capacity Planner. The Performance Manager is a Windows NT client side tool for viewing performance data collected by the Wireless Edition Cartridge. The Capacity Planner allows you to archive and view performance data collected by the Wireless Edition cartridge. This cartridge is used by the Data Gatherer to collect performance data logged by the Wireless Edition runtime.

Fault Management

The event framework of Oracle Enterprise Manager enables asynchronous notification of error information. When an error occurs, Wireless Edition generates an Oracle Enterprise Manager event at the node where the error occurred. The event provides the details necessary for fault diagnosis. The Oracle Intelligent Agent running on each node picks up these events and forwards them to the Oracle Enterprise Manager Console for display. You can configure the Console to page or email the administrator.

Configuration Management

The configuration management framework provides a way to alter some runtime parameters.

6.2 Prerequisites

To configure Oracle Enterprise Manager for use with Wireless Edition you must first install the following components:

- Oracle Enterprise Manager 2.2 with Diagnostic Pack
- Oracle Intelligent Agent 8.1.7
- Oracle Management Server 2.2

6.3 Configuring the Server, Client, and Middle-tier

The Oracle Enterprise Manager configuration includes server side, client side, and middle tier configuration instructions:

- The Oracle Enterprise Manager server side configuration involves the Oracle Intelligent Agent and Data Gatherer. See Section 6.3.1.
- The Oracle Enterprise Manager client side configuration involves the Console (the navigator and events) and Diagnostic Pack (Performance Manager and Capacity Planner). See Section 6.3.2.
- The Oracle Enterprise Manager middle-tier configuration involves the Oracle Management Server, specifically the configuration to receive threshold events from the Data Gatherer Cartridge on the Console. See Section 6.3.3.

6.3.1 Configuring the Server Side

The server side configuration includes the configuration of the Oracle Intelligent Agent and the Wireless Edition Data Gatherer Cartridge.

6.3.1.1 Configuring the Oracle Intelligent Agent

To configure the Oracle Intelligent Agent on the server:

1. Install the Oracle Intelligent Agent 8.1.7 in ORACLE_HOME. If the Oracle Management Server is to reside on the same host as the Oracle Intelligent Agent, ensure that the Oracle Management Server is also installed in the same ORACLE_HOME.

   Note: If running Oracle 8.1.6, perform a custom installation of Oracle 8.1.7 Enterprise Edition. Select the Oracle Intelligent Agent and its subcomponents. Install it in a separate ORACLE_HOME.

2. Copy the file **dgmetric.tcl** from the WE_HOME/oem/agent directory to the Oracle Intelligent Agent ORACLE_HOME/network/agent/events/oracle/generic/metric directory. This operation replaces the existing **dgmetric.tcl** file.
3. Copy the file **ptg_services.tcl** from the WE_HOME/oem/agent directory to the ORACLE_HOME/network/agent/config directory.

4. Append the **ptg_services.tcl** file name to the **nmiconf.1st** file found in the ORACLE_HOME/network/agent/config directory.

   The agent discovers the Wireless Edition service using these files.

5. Edit the **ptg_services.tcl** file to configure the instance(s) information. See Section 6.4 for instructions on setting up the **ptg_services.tcl** file. Configuration for a default instance name (PTG1) is provided as an example.

Note: The Oracle Intelligent Agent must be restarted whenever any change is made to the **ptg_services.tcl** file or to the **nmiconf.lst** file.

After completing the preceding instructions, install the Wireless Edition Data Gatherer cartridge. The Oracle Intelligent Agent uses this cartridge to collect Wireless Edition performance data. The Performance Manager and Capacity Planner use the collected data to provide runtime metrics for diagnosis.

The cartridge installation steps differ according to the hardware platform.

6.3.1.2 Wireless Edition Data Gatherer Cartridge Installation: Unix

To install the data gatherer cartridge:

Important: The cartridge installation requires an Oracle Enterprise Manager 2.2 Console and Oracle Intelligent Agent 8.1.7 on the server side.

---

1. Un-tar the **ptgsolariscartridge.tar** file found in WE_HOME/oem/agent/cartridge/solaris to a directory (referred to as *CART_DIR* in this document).
2. Copy CART_DIR/lib/libvpxptg.so to the ORACLE_HOME/odg/lib directory.
3. Copy CART_DIR/mesg/vpxptgus.msb to the ORACLE_HOME/odg/mesg directory.
4. Copy CART_DIR/html/vpxptgUS.htm to the ORACLE_HOME/odg/html directory.

5. Append the following entry to the **svppcart.dat** file in the ORACLE_HOME/odg directory.

```
PTG ALL libvpxptg vpxptg
```

This completes the server side configuration. Restart the Oracle Intelligent Agent. Restart the Data Gatherer.

Oracle Intelligent Agent Start and Stop

To stop the agent:

```
lsnrctl dbsnmp_stop
```

To start the agent:

```
lsnrctl dbsnmp_start
```

Data Gatherer Start and Stop

To stop the gatherer:

```
vppcntl -stop
```

To start the gatherer:

```
vppcntl -start
```

Note: For more information on commands to restart the Oracle Intelligent Agent, Oracle Management Server, and Data Gatherer, see the Oracle Enterprise Manager Configuration Guide.

6.3.1.3 Wireless Edition Data Gatherer Cartridge Installation: Windows

To install the data gatherer cartridge:

Important: The cartridge installation requires an Oracle Enterprise Manager 2.2 Console and Oracle Intelligent Agent 8.1.7 on the server side.

---

1. Un-zip the **ptgwincartridge.zip** file in WE_HOME/oem/agent/cartridge/win32 to a directory (referred to as *CART_DIR* in this document).
2. Copy the file **vpxptg.dll** in CART_DIR\odg\bin to the ORACLE_HOME\bin directory.
3. Copy the file **vpxptgUS.htm** in CART_DIR\odg\html to the ORACLE_HOME\odg\html directory.
4. Copy the file **vpxptgus.msb** in CART_DIR\odg\mesg to the ORACLE_HOME\odg\mesg directory.
5. Append the following entry to the **svppcart.dat** file in the ORACLE_HOME\odg directory:

   ```
   PTG ALL vpxptg vpxptg
   ```

This completes the server side configuration for Oracle Enterprise Manager integration. Restart the Oracle Intelligent Agent and the Data Gatherer. This starts the Wireless Edition cartridge along with the Oracle Intelligent Agent startup.

Select the Control Panel and then Services to restart the appropriate service.

6.3.2 Configuring the Client Side

To configure the Oracle Enterprise Manager client side:

Note: Requirement for deployment of the integration classes: JRE 1.1.8_10 for Unix and JRE 1.1.7.30o for Windows.

---

1. Install the Oracle Enterprise Manager client, including the Console and client tools, Oracle Management Server, and Diagnostic Pack (Performance Manager and Capacity Planner).

Note: The Oracle Enterprise Manager Diagnostic Pack is only available for Windows.

---

2. Copy the file **oemptg.jar** from the WE_HOME/oem/client directory to the ORACLE_HOME/classes directory.

3. Extract this file in the copied directory. This extracts the console integration classes, Performance Manager integration classes, events integration classes, and RMI client and stub for the Console interaction with the server. For example:

```
jar -xvf oemptg.jar
```

4. The entries in WE_HOME/oem/client/OEMClient.properties.append must be appended to ORACLE_HOME/sysman/config/OEMCLient.properties. The following shows the entries.

```
#
# System definition of types for Wireless Edition
#
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/isa =test_service1
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/service = true
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/display_class =
oracle.sysman.ptg.navigator.PtgDisplayDriver

#
# Console definition of types for navigator tree for
# Wireless Edition
#
/com/oracle/sysman/em/console/navigator/oracle_sysman_ptg/data_source =
oracle.sysman.ptg.navigator.PtgSubItemContainerSource

/com/oracle/sysman/em/console/navigator/oracle_sysman_ptg/proxy
=oracle.sysman.ptg.navigator.PtgProxy

#
# Events Integration
#
# Wireless Edition events integration
#
/com/oracle/sysman/em/console/tests_datasource/ptg/datasource =
oracle.sysman.ptg.PTGTestsDatasource

# PTG Console credentials classes for Events Integration
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/isa =
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/service = true
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/credentials_display =
oracle.sysman.vtcConsole.vtcCredentials.PtgCredentialsDisplay
/com/oracle/sysman/em/system/types/oracle_sysman_ptg/credentials =
oracle.sysman.emSDK.common.credentials.PtgCredentials
```

5. The entries in WE_HOME/oem/client/vtdclient.properties.append must be appended to ORACLE_HOME/sysman/config/vtdclient.properties. The following shows the entries:

```
/target/oracle_sysman_diagpack/oracle_sysman_
ptg=oracle.sysman.vtd.VtdPtgCredentials
/options/standalone=TRUE
```

6. Edit the ORACLE_HOME/classes/oracle/panama/magent/MAgent.properties file to reflect the values of the installed Wireless Edition server name. The port number and the service name are defaulted.

Note: This property file should be in synch with the one on the Wireless Edition server side for each instance to which the file refers.

---

7. For the Oracle Enterprise Manager client side installation the only pertinent values are **magent.server.host** and **magent.server**. The remaining properties may be ignored or defaulted here.

```
# Server Host Name
magent.server.host=<Wireless Edition server host name>


# The service port: Use the same port as the RMI server as
#                   in Rmi.properties
magent.server.port=2008

# Service Name
magent.server.name=Magent
```

8. Verify that the classpath contains the following entries.

For UNIX:

```
$ORACLE_HOME/classes
$ORACLE_HOME/jdbc/lib/classes111.zip
$ORACLE_HOME/lib/vbjorb.jar
$ORACLE_HOME/lib/vbjapp.jar
$ORACLE_HOME/lib/vbjtools.jar
$ORACLE_HOME/classes/classesFromIDLVisi
$ORACLE_HOME/jlib/ewt-3_3_6.jar
```

```
$ORACLE_HOME/jlib/ewtcompat-opt-3_3_6.zip
$ORACLE_HOME/jlib/ewt-swingaccess-1_1_1.jar
$ORACLE_HOME/jlib/share-1_0_8.jar
$ORACLE_HOME/jlib/jssl-1_2.jar
$ORACLE_HOME/jlib/javax-ssl-1_2.jar
$ORACLE_HOME/jlib/netcfg.jar:/
```

For Windows:

```
%ORACLE_HOME%\classes
%ORACLE_HOME%\jdbc\lib\classes111.zip
%ORACLE_HOME%\lib\vbjorb.jar
%ORACLE_HOME%\lib\vbjapp.jar
%ORACLE_HOME%\lib\vbjtools.jar
%ORACLE_HOME%\classes\classesFromIDLVisi
%ORACLE_HOME%\jlib\ewt-3_3_6.jar
%ORACLE_HOME%\jlib\ewtcompat-opt-3_3_6.zip
%ORACLE_HOME%\jlib\ewt-swingaccess-1_1_1.jar
%ORACLE_HOME%\jlib\share-1_0_8.jar
%ORACLE_HOME%\jlib\jssl-1_2.jar
%ORACLE_HOME%\jlib\javax-ssl-1_2.jar
```

The Oracle Enterprise Manager client side configuration is now complete.

6.3.3 Configuring the Middle-tier

The middle-tier hosts the Oracle Management Server. The following configuration is for the Events Integration.

1. Copy the file **oemptg.jar** in the WE_HOME/oem/oms directory to the ORACLE_HOME/classes directory where the Oracle Management Server is installed.
2. Extract this file in the copied directory, for example:

   ```
   jar -xvf oemptg.jar
   ```

   This extracts the Console integration classes, Performance Manager integration classes, events integration classes, and RMI client and stub for the Console interaction with the server.
3. The entries in WE_HOME/oem/oms/OMSRegistry.registry.append must be appended to the **OMSRegistry.registry** file in ORACLE_HOME/sysman/admin.

   ```
   # ORACLE_HOME/sysman/admin/OMSRegistry.registry
   ```

```
# Wireless Edition events integration
#
/com/oracle/sysman/em/tests_datasource/ptg/datasource =
oracle.sysman.ptg.PTGTestsDatasource
```

Note: After updating the **OMSRegistry.registry** file, recreate the Oracle Management Server repository using the Oracle Enterprise Manager configuration assistant. See the Oracle Enterprise Manager Configuration Guide for more information.

This completes the middle-tier configuration. Restart the Oracle Management Server and Oracle Intelligent Agent.

6.4 Setting up the ptg_services.tcl File

To set up the **ptg_services.tcl** file:

1. Set the correct values of the NEW_PARAMETER_NAME, PTG_INSTANCE_NAME, TRACE_DB_HOSTNAME, TRACE_DB_PORT, TRACE_DB_SID, and TRACE_DB_ORACLE_HOME for each instance.
2. To add another instance:
   a. Add a new parameter to store the new ServiceName (the Wireless Edition instance name) to the "global Parameters". See the following example.
   b. Copy the lines between "BEGIN EXAMPLE" and "END EXAMPLE", and replace the information with the new instance's parameter name and remaining information.

```
global Parameters ServiceType HostName tnsaddress argv NEW_PARAMETER_
NAME ServiceName2

set Parameters(oracle_sysman_ptg) {ServiceType HostName tnsaddress };

set Host [lindex $argv 0];

#The Service name is a subnode in the navigator

set NEW_PARAMETER_NAME PTG_INSTANCE_NAME;

lappend ServiceNames $NEW_PARAMETER_NAME;
set ServiceType($NEW_PARAMETER_NAME) oracle_sysman_ptg;
```

```
set HostName($NEW_PARAMETER_NAME) $Host;
set tnsaddress($NEW_PARAMETER_NAME) "(DESCRIPTION=(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = TRACE_DB_HOSTNAME)(PORT = TRACE_DB_
PORT)))(CONNECT_DATA=(SID=TRACE_DB_SID)(ORACLE_HOME=TRACE_DB_ORACLE_
HOME)(SERVER=DEDICATED)))"
```

In the following example a new parameter (ServiceName2) is added and the values for the PTG_INSTANCE_NAME, TRACE_DB... are set. For this example the listed parameters are replaced with the values in the following. The values are similar to the **tnsnames.ora** configuration.

Table 6–1 ptg_services.tcl Parameters

Parameter	Value in Example	Description
NEW_PARAMETER_NAME	ServiceName2	Parameter name declaration.
PTG_INSTANCE_NAME	PTG2	The Wireless Edition instance name.
TRACE_DB_HOSTNAME	ptgserver-sun.us.oracle.com	The host name of the database log database.
TRACE_DB_PORT	1521	The port of the database listener.
TRACE_DB_SID	ptgdev	The SID of the log database.
TRACE_DB_ORACLE_ HOME	/private/oracle/8.1.7	The Oracle home directory.

```
# Example:
# The Service name "PTG2" (Wireless Edition Instance name) is a subnode in the
# Navigator tree

# In the line below, just append the new parameter name
#  i.e. ServiceName2. Don't duplicate this line.
global Parameters ServiceType HostName tnsaddress argv ServiceName2

set Parameters(oracle_sysman_ptg) {ServiceType HostName tnsaddress };

set Host [lindex $argv 0];

# Lines below need to be duplicated every time a new instance is added
set ServiceName2 PTG2;
lappend ServiceNames $ServiceName2;
set ServiceType($ServiceName2) oracle_sysman_ptg;
```

```
set HostName($ServiceName2) $Host;
set tnsaddress($ServiceName2) "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL
= TCP)(HOST = ptgserver-sun.us.oracle.com)(PORT = 1521)))(CONNECT_
DATA=(SID=ptgdev)(ORACLE_HOME=/private/oracle/8.1.7)(SERVER=DEDICATED)))"
```

Note: Place this file in ORACLE_HOME/network/agent/config. Add this file name to **nmiconf.lst**. To reflect any change to either file restart the Oracle Intelligent Agent and rediscover the node.

6.5 Configuration for Another Wireless Edition Instance

The preceding configuration is for one Wireless Edition instance. To configure another Wireless Edition instance:

1. **Ptg_services.tcl**: Add another entry in the ORACLE_HOME/network/agent/config/ptg_services.tcl file. Give this instance a different service name. See Section 6.4 for instructions and an example.
2. **Minstances.properties**: Add a corresponding instance's <instance name > = <property file name> entry to the ORACLE_HOME/classes/oracle/panama/magent/config/MInstances.properties file. The RMI client uses this information to communicate with the corresponding server. Each instance is monitored using a **Magent.properties** file which contains the Wireless Edition server host name, port, and service name. In the case of multiple instances, these are identified by these values. The **Minstances.properties** file should point the instance name to the corresponding **Magent.properties** file. For example, for two instances:

   ```
   PTG1=oracle.panama.magent.config.Magent1
   PTG2=oracle.panama.magent.config.Magent2
   ```

3. **Magent.properties**: Configure the **Magent.properties** file on both the client and server side to point to the new instance's classes and port. Users can have any number of instances but each instance's information is stored in a separate property file.
4. Ensure that an Oracle Enterprise Manager Agent is installed on the node where the Wireless Edition instance is running. The Wireless Edition instance uses the Agent to pipe error messages to the Oracle Enterprise Manager Console.
5. The **System.properties.templ** file contains parameters for the system logger and the logger schema. Ensure that these are set up appropriately.

6.6 Configuring the Database Logger

The database logger of a Wireless Edition instance must point to a database schema to log the information. This configuration is done in the **System.properties** file of each Wireless Edition instance. See Chapter 4 for more information on the database logger.

6.7 Asynchronous Notification of Error Information

Asynchronous notification of error information is provided using the event framework provided by Oracle Enterprise Manager. In case of an error, the Wireless Edition server generates an Oracle Enterprise Manager event at the node where the error occurred providing all the necessary details for the fault diagnosis.

Note: An Oracle Enterprise Manager agent must be running on the node where **oemevent** is invoked. The event must be defined as an unsolicited event and the node must be added as a monitored destination.

The Oracle Enterprise Manager Event properties can be configured by editing the **oemevent.properties** file in the WE_HOME/server/oracle/panama/core/admin directory.

The **oemevent.properties** file has the following parameters:

Table 6–2 Properties of the oemevents.properties File

Key	Type	Customizable	Description
oemevent.path	string	Y	Complete path name for the oemevent executable.
oemevent.event.name	string	Y	Unsolicited event name as defined in the console.
oemevent.object.name	string	Y	Object name specified in the event definition.
oemevent.event level	string	Y	Event level specified in the event definition. The level should be defined as "2" and specified as "alert".

6.8 Launching the Oracle Enterprise Manager Tools

Performance Monitoring is accomplished through the Oracle Enterprise Manager Performance Manager and Capacity Planner tools (provided only for Windows NT). Once the installation is complete:

1. Launch the necessary performance monitoring tool. See the Oracle Enterprise Manager documentation for instructions on how to launch it.
2. At the login screen select Stand-alone and no repository connection. Click OK.
3. Double-click *PTG PM Targets*.
4. Enter the Wireless Edition login information to the schema where the Wireless Edition dumps the runtime log information. This information is the same as the system logger configuration information that is entered in Section 6.6.
5. Enter the Data Gatherer Location. This is the host on which you installed the Data Gatherer and Wireless Edition cartridge.
6. Click OK.

This allows you to begin using the performance monitoring and capacity planning.

The Fault and Configuration Management is accomplished through the Oracle Enterprise Manager Console.

1. Launch the Oracle Enterprise Manager Console. See the Oracle Enterprise Manager documentation for instructions on how to launch it.
2. Log on to the repository which you created during your Oracle Enterprise Manager installation.
3. Discover the node where the Oracle Intelligent Agent is running.
4. Configure the Console to receive unsolicited events from the Nodes (hostnames) on which Wireless Edition is running. See the Oracle Enterprise Manager documentation for instructions on how to configure it.

7

Multi-byte Character Support

Multi-byte Character Support describes multi-byte character support in Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 7.1, "Overview"
- Section 7.2, "Multi-byte Encoding Schemes"
- Section 7.3, "Setting the Multi-Byte Encoding for the Personalization Portal"
- Section 7.4, "Setting up a Netscape Browser to Display Multi-byte Data"
- Section 7.5, "LocalStrings.properties Files and Localization"

Important: In this document:

- WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

  ```
  c:\ias10210\panama (Windows)
  /private/ias10210/panama (UNIX)
  ```

  Replace "WE_HOME" with the fully-qualified directory path.

- ORACLE_HOME is the directory in which Oracle9*i* Application Server is installed.

7.1 Overview

This release of Wireless Edition supports single-byte, multi-byte, and fixed-width encoding schemes which are based on national, international, and vendor-specific standards.

If the character set is single byte, and that character set includes only composite characters, the number of characters and the number of bytes are the same. If the character set is multi-byte, there is generally no such correspondence between the number of characters and the number of bytes. A character can consist of one or more bytes, depending on the specific multi-byte encoding scheme.

A typical situation is when character elements are combined to form a single character. For example, in the Thai language, up to three separate character elements can be combined to form one character, and one Thai character would require up to 3 bytes when TH8TISASCII or another single-byte Thai character set is used. One Thai character would require up to 9 bytes when the UTF8 character set is used.

7.2 Multi-byte Encoding Schemes

Multi-byte encoding schemes are needed to support ideographic scripts used in Asian languages like Chinese or Japanese since these languages use thousands of characters. These schemes use either a fixed number of bytes to represent a character or a variable number of bytes per character.

7.2.1 Fixed-width Encoding Schemes

In a fixed-width Multi-byte encoding scheme, each character is represented by a fixed number of *n* bytes, where *n* is greater than or equal to two.

7.2.2 Variable-width Encoding Schemes

A variable-width encoding scheme uses one or more bytes to represent a single character. Some Multi-byte encoding schemes use certain bits to indicate the number of bytes that represent a character. For example, if two bytes is the maximum number of bytes used to represent a character, the most significant bit can be toggled to indicate whether that byte is part of a single-byte character or the first byte of a double-byte character. In other schemes, control codes differentiate single-byte from double-byte characters. Another possibility is that a shift-out code is used to indicate that the subsequent bytes are double-byte characters until a shift-in code is encountered.

7.3 Setting the Multi-Byte Encoding for the Personalization Portal

The Personalization Portal receives the encoding for the text of the site from the setting in the PAPZ logical device, which is in the repository. The default encoding is VTF-8, which can be used for both Western European and Asian languages. The portal sets the content for each page with the encoding specified by the logical device. To change the default encoding to multi-byte encoding click PAPZ under Logical Devices in the Service Designer and change the encoding for your particular language.

7.4 Setting up a Netscape Browser to Display Multi-byte Data

To set up a Netscape 4.6 web browser to display Multi-byte data:

1. Click Edit, Preference, Appearance, and Fonts.
2. Select Unicode in the For the Encoding field.
3. For example, for Chinese, select MS Song in the Variable Width Font field.
4. For example, for Chinese, select MS Song in the Fixed Width Font field.
5. Select the "Use my default fonts, overriding document-specified fonts" radio button.

7.5 LocalStrings.properties Files and Localization

Localization has been simplified through the use of a property file called **LocalStrings.properties**. This file contains text labels used by screens within various adapters and JSP pages.

7.5.1 Service Designer Localization

Modify the **LocalStrings.properties** file in the panama_pasm.zip file in the ORACLE_HOME\panama\tools\ServiceDesigner\lib directory.

7.5.2 Personalization Portal Localization

Modify the **LocalStrings.properties** file in the WE_HOME/server/classes/oracle/panama/adapter/webui directory.

7.5.3 Localization for LDAP, Mail Adapter, and Personalization from a Device

Localization text for these adapters can be found in **LocalStrings.properties** files in the WE_HOME/server/classes/oracle/panama/adapter directory. The sub-directories are specified.

Table 7–1 LocalStrings.properties Files Details

Location	Contents
/ldap/LocalStrings.properties	Text labels used by the LDAP adapter.
/mail/LocalStrings.properties	Text labels used by the Mail adapter.
/papzlite/LocalStrings.properties	Text labels used by the device interface to the Personalization Portal.

8

Utilities

Utilities describes the XML utilities used for management and deployment. Each section of this document presents a different topic. These sections include:

- Section 8.1, "System Password Encryption / Decryption"
- Section 8.2, "LoadXml"
- Section 8.3, "Upload and Download Utilities"
- Section 8.4, "Xslt"
- Section 8.5, "CopyObjects"

8.1 System Password Encryption/Decryption

The Wireless Edition database password for the repository is stored in clear text in the **System.properties** file by default. The password can be encrypted by running the following tool to modify the **db.connect.string**:

```
encryptPassword.sh WE_HOME/server/classes/oracle/panama/core/admin
```

Then set the following property in the **System.properties** file:

```
passwordEncrypted=true
```

8.1.1 Extensibility

The SystemPasswordEncryptionHook.class identifies the encryption algorithm. To provide extensibility, the class can be used to invoke a user's algorithm in place of the default algorithm.

The default setting is:

```
locator.SystemPasswordEncryptionHook.class=
     oracle.panama.rt.common.SystemPasswordEncryption
```

To use an alternate encryption algorithm, implement the following interface:

```
public interface SystemPasswordEncryptionHook {
  /** Encrypts the text
    * @param text the text to be encrypted
    * @return the encrypted text
    */
public String encrypt(String text);

  /** Decrypts the encrypted text
    * @param encryptedText the encrypted text
    * @return the decrypted text
    */
public String decrypt(String encryptedText);
}
```

This allows you to replace the default algorithm, for example:

```
locator.SystemPasswordEncryptionHook.class=
     oracle.panama.MyEncryption
```

8.2 LoadXml

The LoadXML utility allows you to download and upload Wireless Edition repository objects as XML files.

LoadXml reads from stdin and writes to stdout. All logging and error messages are written to stderr. The XML in the file you import with LoadXml must conform to the Repository XML.

The upload function performs the following:

1. Checks for the objects in the repository by logical unique name.
2. Loads all dependencies.
3. If the objects exist in the repository, LoadXml updates the objects.
4. If the objects do not exist, LoadXml creates them.
5. After each object type is successfully loaded, LoadXml performs a commit. The commit includes all referenced objects (dependencies).

In the unload XML result, all objects have an attribute called _objectId; this is the system unique object key. You must look up objects by unique name attribute and not the object key. If you start the program without giving an option, all options are listed.

LoadXML imports and exports the repository identified by the database connect string in the **System.properties** file. In the development environment, this file is located in the WE_HOME/server/classes/oracle/panama/core/admin directory.

Wireless Edition does not validate the XML file you import into the repository with LoadXml. To avoid errors, work in an XML file that you have exported from the repository. This gives you a "known good" Repository XML framework for adding, removing, and modifying individual elements.

Syntax

```
oracle.panama.core.util.LoadXml [-l username/password] [-x[adgnstu] [expr]] [-c# [-p]] [-r rmi://rmi-host:port/server_name]
```

Options

The LoadXml utility accepts the following options:

Table 8–1 LoadXml Utility

Option	Description
`-l usr/pwd`	Log on to Wireless Edition using a user name and password. If no administrator is defined in the system, the program allows any user to log on. Otherwise, the user must be an administrator to log on.
`-x`	Unload all repository data to `stdout`. The data can be filtered by adding these options to the -x option.
`-a`	Adapter filter.
`-d`	Logical device filter.
`-g`	Group filter.
`-n`	Agent filter.
`-s`	Service filter.
`-t`	Transformer filter.
`-u`	User filter.
`expr`	Name expression filter. This option can include wildcards, such as: [*%?_].
`-c#`	Upload repository data read from `stdin`. The argument # is a number that, if set, causes a commit after the specified number of objects are uploaded. An argument of `0` causes a commit after a complete load of the XML data.
`-p`	This option activates provisioning of user data when uploading. This mode is only used with the -c option. The provisioning upload handles the provisioning DTD, and makes it possible to create, enable, disable, and remove users in the repository. This mode always creates new users. If the `createUserRoot` attribute is set to `YES`, `LoadXml` creates a root folder for each user. The provisioning upload uses streaming to load users, and therefore does not resolve dependencies.

Option	Description
-r	This option ensures that the repository and object cache for a running instance of Wireless Edition are current. The following is sample syntax: -r rmi://stpc250.us.oracle.com:2008/PanamaServer -c0 < bootstrap.xml rmi-host is the machine on which the RMI registry is running. port is the port at which the registry is listening. server-name is the name of the Wireless Edition instance. port and server-name can be found in the **Rmi.properties** file.

Unload Example

In this example, XML data is written to standard output.

```
java oracle.panama.core.util.LoadXml -l adm/adm -x > outputfile.xml
```

Upload Example

In this example, XML data is read from standard input. It must contain all referenced objects.

```
prompt$ java oracle.panama.core.util.LoadXml -l adm/adm -c0 < inputfile.xml
```

8.3 Upload and Download Utilities

You can use the `upload` and `download` utilities to import and export the Wireless Edition repository as an XML file. These utilities invoke `LoadXml`. They are located in the WE_HOME/sample directory of your development environment. You invoke `upload` from a command prompt as follows:

```
upload.bat repository.xml
```

This loads the contents of the file **repository.xml** into the repository. It accesses the repository specified by the connect string in the client-side **System.properties** file.

To `download` a repository:

```
download.bat repository.xml
```

This places the contents of the repository in a file named **repository.xml**.

Note: Please make sure the -l option has been properly specified in either the **upload.bat** or **upload.sh** script file.

The **xmlloader.properties** file specifies the mapping between the object names and their handler Java classes. The objects are specified as fully-qualified from the root, as dot-separated names. For example, the `PanamaObjects` property file is specified as oracle.panama.util.XMLLoader.PGRPHandler.

The **System.properties** file has the following entries to determine behavior.

Table 8–2 Behaviors for the XML Loader in the Systems.property File

Entry	Behavior
`xmlloader.pathExpression`	Specifies the regular expression which determines the path of the objects that you intend to upload and download.
`xmlloader.activityLog`	Specifies the location of logging activity.
`xmlloader.errorLog`	Specifies the location for logging errors.
`xmlloader.createUserRoot`	Determines whether a root folder is to be created for each user when uploading provisioning data.
`xmlloader.commitFrequency`	Frequency in seconds to commit the uploaded information to the back-end repository.
`xmlloader.replaceObjects`	Determines whether a pre-existing objects should be overwritten.

Note: 'All the XML Loader parameters in the System.properties file are default values that are overwritten by the XML Loader API.

8.4 Xslt

The `Xslt` utility is a tool you can use to test stylesheets. `Xslt` uses the XML DOM parser and the XSL processor included with the Oracle XML processor. You can use `Xslt` to apply the stylesheets you create to any XML document.

`Xslt`, which is a command-line utility, reads from standard input and writes its results to standard output.

Synopsis

```
oracle.panama.util.Xslt [stylesheet]
```

Options

The `Xslt` utility takes the following option:

Table 8–3 Options for the Xslt Utility

Option	Description
`stylesheet`	The XSL stylesheet that the utility applies to the XML document.

Xslt Example

```
java oracle.panama.util.Xslt mystylesheet.xsl < myxml.xml
```

8.5 CopyObjects

The `CopyObjects` utility allows you to copy services from one Wireless Edition site to one or more other sites. You can use this utility, for example, to deploy services from a testing and development environment to production servers.

`CopyObjects` sends services to target servers as XML elements. It takes a folder object as a command-line argument. To copy a service, therefore, you must first place the service in a folder. When invoked, `CopyObjects` copies the folder and all its contents to the target Wireless Edition sites you specify at the command line.

Requirements

All source and target Wireless Edition servers must have the Wireless Edition RMI server running.

The basic configuration of adapters and transformers must be the same on the source and target servers. `CopyObjects` only sends services, links, and folders. It does not send configuration data.

Syntax

```
oracle.panama.core.util.CopyObjects [-f folder] [-s source ][targets...]
```

Options

The `CopyObjects` utility takes the following options:

Table 8–4 Options for the CopyObjects Utility

Option	Description
`-f` *folder*	The name of an existing folder in the source system.
`-s` *source*	The source Wireless Edition server. Specify the source in the format: *username*/*password*@/ /*hostname*:*port*/*servername*
targets	The target Wireless Edition servers. Specify targets in the format: *username*/*password*@/ /*hostname*:*port*/*servername*

Example

This example copies a folder and its content from a local server to production sites at `m1` and `m2`.

```
java oracle.panama.core.util.CopyObjects \
          -f /master/finance \
          -s user/pw@//:2008/WEServer \
             user/pw@//m1:2008/WEServer \
             user/pw@//m2:2008/WEServer
```

9

Recommended Settings

Recommended Settings defines recommended parameters for Wireless Edition. Each section of this document presents a different topic. These sections include:

- Section 9.1, "Mod_JServ Configuration"
- Section 9.2, "Recommended Unix Kernel Parameters"
- Section 9.3, "Recommended Unix TCP Parameters"

9.1 Mod_JServ Configuration

To adjust the maximum number of socket connections that Mod_ JServ can handle simultaneously, modify the following parameter in the **jserv.properties** file.

Table 9–1 Mod_JServ Configuration

Name	Value
security.maxConnections	200

9.2 Recommended Unix Kernel Parameters

Oracle recommends that you modify the following kernel parameters in the **/etc/system** file:

Table 9–2 Recommended Unix Kernel Parameters

Parameter	Value
priority_paging	1
rlim_fd_max	8192
rlim_fd_cur	2048
lwp_default_stksize	0x4000
rpcmod:svc_run_stksize	0x4000
tcp:tcp_conn_hash_size	262144
sq_max_size	1600

9.3 Recommended Unix TCP Parameters

The following TCP parameters should be modified through the **ndd** command:

Table 9–3 Recommended Solaris TCP Parameters

Parameter	Value
tcp_rexmit_interval_initial	3000
tcp_rexmit_interval_min	3000
tcp_rexmit_interval_max	10000

Parameter	Value
tcp_ip_abort_interval	60000
tcp_ip_abort_cinterval	60000
tcp_keepalive_interval	120000
tcp_fin_wait_2_flush_interval	16000
tcp_conn_req_max_q	10240
tcp_conn_req_max_q0	10240
tcp_xmit_hiwat	65536
tcp_xmit_lowat	32768
tcp_recv_hiwat	65536
tcp_slow_start_initial	2
tcp_time_wait_interval	32767

10

Property Files

The Wireless Edition supports system-wide extensibility using property files. Each property file stores runtime parameters and processing information for a Wireless Edition component. The Wireless Edition property files are named with the extension **.properties** or **.xml**.

Each section of this document presents a different topic. These sections include:

- Section 10.17, "MailAdapter.properties"
- Section 10.18, "Master.Properties"
- Section 10.19, "positioner.xml"
- Section 10.20, "Geocoders.xml"
- Section 10.21, "Mappers.xml"
- Section 10.22, "Routers.xml"
- Section 10.23, "Strip.properties"
- Section 10.24, "MAgent.properties"
- Section 10.25, "MInstances.properties"
- Section 10.26, "Cookies.properties"

Important: In this document:

- WE_HOME is the directory in which Wireless Edition is installed followed by the "panama" sub-directory. For example:

  ```
  c:\ias10210\panama (Windows)
  /private/ias10210/panama (UNIX)
  ```

Replace "WE_HOME" with the fully-qualified directory path.

- ORACLE_HOME is the directory in which Oracle9*i* Application Server is installed.

10.1 Overview

Property files enable you to plug in new components easily, or extend and configure existing ones. The Wireless Edition includes many property files. The following table lists the property files you can use to configure Wireless Edition.

Note: Verify that the **.properties** files do not contain duplicate entries. If duplicate entries are present, the entry nearest the end of the file has precedence.

Table 10–1 Wireless Edition Properties Files

Property File	Description
System.Properties	Configuration settings for the entire Wireless Edition system.
Notification.properties	Email and SMS server configuration properties.
Rmi.properties	Identifies the server name and listening port of the Wireless Edition RMI server.
oemevent.properties	Properties for event logging using oemevent.
Ftp.properties	Identifies the listening port, timing characteristics, and character set for the repository FTP server.
Spatial.Properties	Location of properties file specifying spatial providers.
www-server.properties	Properties for a runtime probe.
Provisioning.properties	Identifies the common root folder for users' service trees.
useragent.properties	Links logical devices in the Wireless Edition repository to the actual user agent parameter received in an HTTP header. Also specifies a default logical device to use if the device type cannot be determined.
LDAP.properties	Server timing and referral handling properties for LDAP adapter.
AsynchNotification.properties	Outgoing message queue Asynchronous Dequeuer properties.
AsynchRequest.properties	Incoming Asynchronous Request Dequeuer properties.
EncodingSets.properties	Converts the encoding set names from IANA names to Java names.
LocationMark.properties	Specifies which attribute of a location mark should be displayed.
ProxyFirewall.properties	Specifies proxy properties for HTTP or FTP protocols.
MailAdapter.properties	Email adapter properties.
Master.Properties	Read by the master instance when the multiple Java Virtual Machine option is enabled.
positioner.xml	Specifies the automatic positioning capability provider.
Geocoders.xml	Specifies the geocoding provider's preference in descending order.

Property File	Description
Mappers.xml	Specifies the possible mapper capability providers' preferences, in descending order.
Routers.xml	Specifies the routing capability provider's preference in descending order.
Strip.properties	Acquires and converts arbitrary Web content; reads configuration settings from the **Strip.properties** file. Uses the settings in the file to locate classes that process the markup tags in content. Each processing class implements a strip level for the adapter. To add a strip level, create the class that implements the strip level and reference it in. This makes the strip level available to the adapter, without requiring you to make changes to the stripper adapter class.
MAgent.properties	Points to an instance's classes and port number.
MInstances.properties	Refers an instance back to the appropriate **MAgent.properties** file for that instance.
Cookies.properties	Enables the use of cookies and defines the login duration.

10.2 System.Properties

The Wireless Edition system configuration file, **System.properties**, contains parameters and component locators for the entire Wireless Edition system. These settings can be found in **System.properties.templ** and should be copied to **System.properties**. To start the Wireless Edition, you must have this file in the class package `oracle.panama.core.admin`. This package must be in the environment CLASSPATH.

This file is in the WE_HOME/server/classes/oracle/panama/core/admin directory.

The **System.properties** file contains the following parameters:

Table 10–2 Parameters for the System.properties File

Key	Type	Editable	Description
version	String	N	Wireless Edition version.

Key	Type	Editable	Description
instance.identifier	String	Y	Wireless Edition server instance identifier. Must be unique among Wireless Edition server instances and should be modified if multiple instances are running. The default is instance1.
locator.boot.check	Boolean	N	If set to true, the Wireless Edition performs a bootstrap check procedure during system boot. The default is true.
participateInCacheSyncronization	Boolean	Y	Determines whether the instance should participate in Cache Synchronization. The default is false.
masterInstance.url	String	Y	The URL for the master instance's registry. The default is / /localhost:2007.
enable.http.session.binding	Boolean	Y	Enables binding to HTTP session to allow URL rewriting based on JServ load-balancing. The default is false.
db.connect.string	String	Y	A valid connect string, including user name and password. The precise format depends on the driver you use.
db.driver	String	Y	The JDBC driver that the Wireless Edition uses to access the repository. Possible values are THIN,V7, V8, INTERNAL, and CUSTOM. The default is THIN. If set to CUSTOM, you must also set the db.driver.class parameter.

Key	Type	Editable	Description
db.session.min	Integer	Y	The minimum or optimal number of open database sessions. This parameter is used to tune the session pool. It can increase when necessary, but always attempts to return to the specified number of concurrent open database sessions.
db.connect.minConnections	Integer	Y	The minimum number of connections for the connection pool. The default is 5.
db.connect.maxConnections	Integer	Y	The maximum number of connections for the connection pool. The default is 100.
db.connect.incConnections	Integer	Y	The incremental allocation of new connections to the connection pool. The default is 1.
name.separator	Character	N	The separator used in the service path. This cannot be modified after installation. The default is a forward slash: "/".
repository.session.check.interval	Seconds	Y	Time interval for unused connections to be found and released to the connection pool. The default is 1.
passwordEncrypted	Boolean	Y	Declares whether the password in the database connect string is in clear text or encrypted. The default is false.
locatorSystemPasswordEncryptionHook.class	Class Name	Y	If the value of passwordEncrypted is true, this declares which class implements the password encryption hook. The default is oracle.panama.rt.common.SystemPasswordEncryption.

Key	Type	Editable	Description
locator.objectcache.class	Class Name	N	The object cache implementation class. The default is oracle.panam.core.util.ObjectCacheHardWeakImpl.
objectcache.check.timetolive	Seconds	Y	The time-to-live, in seconds, of a persistent object. After this time, the Wireless Edition reconstructs the object. The default is 600.
objectcache.check.interval	Seconds	Y	The time required for the cache monitor to check the cache. If set to -1, the Wireless Edition does not invoke the cache monitor and the cache is not cleared. The default is 60.
session.expiration.time	Seconds	Y	The time-to-live attribute of a session. The default is 600.
session.expiration. checkinterval	Seconds	Y	The time required for the session monitor to check an open session. The default is 60.
locator.persistent.class	Class name	N	The locator implementation for persistent objects. The default is oracle.panama.core.PersistentLocatorImpl.
locator.logger.class	Class name	N	The logger implementation. The default is oracle.panama.core.admin.LoggerImpl.
locale.language	String	Y	The language used. This is represented by the ISO code for the representation of names of languages. The default is "en".
locale.country	String	Y	The location. This uses the ISO 3166 country code. The default is "US".

Key	Type	Editable	Description
log.level	String list	Y	Log level. Can contain any of the following: error, warning, notify, or transaction. The default is "warning, error, notify".
log.directory	Path	Y	Log file directory.
log.file.name	File name	Y	Log file name pattern. The default is sys_panama.log.
log.file.maxsize	Integer	Y	Maximum number of log records in the same file. The default is 1000.
log.tx.directory	Path	Y	Transaction log file directory. The default is /tmp.
log.tx.file.name	File name	Y	Transaction log file name pattern. The default is tx.panama.log.
log.tx.file.maxsize	Integer	Y	Maximum number of transaction entries in the same file. The default is 1000.
log.tx.file.pattern	Formatted string	Y	Pattern used to write transaction log records. The default is {user id}{user external id}"{service name path}"{cost}.
log.tx.min.cost	Integer	Y	Minimum cost for the services to be logged. The value specified is compared with the service cost. Only services with a cost greater than this value are logged. If an invalid number is specified here, the default value of 0 is used.
log.console	Boolean	Y	Determines if true log output is written to `stderr`. The default is false.
algorithm.password	Java algorithm name	Y	Hash algorithm for passwords. The default is SHA-1.

Key	Type	Editable	Description
locator.external.hook.class	Class name	Y	Default implementation of `UserAuthenticationHook`. The default is oracle.panama.core.PanamaExternalHookDefImpl.
locator.useragent.class	Class name	Y	Default implementation of the device recognition class. The default is oracle.panama.core.xform.UserAgentImpl.
locator.request.daemon .classes	Class name list	Y	System daemons to start. See **System.properties** file for options.
locator.request.queue.class	Class name	N	Asynchronous request queue implementation. The default is oracle.panama.core.rdbms.AsynchNotificationQueueImpl.
locator.notification.queue.class	Class name	N	Asynchronous notification queue implementation. The default is oracle.panama.util.NotificationDispatcherImpl.
locator.notification.dispatcher.class	Class name	Y	Default notification engine implementation. Default is oracle.panama.util.NotificationDispatcherImpl.
locator.postprocessor.class	Class name	N	Transformer post-processor implementation. The default is oracle.panama.core.xform.PostProcessor.
locator.provisioning.class	Class name	Y	TDefault implementation of the provisioning hook. The default is oracle.panama.core.util.ProvisioningHookImpl.
locator.folder.renderer.hook.class	Class Name	Y	Hook for a folder renderer. The default value is oracle.panama.rt.common.FolderRenderer.

Key	Type	Editable	Description
locator.home.folder.sorter.hook.class	String	Y	Value used by the oracle.panama.rt.common.HomeFolderSorter class to sort the services in a user's home folder. The default is USER_SERVICES_FIRST.
locator.service.visibility.hook.class	Class Name	Y	Declares a hook to show or hide a service when Wireless Edition is started. The default is oracle.panama.rt.common.ServiceVisibility.
locator.location.service.visibility.hook.class	Class Name	Y	Declares a hook to show or hide the contents of a folder based on its current location. The default is oracle.panama.rt.common.LocationServiceVisibility.
locator.authentication.hook.class	Class Name	Y	Declares the hook for user authentication. The default is oracle.panama.rt.common.Authenticator.
locator.authorization.hook.class	Class Name	Y	Declares the hook for user service authorization. The default is oracle.panama.rt.common.Authorizer.
locator.device.identification.hook.class	Class Name	Y	Declares the hook for identifying a logical device. The default is oracle.panama.rt.hook.DeviceModels.
locator.service.visibility.hook.class	Class Name	Y	Declares the hook to check for the show or hide status when Wireless Edition starts. The default is oracle.panama.rt.common.ServiceVisibility.

Key	Type	Editable	Description
locator.session.id.hook.class	Class Name	Y	Declares the hook for generating the session ID. The default is oracle.panama.rt.common.SessionIdGenerator.
locator.listener.registration.hook.class	Class Name	Y	Declares the hook for the event registration listener. The default is oracle.panama.rt.common.ListenerRegistration.
locator.caller.location.hook.class	Class Name	Y	Declares the hook which acquires the user's current location. The default is oracle.panama.rt.common.CallerLocator.
subscriber.id.request.parameter.name	String	Y	Predefined request parameter used to acquire the user's subscriber ID.
locator.subscriber.id.hook.class	Class Name	Y	Declares a hook to acquire a subscriber's ID.
locator.pre.processor.hook.class	Class Name	Y	Declares a hook to be invoked before device transformation.
locator.post.processor.hook.class	Class Name	Y	Declares a hook to be invoked after device transformation.
locator.combined.listener.classes	Class Name	Y	Declares a listener for the request, session and response events.
locator.session.listener.classes	Class Name	Y	Declares a class to implement the SessionListener interface, or oracle.panama.core.admin.SystemLogger to perform database session logging. The properties event.session.begin and event.session.end must be set to 1 to enable them.

Key	Type	Editable	Description
locator.request.listener.classes	Class Name	Y	Declares a class to implement the RequestListener interface, or oracle.panama.core.admin.SystemLogger to perform database session logging. The event.request.begin and event.request.end, and event.after.request properties must be set to 1 to enable them
locator.response.listener.classes	Class Name	Y	Declares a class to implement the ResponseListener interface.
event.before.request	Boolean	Y	Declares a request event to be "just received." The default is false, or disabled. Set to true to enable.
event.request.begin	Boolean	Y	Declares a request event to be "begin being processed". The default is false, or disabled. Set to true to enable.
event.request.end	Boolean	Y	Declares a request event to be "request has been completely processed". The default is false, or disabled. Set to true to enable.
event.service.begin	Boolean	Y	Declares a request event to be "before the adapter is invoked". The default is false, or disabled. Set to true to enable.
event.service.end	Boolean	Y	Declares a request event to be "adapter execution complete". The default is false, or disabled. Set to true to enable.
event.transform.begin	Boolean	Y	Declares a request event to be "before the transformation". The default is false, or disabled. Set to true to enable.
event.transform.end	Boolean	Y	Declares a request event to be "transformation complete". The default is false, or disabled. Set to true to enable.

Key	Type	Editable	Description
event.request.error	Boolean	Y	Declares a request event to be "error occurs during request processing". The default is false, or disabled. Set to true to enable.
event.after.request	Boolean	Y	Declares a request event to be "request object has been released". The default is false, or disabled. Set to true to enable.
event.response.error	Boolean	Y	Declares a response event to be "error in response object. The default is false, or disabled. Set to true to enable.
event.before.session	Boolean	Y	Declares a session event to be "before session starts". The default is false, or disabled. Set to true to enable.
event.session.begin	Boolean	Y	Declares a session event to be "session has been validated". The default is false, or disabled. Set to true to enable.
event.session.end	Boolean	Y	Declares a session event to be "session has expired (implicitly and explicitly)". The default is false, or disabled. Set to true to enable.
event.after.session	Boolean	Y	Declares a session event to be "session object has been released". The default is false, or disabled. Set to true to enable.
system.java.protocol. handler.pkgs	Class name	Y	The Java class package that provides SSL (Secure Sockets Layer) support. Default is com.sun.net.ssl.internal.www.protocol.

Key	Type	Editable	Description
order.services	String	Y	The sorting order for the Wireless Edition services, folders, and bookmarks on the output devices. The default is ORDER_SEQNO_ASC, ORDER_NAME_ASC.
SystemLog.enableServiceLogging	Boolean	Y	Specifies whether Service Logging should be enabled. The default is false.
SystemLog.enableSessionLogging	Boolean	Y	Specifies whether Session Logging should be enabled. The default is false.
SystemLog.logger.maxSize	Integer	Y	Specifies the maximum number of entries that will be buffered before flushing the entries to the database.
SystemLog.logger. wakeupFrequency	Minutes	Y	Specifies the number of minutes after which the logger thread wakes up to flush the entries in the database tables. The default is 10.
SystemLog.db.connect.string	String	Y	JDBC connect string for logging to a second database.
transformer.poolsize	Integer	Y	Specifies the number of XML transformers in the pool. The pool size should be around 1.5 times the number of CPUs. The default is 10.
URLStreamHandlerFactory.clas	Class Name	N	URL StreamHandlerFactory used to create URLs. The default value is oracle.panama.core.util.URLStreamHandlerFactoryImpl.
folder.subfolder.pagesize	String	Y	Control number of services to be displayed under a folder at any one time.
folder.subfolder.nextlabel	String	Y	Label used to link the next set of folder.subfolder.pagesize services if the folder.subfolder.pagesize property is set.

10.3 Notification.properties

This file provides basic configuration for the notification engine.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–3 Parameters of the Notification.properties File

Key	Type	Editable	Description
dispatcher.logical.device.email.name	String	N	Repository name of the logical device for email. The default is EMAIL.
dispatcher.logical.device.sms.name	String	N	Repository name of the logical device for short messaging service (SMS). The default is SMS.
sms.account.id	Number	Y	Wireless Edition System Account ID, if any, for the SMS-C.
sms.account.password	String	Y	Password for the sms.account.id
sms.server.name	String	Y	Name of the host where SMS-C is running.
sms.server.port	Integer	Y	Port number on which the SMS-C is listening.
sms.driver.class	String	Y	Use oracle.panama.util.sms.SmsUCPDriver for UCP protocol. Use oracle.panama.util.sms.SmsSMPPDriver for SMPP protocol. The default is the UCP protocol.
sms.ucptype	String	Y	Specify the UCP command set type. The default is 01. For 51 operations sent, use 51.
sms.message.maxchunks	Number	Y	Max number of SMS chunks per message. The default is 3.
sms.message.maxsize	Number	Y	Maximum number of bytes per message. The default is 160.
sms.smpp.system.id	String	Y	System account identification for the Short Message Peer to Peer protocol (SMPP) SMS-C.
sms.smpp.system.password	String	Y	Password for the sms.smpp.system.id account.
sms.smpp.system.type	String	Y	SMPP system type.

Key	Type	Editable	Description
mail.server.name	String	Y	Send Mail Transport Protocol (SMTP) server host name if email is used as the means of notification.
mail.domain	String	Y	Optional property. The mail domain name used if required by the SMTP server.
mail.from	String	Y	Mandatory property. The "From" mail address.
mail.organization	String	Y	Optional property. The organization.
mail.subject	String	Y	Mandatory property. The mail subject.
mail.content.transfer.encoding	String	Y	Mandatory. The Content-Transfer-Encoding.

10.4 Rmi.properties

You can manage objects in the Wireless Edition repository from a remote client using the RMI (Remote Method Invocation) server. The RMI server is a daemon that enables you to access and manipulate Wireless Edition objects using an RMI interface. Currently, the Wireless Edition automatically generates the RMI component from the interface definitions using introspection of the compiled Java interfaces. See `oracle.panama.core.util.GenerateRMI` for more information.

An RMI session is stateless and each update call is completed with a commit or rollback. For this reason, you can manage all persistent objects from a remote client. Management actions take immediate effect when the RMI server works at the object level of the system. To connect to an RMI daemon, you must provide a valid administrator user name and password.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

The **Rmi.properties** file has the following parameters:

Table 10–4 Parameters of the Rmi.properties File

Key	Type	Editable	Description
rmi.server.port	Integer	Y	TCP port on which the daemon listens. The default is 2008.

Key	Type	Editable	Description
rmi.server.name	String	Y	Name of the RMI server. The default is PanamaServer.

The implementation is managed by the daemon:

```
oracle.panama.core.rmi.server.ServerImpl
```

10.5 oemevent.properties

Asynchronous notification of error information is provided using the event framework provided by Oracle Enterprise Manager. In case of an error, the Wireless Edition server generates an Oracle Enterprise Manager event at the node where the error occurred providing all the necessary details for the fault diagnosis.

> **Note:** An OEM agent must be running on the node where oemevent is invoked. The event must be defined as an unsolicited event and the node must be added as a monitored destination.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

The **oemevent.properties** file has the following parameters:

Table 10–5 Parameters of the oemevents.properties File

Key	Type	Editable	Description
oemevent.path	string	Y	Complete path name for the oemevent executable.
oemevent.event. name	string	Y	Unsolicited event name as defined in the console.
oemevent.object. name	string	Y	Object name specified in the event definition.
oemevent.event level	String	Y	Event level specified in the event definition. The level should be defined as "2" and specified as "alert".

Note: You must be connected to the spatial database to use the spatial adapters. In addition, data stored in the spatial database (images and geometry) must be purchased from a third-party provider.

10.6 Ftp.properties

The FTP server is a daemon that publishes the repository as an FTP file system. The repository folder is represented as an FTP directory, and repository objects are represented as FTP files. The FTP server makes it possible to upload and download users, services, and other repository objects as XML documents.

FTP support allows you to:

- Authenticate users (only administrators can use this utility).
- List directories and files.
- Download XML-enabled persistent objects.
- Upload XML-enabled persistent objects.
- Create subdirectories (service tree folders only).
- Delete directories (service tree folders only).
- Delete any object.
- Rename directories and files.

Note: When uploading new objects, you do not need to assign values to the unique name or `objectid_` attributes. The upload feature uses "update or create" logic. If either the unique name or `objectid_` attribute is set to the same value as an existing object, then the existing object is updated. The unique name of a service is the `url` attribute. For all other objects, the unique name is the `name` attribute.

This file is in the WE_HOME/server/classes/oracle/panama/core/admin directory.

The **Ftp.properties** file has the following parameters:

Table 10–6 Parameters of the Ftp.properties File

Key	Type	Editable	Description
ftp.server.port	Integer	Y	TCP listening port. The default is 9100.
ftp.connection.timeout	Seconds	Y	Client connection timeout in seconds, a value of 0 is equal to "no timeout." The default is 30.
ftp.charset	String	Y	Character set used to send "Object" names, it must be a valid Java character set. The default is ISO8859_1.

10.7 Spatial.Properties

You must modify the **.xml** files to get third-party provider information. If you wish to provide your own spatial data, you must add yourself as a provider to the **.xml** file.

To enable Wireless Edition to render geocoding services from a third-party provider:

1. In the WE_HOME/server/classes/oracle/panama/spatial directory, rename **spatial.properties.tmpl** to **spatial.properties**.
2. Modify spatial.properties to indicate the file to be used for geocoding, routing, ypcategory, ypproviders, mapping, and positioner xml properties.

Table 10–7 Parameters of the spatial.properties File

Key	Type	Editable	Description
file.providers.config.xml.geocoding	String	Y	XML file for the geocoding.
file.providers.config.xml.routing	String	Y	XML file for the routing.
file.providers.config.xml.ypcategory	String	Y	XML file for the yp category mapping.
file.providers.config.xml.ypproviders	String	Y	XML file for the yp providers.
file.providers.config.xml.mapping	String	Y	XML file for the mapper providers.
file.xml.positioner	String	Y	XML file for the automatic positioning provider.

10.8 www-server.properties

The probe is a daemon that displays runtime information about:

- Log files
- Active objects
- Active sessions
- Active threads

The probe performs simple runtime management including:

- Shutting down an active object instance.
- Refreshing persistent attributes on an active object.
- Setting the global debug flag to on or off.
- Reloading properties for device recognition when adding a new device to a running system.

By default, Wireless Edition installation process installs and initiates the probe runtime monitor with the listening port 8090. To access the probe, from a browser enter the following URL:

http:/ /*host_name*:8090

This file is in the WE_HOME / server / classes / oracle / panama / core / probe directory.

The **www-server.properties** file has the following parameters:

Table 10–8 Parameters of the www-server.properties File

Key	Type	Editable	Description
port	Integer	Y	TCP listening port.
root	Path	Y	Root directory for log files.
timeout	Milliseconds	Y	Timeout for a request TCP socket.
workers	Integer	Y	Number of working request threads to start up.
trusted	IP List	Y	List of "trusted" IP addresses. Each IP address must be separated with a semi-colon, ";". The asterisk, "*"' wildcard may be used. An empty statement grants trusted status to all.

The following daemon manages the implementation:

```
oracle.panama.core.probe.WebServer
```

10.9 Provisioning.properties

This property file is used by the Provisioning adapter and the Personalization Portal.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–9 Parameters of the Provisoning.properties File

Key	Type	Editable	Description
common.root	Folder Name	Y	Common root folder for the user's home folders. A new subfolder is created for every new user. The default is /Users Home.
common.root.owner	String	Y	Owner of the subfolder for users. Subfolders must have an owner which should have administrator privileges. The default is Administrator.
common.group.list	String	Y	List of groups the new user should belong to, for example <group_name, group_name,. . . > If a group does not exist, it is created automatically.
label.create_user label.user_exist label.user_created label.search_user label.view_user label.edit_user label.change_user label.user_changed label.remove_user label.user_removed label.attribute label.value label.error	String	Y	Text label for the Provisioning adapter, for use in localization.

10.10 useragent.properties

This property file lists the mapping from the HTTP user agent to the Wireless Edition logical device.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–10 Parameters of the UserAgents.properties File

Key	Type	Editable	Description
default.logicaldevice	Device	Y	Default logical device if no mapping is found. The default is WML11.
[ID].useragent	String	Y	UserAgent starting string from HTTPRequest.
[ID].logicaldevice	String	Y	Logical device name to be used.

10.11 LDAP.properties

This property file is read by the LDAP adapter. It is in the WE_HOME / server / classes / oracle / panama / adapter / ldap directory.

Table 10–11 Parameters of the LDAP.properties File

Key	Type	Editable	Description
auto_referral	Boolean	Y	Sets Wireless Edition to use automatic referral. The default value is true, or enabled. Set to false to disable.
max_search_time	Integer	Y	Maximum allowed search time in milliseconds. The default is 100.
max_result_count	Integer	Y	Maximum results from a search. The default is 1000.

10.12 AsynchNotification.properties

This is the property file read by the asynchronous notification Dequeuer. It is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–12 Parameters of the AsynchNotification.properties File

Key	Type	Editable	Description
worker.classname	Class Name	N	Implementation class for the asynchronous notification Dequeuer. The default is oracle.panama.core.parm.asynch.AsynchNotificationImpl.
init.pool.size	Integer	Y	Thread pool size at startup. The default is 1.
minimum.pool.size	Integer	Y	Minimum thread pool size. The default is 1.
server.listen.port	Integer	Y	Asynchronous notification Dequeuer management server listening port. The default is 55000.
server.listen.host	Host Name	Y	Host which is running notification Dequeuer management. The default is localhost.

10.13 AsynchRequest.properties

This is the property file read by the asynchronous request Dequeuer. It is in the WE_HOME/server/classes/oracle/panama/core/admin directory.

Table 10–13 Parameters of the AsynchRequest.properties File

Key	Type	Editable	Description
worker.classname	Class Name	N	Implements the asynchronous request Dequeuer.
init.pool.size	Integer	Y	Thread pool size at startup. The default is 1.
minimum.pool.size	Integer	Y	Minimum thread pool size. The default is 1.
server.listen.port	Integer	Y	Asynchronous notification Dequeuer management server listening port. The default is 55000.
server.listen.host	Host Name	Y	Host which is running notification Dequeuer management. The default is localhost.

10.14 EncodingSets.properties

This file is used for converting the encoding set names from IANA names to Java names. The order of the entries is not important, but the entries must be unique. If the IANA name and the Java name are identical, no new entry is necessary.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

The IANA standard is available on the Information Sciences Institute web site. The Java encoding set names are published on the Javasoft web site.

Table 10–14 Parameters of the EncodingSets.properties File

Key	Description
[ID].iananame	IANA name for the encoding set.
[ID].javaname	Corresponding Java name for the encoding set.

10.15 LocationMark.properties

The **LocationMark.properties** file specifies which attribute of a location mark should be displayed. If you have a specific property file for the location you are in, it is used in place of the default **LocationMark.properties** file. For example, the properties file for locations described in US English is **LocationMark_en_US.properties**. Attributes which are set to true are displayed in the Personalization Portal, and on mobile devices connected to the Personalization Portal for personalization.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–15 Parameters of the LocationMark.properties File

Key	Type	Editable	Description
display.Name	String	Y	Name of the location mark.
display.Label	String	Y	Label of the location mark.
display.Description	String	Y	Description of the location mark.
display.CompanyName	String	Y	Company name for the location mark.
display.AddressLine1	String	Y	First address line of the location mark.
display.AddressLine2	String	Y	Second address line of the location mark.
display.AddressLastLine	String	Y	Last address line of the location mark.
display.Block	String	Y	Block of the location mark.
display.City	String	Y	City of the location mark.

Key	Type	Editable	Description
display.County	String	Y	County of the location mark.
display.State	String	Y	State of the location mark.
display.Country	String	Y	Country of the location mark.
display.PostalCodeExt	String	Y	Postal extension code of the location mark.
display.PostalCostCode	String	Y	Postal code of the location mark.

10.16 ProxyFirewall.properties

This property file specifies the proxy properties used by Wireless Edition when HTTP or FTP protocols are required. If no proxy server is in use, no modifications are necessary.

Table 10–16 Parameters of the ProxyFirewall.properties File

Key	Type	Editable	Description
http.use.proxy	Boolean	Y	Declares the use of an HTTP proxy server. If this is false then proxyhost and proxy.port settings are ignored. The default is false.
http.proxy.host	Host Name	Y	Proxy host name. The default is localhost.
http.proxy.port	Port Number	Y	Proxy server port number. The default is 80.
http.non.proxy.host	String	Y	Lists hosts that should not use a proxy server. Each host name is separated by the pipe character, "|". The default is localhost|127.0.0.1.
ftp.use.proxy	Boolean	Y	Declares the use of an FTP proxy server. If the is false then ftp.proxy.host and ftp.proxy.port settings are ignored. The default is false.
ftp.proxy.host	Host Name	Y	FTP proxy host name. The default is localhost.
ftp.proxy.port	Port Number	Y	FTP proxy server port number. The default is 80.

Key	Type	Editable	Description
authentication.set	Boolean	Y	Declares whether the proxy server requires authentication. The default is false.
authentication.username	String	Y	User name if authentication is required.
authentication.password	String	Y	Password if authentication is required.

10.17 MailAdapter.properties

This is the properties file for the email adapter. It is in the WE_HOME/server/classes/oracle/panama/adapter/mail directory.

Table 10–17 Parameters of the MailAdapter.properties File

Key	Type	Editable	Description
JavaMail.USER_DISPLAY_NUM	Integer	Y	Number of message headings displayed at one time under a message folder. The default is 6.
JavaMail.EMAIL_CONTENT_NUM	Integer	Y	Number of characters displayed for a long message. The default is 100.

10.18 Master.Properties

This property file is read by the master instance when the Multiple Virtual Machine option is used. It is in the WE_HOME/server/classes/oracle/panama/master directory.

Table 10–18 Parameters of the Master.properties File

Key	Type	Editable	Description
rmi.registry.port	Integer	Y	Master instance RMI registry port. The default is 2007.
db.connect.string	String	Y	Database connect string.
db.driver	JDBC Driver	Y	Specifies which of the following type of JDBC driver will be used, THIN, V7, or V8. Default is THIN.
rmiThreadPool.minThreads	Integer	Y	Minimum number of threads in the RMI thread pool. The default is 10.

Key	Type	Editable	Description
rmiThreadPool.minThreads	Integer	Y	Maximum number of threads in the RMI thread pool. The default is 20.
rmiThreadPool.idleTimeout	Integer	Y	Idle timeout, in minutes, for the thread. The default is 10.
remoteObjects.length	Integer	Y	Default is 1.
remote.Object1.name	String	N	Default is CacheSynchServer.
remoteObject1.implementation	String	N	Default is oracle.panama.master.CacheSyncServer.
ptginstances.length	Integer	Y	Number of instances that will participate in the multiple Java Virtual Machine synchronization.
ptginstance[n]rmi.host	String	Y	Host name from which the *n*th instance is being run.
ptginstance[n]rmi.port	Integer	Y	Port number from which the *n*th instance is listening.
ptginstance[n]rmi.objectName	String	N	Default value is CacheSynchClient.

10.19 positioner.xml

This XML file specifies the automatic positioning capability provider. It is in the WE_HOME / server / classes / oracle / panama / mp directory.

Table 10–19 Parameters of the positioner.xml File

Key	Description
ProviderName	Provider name.
ProviderImpl	Class that implements the provider.
the rest	Additional arguments read by the provider.

10.20 Geocoders.xml

This XML file specifies the geocoding provider's preference in descending order. It is in the WE_HOME / server / classes / oracle / panama / spatial / geocoder directory.

Table 10–20 Parameters of the Geocoders.xml File

Key	Description
ProviderName	Provider name.
ProviderImpl	Class that implements the provider.
the rest	Additional arguments read by the provider.

10.21 Mappers.xml

This XML file specifies all the possible mapper capability providers. The sequence in the XML file determines the mapper provider's preference in descending order. It is in the WE_HOME / server / classes / oracle / panama / spatial / mapper directory.

Table 10–21 Parameters of the mappers.xml File

Key	Description
ProviderName	Provider name.
ProviderImpl	Class that implements the provider.
the rest	Additional arguments read by the provider.

10.22 Routers.xml

This XML file specifies all the possible routing capability providers. The sequence in the XML file determines the routing provider's preference in descending order. It is in the WE_HOME / server / classes / oracle / panama / spatial / region directory.

Table 10–22 Parameters of the routers.xml File

Key	Description
ProviderName	Provider name.
ProviderImpl	Class that implements the provider.
the rest	Additional arguments read by the provider.

10.23 Strip.properties

The **Strip.properties** file notifies the stripper adapter of proxy settings and strip levels. This properties file also defines classes which provide the stripper adapter

with a translation table to convert a target entity into an appropriate string. It in the WE_HOME / server / classes / oracle / panama / adapter / stripper directory.

Table 10–23 Parameters of the Strip.properties File

Key	Type	Editable	Description
stripper.useProxy	Boolean	Y	Enables the use of a proxy server. The default is true.
stripper.proxyServe	Host Name	Y	Name of the proxy server. The default is proxy.domain.com.
stripper.proxyPort	Integer	Y	Port number of the proxy server. The default is 80.
stripper.noProxySuffixes	Host Name	Y	Name of any proxy suffix to be specified. The default is domain.com.

After retrieving content, the stripper adapter calls a Java class to process the markup tags in the content. The stripper adapter locates the processing classes, or strip levels, that are available to it in the **Strip.properties** file.

Table 10–24 Strip Processing Classes

Level	Description	Implementing Class
0	Retains all tags in the content.	Strip0
1	Strips all tags from the content.	Strip1

Example

```
# Classes used for stripping
# Strip nothing
stripper.strip0 = oracle.panama.adapter.stripper.Strip0

# Strip everything
stripper.strip1 = oracle.panama.adapter.stripper.Strip1

# Translation table for translating "&target;" entities => String
# Does not recurse - e.g. &amp;target; will not work
oracle.panama.adapter.stripper.Strip1.useTranslationTable = true
oracle.panama.adapter.stripper.Strip1.targets = auml Auml aring Aring
ouml Ouml amp Amp nbsp euml Euml #150
oracle.panama.adapter.stripper.Strip1.targets.auml = ä
oracle.panama.adapter.stripper.Strip1.targets.Auml = Ä
```

```
oracle.panama.adapter.stripper.Strip1.targets.aring = å
oracle.panama.adapter.stripper.Strip1.targets.Aring = Å
oracle.panama.adapter.stripper.Strip1.targets.ouml = ö
oracle.panama.adapter.stripper.Strip1.targets.Ouml = Ö
oracle.panama.adapter.stripper.Strip1.targets.amp = &
oracle.panama.adapter.stripper.Strip1.targets.Amp = &
oracle.panama.adapter.stripper.Strip1.targets.AMP = &
oracle.panama.adapter.stripper.Strip1.targets.nbsp = -
oracle.panama.adapter.stripper.Strip1.targets.euml = ë
oracle.panama.adapter.stripper.Strip1.targets.Euml = Ë
oracle.panama.adapter.stripper.Strip1.targets.#150 = -
```

10.24 MAgent.properties

The **Magent.properties** file on both the client and server side points to an instance's classes and port. Users can have any number of instances but each instance's information is stored in a separate property file.

This file is in the WE_HOME / server / classes / oracle / panama / config directory.

Table 10–25 Parameters of the MAgent.properties File

Key	Type	Editable	
magent.server.host	Host Name	Y	Server host name.
magent.server.port	Integer	Y	Service port number. This port number should match the port number used by the RMI Server listed in **RMI.properties**. The default is 2080.
magent.server.name	String	Y	Service name. The default is Magent.

10.25 MInstances.properties

Every database instance needs an <instance name > = <property file name> entry added to the **MInstances.properties**.

This file is in the WE_HOME / server / classes / oracle / panama / config directory.

The RMI client uses this information to talk to the corresponding server. Each instance is monitored using a **Magent.properties** file which contains the Wireless Edition server host name, port, and service name. In the case of multiple instances, these are identified by these values. The **Minstances.properties** file should point the

instance name to the corresponding **Magent.properties** file. For example, for two instances:

```
PTG1=oracle.panama.magent.config.Magent1
PTG2=oracle.panama.magent.config.Magent2
```

10.26 Cookies.properties

This property file enables Wireless Edition to use cookies and sets the duration of the login session.

This file is in the WE_HOME / server / classes / oracle / panama / core / admin directory.

Table 10–26 Parameters of the Cookies.properties File

Key	Type	Editable	Description
cookie.use.userinfo	Boolean	Y	When true, this parameter instructs the system to use cookies when talking to clients that have cookie support. The default is true.
cookie.userinfo.maxage	Integer	Y	Duration of a session, in seconds, in which the user is not required to re-enter a user name or password. The default is 86400.

Index

R

S

T

U

W

X