Programming WebLogic Server for Wireless Services

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Date</th>
<th>Software Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>June 28, 2002</td>
<td>BEA WebLogic Server Version 7.0</td>
</tr>
</tbody>
</table>
Contents

About This Document

Audience................................................................................................................v
e-docs Web Site................................................................................................... vi
How to Print the Document................................................................................. vi
Related Information............................................................................................. vi
Contact Us!......................................................................................................... vii
Documentation Conventions.............................................................................. vii

1. Introduction

Overview ........................................................................................................... 1-1
Wireless Data Protocols .................................................................................... 1-2
The WAP Protocol at a Glance ......................................................................... 1-3
The i-Mode Protocol at a Glance....................................................................... 1-3
Other Wireless Data Protocols .......................................................................... 1-4
Other Wireless Markup Languages ................................................................... 1-4
Evolution of Wireless Data Protocols and Wireless Markup Languages........ 1-5
Additional Information...................................................................................... 1-5

2. Using WAP with WebLogic Server

Overview ........................................................................................................... 2-1
WAP Application Environment ........................................................................ 2-2
  WML .............................................................................................................. 2-2
  WMLScript................................................................................................. 2-3
WAP Gateway ................................................................................................... 2-3
  WAP Gateway Functionality ....................................................................... 2-4
  WAP Gateway Security and Security Concerns ........................................ 2-4
  WAP Gateway Vendors ............................................................................. 2-5
3. Using i-Mode with WebLogic Server

Overview ........................................................................................................... 3-1
i-Mode Markup Language ................................................................................. 3-2
i-Mode Gateway ................................................................................................ 3-2
Additional Resources .......................................................................................... 3-3
About This Document

This document explains how to use the BEA WebLogic Server™ platform to design and write Web applications that interface not only with the traditional desktop browser but also with the different types of wireless devices.

This document is organized as follows:

- **Chapter 1, “Introduction,”** provides the basic information you need to know before using WebLogic Server to extend Web applications to wireless subscribers.

- **Chapter 2, “Using WAP with WebLogic Server,”** discusses how to provide content suitable for the Wireless Application Protocol (WAP) application environment and how to configure and use WebLogic Server with a WAP Gateway.

- **Chapter 3, “Using i-Mode with WebLogic Server,”** discusses how to provide content suitable for the i-Mode application environment and how to configure and use WebLogic Server with an i-Mode Gateway.

- **Chapter 4, “Writing Web Applications to Include Wireless Subscribers,”** demonstrates through example how to use WebLogic Server to extend Web applications to wireless subscribers.

Audience

This document is written for application developers who are interested in building transactional Java applications that run in the WebLogic Server environment. It assumes a familiarity with WebLogic Server and Java™ 2, Enterprise Edition (J2EE) programming and wireless Web technologies.
BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation.

How to Print the Document

You can print a copy of this document from a Web browser, one main topic at a time, by using the File—>Print option on your Web browser.

A PDF version of this document is available on the WebLogic Server documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the WebLogic Server documentation Home page, click Download Documentation, and select the document you want to print.


Related Information

The BEA corporate Web site provides all documentation for WebLogic Server. Other WebLogic Server documents that you may find helpful when using WebLogic Server to write application services are:

- **Introduction to BEA WebLogic Server** at
  http://e-docs.bea.com/wls/docs81b/intro/index.html

- **Administration Guide** at
  http://e-docs.bea.com/wls/docs81b/adminguide/index.html

- **Programming WebLogic XML** at
  http://e-docs.bea.com/wls/docs81b/xml/index.html
Contact Us!

Your feedback on BEA documentation is important to us. Send us e-mail at docsupport@bea.com if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the documentation.

In your e-mail message, please indicate the software name and version you are using, as well as the title and document date of your documentation. If you have any questions about this version of BEA WebLogic Server, or if you have problems installing and running BEA WebLogic Server, contact BEA Customer Support through BEA WebSupport at http://www.bea.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+Tab</td>
<td>Keys you press simultaneously.</td>
</tr>
<tr>
<td><em>italics</em></td>
<td>Emphasis and book titles.</td>
</tr>
</tbody>
</table>

Programming WebLogic Server for Wireless Services    vii
<table>
<thead>
<tr>
<th>Convention</th>
<th>Usage</th>
</tr>
</thead>
</table>
| **monospace** text | Code samples, commands and their options, Java classes, data types, directories, and file names and their extensions. Monospace text also indicates text that you enter from the keyboard. 

*Examples:*
- `import java.util.Enumeration;`
- `chmod u+w * config/examples/applications .java config.xml`
- `float`

| **monospace italic** text | Variables in code. 

*Example:*
- `String CustomerName;`

| **UPPERCASE** TEXT | Device names, environment variables, and logical operators. 

*Examples:*
- `LPT1`
- `BEA_HOME`
- `OR`

| `{ }` | A set of choices in a syntax line. 

| `[ ]` | Optional items in a syntax line. *Example:*

- `java utils.MulticastTest -n name -a address [-p portnumber] [-t timeout] [-s send]`

| `|` | Separates mutually exclusive choices in a syntax line. *Example:*

- `java weblogic.deploy [list|deploy|undeploy|update] password (application) (source)`

| `...` | Indicates one of the following in a command line:

- An argument can be repeated several times in the command line.
- The statement omits additional optional arguments.
- You can enter additional parameters, values, or other information
## Convention Usage

<table>
<thead>
<tr>
<th>Convention</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>Indicates the omission of items from a code example or from a syntax line.</td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
</tbody>
</table>
Programming WebLogic Server for Wireless Services
CHAPTER

1 Introduction

The following sections provide information that you need to know before using the BEA WebLogic Server™ platform to extend Web applications to wireless subscribers:

- Overview
- Wireless Data Protocols
- The WAP Protocol at a Glance
- The i-Mode Protocol at a Glance
- Other Wireless Data Protocols
- Other Wireless Markup Languages
- Evolution of Wireless Data Protocols and Wireless Markup Languages
- Additional Information

Overview

The wireless Internet-enabled subscriber base continues to grow. Telecommunications companies and wireless service providers are looking for ways to rapidly create and deploy new revenue-generating Internet services while providing their wireless subscribers with a more personalized experience.
Because wireless subscribers have a different set of essential desires and needs than desktop or even laptop Internet users, Internet services must be designed to present the optimum experience for subscribers using different types of wireless devices. Writing Internet services in this manner requires a deep understanding of the technical issues unique to the wireless environment.

## Wireless Data Protocols

Using Internet technologies such as Hypertext Transfer Protocol (HTTP), Transport-Layer Security (TLS), Transmission Control Protocol (TCP), and Hypertext Markup Language (HTML) to set up and tear down connections and to transport Web content on a wireless network is inefficient for several reasons, including:

- HTTP and TCP are not optimized for the intermittent coverage, long latencies, and limited shared bandwidth associated with wireless networks.
- HTTP sends its headers and commands in a text format, which is not well suited for the low-bandwidth constraints of wireless networks, instead of the more efficient compressed binary format.
- HTTP session setup and teardown require many messages to be exchanged between the wireless device (client) and the server.
- HTML Web content intended for the desktop browser cannot be displayed in an effective way on the small-size screens of handheld mobile phones, personal digital assistants (PDAs), pagers, two-way radios, and smartphones.
- Navigation around and between screens is not easy due to no keyboard and no mouse; navigation is accomplished by pressing numeric keys or writing with a stylus.

Wireless services using these protocols are often slow, costly, and difficult to use, which is why special wireless data protocols were created to transport Web-based data on a wireless network. Wireless data protocols, such as Wireless Application Protocol (WAP), i-Mode, and others, are designed for the unique constraints of the wireless environment. They use binary transmission for greater compression of data and are optimized for long latency and low to medium bandwidth. They support protocol data
unit (PDU) concatenation and delayed acknowledgement to help reduce the number of messages sent. Their associated markup languages make optimum use of small screens and allow easy navigation around and between screens.

To read the markup language associated with a wireless data protocol, a wireless device requires a microbrowser, which is client software specially designed to interpret the markup language. As examples, a WAP-enabled device has a Wireless Markup Language (WML) microbrowser, and an i-Mode-enabled device has a compact HTML (cHTML) microbrowser.

The WAP Protocol at a Glance

The Wireless Application Protocol (WAP) specification consists of a wireless data protocol—a standardized way that the microbrowser on a wireless device communicates with a WAP gateway installed in the wireless network—and a markup language, Wireless Markup Language (WML). WML is an Extensible Markup Language (XML) used to specify the content and user interface for WAP-enabled devices.

WAP sessions cope with intermittent coverage and can operate over a wide variety of wireless bearer networks including but not limited to Cellular Digital Packet Data (CDPD), Code Division Multiple Access (CDMA), Global System for Mobiles (GSM), Time Division Multiple Access (TDMA), and General Packet Radio Service (GPRS). Although most WAP services in Europe and the United states are circuit-switched (dial-up), WAP will also work on packet-switched networks.

The i-Mode Protocol at a Glance

The i-Mode specification consists of a wireless data protocol—a standardized way that the microbrowser on a wireless device communicates with an i-Mode gateway installed in the wireless network—and a markup language, compact HTML (cHTML). cHTML is a subset of HTML with extensions and is used to specify the content and user interface for i-Mode-enabled devices.
Note: In publications, i-Mode appears in many different forms including *i-Mode*, *I-Mode*, *I-mode*, *i-mode*, and *imode*. *i-Mode* is used throughout this discussion and the discussions that follow.

While WAP is an open, global specification, i-Mode is currently a proprietary, closed specification developed and deployed by NTT DoCoMo of Japan. The only i-Mode implementation is NTT DoCoMo’s mobile internet access system, although several telecommunication companies in Europe and the United States have expressed an interest in i-Mode.

Although i-Mode currently operates only on NTT-DoCoMo’s PDC-P mobile voice system, i-Mode will work equally well on any underlying wireless bearer network.

## Other Wireless Data Protocols

Other wireless data protocols include:

- Handheld Device Transport Protocol (HDTP) developed by Phone.com (formerly Unwired Planet and now Openwave Systems Inc.)
- OmniSky developed by the OmniSky Corporation
- Mobitex developed by the Mobitex Operators Association

The wireless industry is also very interested in the Voice over IP (VoIP) technology, which moves voice and data traffic over a common IP infrastructure. The wireless industry is looking for ways to converge the VoIP and wireless data protocol technologies so that voice and Web data can be carried on the same over-the-air channel.

## Other Wireless Markup Languages

Other wireless markup languages include:
Evolution of Wireless Data Protocols and Wireless Markup Languages

- Handheld Device Markup Language (HDML) developed by Phone.com (formerly Unwired Planet and now Openwave Systems Inc.)
- Web Clipping developed by Palm, Inc.
- Extensible HTML (XHTML) (Basic) developed by the World Wide Web Consortium (W3C)
- VoiceXML developed by the VoiceXML Forum, an industry organization founded by AT&T, IBM, Lucent, and Motorola

Note: HDTP carries Web content tagged with HDML. HDTP and HDML heavily influenced the development of WAP and WML.

Evolution of Wireless Data Protocols and Wireless Markup Languages

The various wireless data protocols and wireless markup languages in existence today may very well evolve into a single wireless data protocol and wireless markup language tomorrow. As evidence, proponents of the WAP protocol are about to migrate to a new generation of the protocol known as WAP-NG, and both AT&T and NTT DoCoMo have made allegiances to WAP-NG. In addition, proponents of WAP and i-Mode are talking about migrating to the common markup language XHTML (Basic).

Additional Information

Here are some Web sites (organized by category) to visit for additional information about wireless data protocols, wireless markup languages, manufacturers of wireless devices, and wireless standards and specifications.
1 Introduction

Related WebLogic Wireless Information


BEA Wireless Newsgroup at news://newsgroups.bea.com/weblogic.developer.interest.wap

BEA Wireless FAQ at http://e-docs.bea.com/wls/docs81b/faq/wireless.html

General Wireless Information

The Wireless FAQ at http://allnetdevices.com/faq
FierceWireless at http://www.fiercewireless.com
MBizCentral at http://www.mbizcentral.com
Unstrung at http://www.unstrung.com
WirelessDevNet at http://www.wirelessdevnet.com

Wireless-Device Manufacturers

OmniSky at http://www.omnisky.com
Palm at http://www.palm.com/wireless
RIM BlackBerry at http://www.rim.net/products/handhelds/index.shtml
Symbian Limited at http://www.symbian.com

Wireless Standards and Specifications

XHTML Basic W3C Recommendation at http://www.w3.org/TR/xhtml-basic
The Short Message Peer to Peer (SMPP) Forum at http://www.smpp.org/index.html

1-6 Programming WebLogic Server for Wireless Services
Additional Information

SMS Protocol - UCP/EMI Specification at
http://www.cmgtelecom.com/asp/index.asp?id=100

SNPP pagers at http://www.faqs.org/rfcs/rfc1861.html
1 Introduction
CHAPTER 2

Using WAP with WebLogic Server

The following sections describe how to provide content suitable for the Wireless Application Protocol (WAP) application environment and how to configure and use WebLogic Server with a WAP gateway:

- Overview
- WAP Application Environment
- WAP Gateway
- Additional Resources

Overview

Wireless Application Protocol (WAP) is an open, global specification, developed and deployed by the WAP Forum, that allows for the development of Internet and Web-based services for mobile phones and other wireless digital devices. Its founder members include the major wireless vendors of Ericsson, Motorola, Nokia, and Phone.com (formerly Unwired Planet and now Openwave Systems Inc).

The WAP specification addresses the limitations of wireless networks (low bandwidth, high latency, and unpredictable availability and stability) and wireless devices (limited CPU, memory, and battery life, and a simple user interface). It specifies two essential elements of wireless communication: an over-the-air wireless protocol and an application environment.
WAP gateways form the connection between clients on the wireless network and applications hosted on application servers on the Internet. The WAP gateway builds a bridge between the telecommunication and computer networks by routing requests from wireless clients to the application servers. It can be physically located in either network, though it is needed in only one of them.

**WAP Application Environment**

The WAP application environment defines the framework for network-neutral, wireless applications for narrowband devices. Two of the main components of the WAP application environment are Wireless Markup Language (WML) and WMLScript.

**WML**

WML for WAP applications is analogous to HTML for TCP/IP applications. It is an XML-based language that is specifically designed to interface with the microbrowsers that exist in WAP-enabled devices. The Wireless Markup Language Specification at http://www.wapforum.org/what/technical.htm defines the tags and structure of a WML document.

A WML document is a collection—referred to as a deck—of one or more cards. Each card in a deck of cards is considered a well defined unit of interaction. The general rule of thumb is that a card carries enough information to fit in one screen of a wireless device. For information on ways to serve WML documents to wireless clients, see “WAP Gateway” on page 2-3. For general information on WML, see “Additional Resources” on page 2-5.
WMLScript

WMLScript provides general scripting capability to the WAP architecture. It is designed to overcome the limitations of narrowband communication and wireless clients. For example, WMLScript is a good way to validate form input without making a round trip to the server.

WMLScript resides in .wmls files that are made available to wireless clients by placing them into the document root. The document root is the root directory for files that are publicly available on WebLogic Server. For more detail, see the information on directory structures in Web Applications Basics at http://e-docs.bea.com/wls/docs81b/webapp/basics.html. For general information on WMLScript, see “Additional Resources” on page 2-5.

WAP Gateway

As shown in the following figure, the WAP gateway acts as the bridge between the wireless network containing wireless clients and the computer network containing application servers.

Figure 2-1 WAP Application Architecture
WAP Gateway Functionality

A WAP gateway typically includes the following functionality:

- Protocol gateway—the protocol gateway translates requests from the WAP protocol stack to the WWW protocol stack (HTTP and TCP/IP).

- Content encoders and decoders—the content encoders translate Web content into compact encoded formats to reduce the number and size of packets traveling over the wireless data network.

When a wireless client sends a request to a WAP application running on WebLogic Server, the request is first routed through the WAP gateway where it is decoded, translated to HTTP, then forwarded to the appropriate URL. The response is then re-routed back through the gateway, translated to WAP, encoded, and forward to the wireless client. This proxy architecture allows application developers to build services that are network and terminal independent.

WAP Gateway Security and Security Concerns

The security layer of the WAP protocol stack is called Wireless Transport Layer Security (WTLS). WTLS is based upon the established Transport-Layer Security (TLS) protocol standard.

For a secure connection employing the WAP protocol, a very small security risk exists at the WAP gateway during the switching of WTLS (WAP side) to SSL (IP side) and SSL to WTLS. Since the WAP protocol allows a session to be redirected from the carrier’s gateway to the enterprise’s gateway, an enterprise may want to control this minimal risk by including a WAP gateway behind its firewall. As shown in the following figure, the enterprise secures the server running the WAP gateway in a controlled environment to eliminate any exposure to the security risk.
Since the carrier is a trusted entity and is continuously responsible for protecting voice, fax, computer and other types of data, enterprises probably do not need to host their own WAP gateway.

**WAP Gateway Vendors**

There are a growing number of vendors that provide WAP gateways. WebLogic Server should work with any WAP-compliant gateway. For a list of WAP-compliant gateways and other WAP products, refer to the WAP Deployment Fact Sheet at http://www.wapforum.org/new/index.htm compiled by the WAP Forum.

**Additional Resources**

Here are some Web sites (organized by category) to visit for additional information about WebLogic Server programming, WAP, and WML.

**Related WebLogic Technologies**

Programming WebLogic JSP at http://e-docs.bea.com/wls/docs81b/jsp/index.html
Programming WebLogic HTTP Servlets at
http://e-docs.bea.com/wls/docs81b/servlet/index.html

Programming WebLogic XML at
http://e-docs.bea.com/wls/docs81b/xml/index.html

Web Applications Basics at
http://e-docs.bea.com/wls/docs81b/webapp/basics.html

General WAP Information

WAP Forum at http://www.wapforum.org

Ericsson: Developers’ Zone at http://www.ericsson.com/developerszone

Motorola at http://developers.motorola.com/developers/wireless

Nokia: WAP Solutions for Mobile Business at

Openwave Developer Resources at

WAP Specifications and White Papers

WAP specifications at http://www.wapforum.org/what/technical.htm

WAP white papers at http://www.wapforum.org/what/whitepapers.htm

WAP Toolkits


Motorola Tools and Downloads at
http://developers.motorola.com/developers/wireless/tools

Openwave Software Development Kit at

Ericsson Developers’ Zone at http://www.ericsson.com/developerszone
CHAPTER

3 Using i-Mode with WebLogic Server

The following sections describe how to provide content suitable for the i-Mode application environment and how to configure and use WebLogic Server with an i-Mode gateway:

- Overview
- i-Mode Markup Language
- i-Mode Gateway
- Additional Resources

Overview

i-Mode is currently a proprietary, closed specification, developed and deployed by NTT DoCoMo of Japan, that allows for the development of Internet and Web-based services for mobile phones and other wireless digital devices. The only i-Mode implementation is DoCoMo’s mobile internet access system, although several telecommunication companies in Europe and the United States have expressed an interest in i-Mode.

Note: i-Mode is a trademark and/or service mark owned by NTT DoCoMo.
Using i-Mode with WebLogic Server

The i-Mode specification addresses the limitations of wireless networks (low bandwidth, high latency, and unpredictable availability and stability) and wireless devices (limited CPU, memory, and battery life, and a simple user interface). It specifies two essential elements of wireless communication: an over-the-air wireless protocol and a markup language.

i-Mode gateways form the connection between clients on the wireless network and applications hosted on application servers on the Internet. The i-Mode gateway builds a bridge between the telecommunication and computer networks by routing requests from wireless clients to the application servers.

i-Mode Markup Language

i-Mode applications use the compact Hypertext Markup Language (cHTML), which is specifically designed to interface with the microbrowsers in i-Mode-enabled devices. cHTML is a subset of HTML, with a number of extensions for the mobile phone environment. The NTT DoCoMo (English) Supported Tags and Specs at http://www.nttdocomo.co.jp/english/p_s/imode/ defines the tags and structure of a cHTML document.

i-Mode Gateway

As shown in the following figure, the i-Mode gateway acts as the bridge between the wireless network containing wireless clients and the computer network containing application servers.

Figure 3-1  i-Mode Application Architecture
An i-Mode gateway typically includes a protocol gateway, which translates requests from the i-Mode protocol stack to the WWW protocol stack (HTTP and TCP/IP).

When a wireless client sends a request to an i-Mode application running on WebLogic Server, the request is first routed through the i-Mode gateway and then forwarded to the appropriate URL. The response is then re-routed back through the gateway, translated to i-Mode, and forward to the wireless client. This proxy architecture allows application developers to build services that are network and terminal independent.

**Additional Resources**

Here are some Web sites (organized by category) to visit for additional information about WebLogic Server programming, i-Mode, and cHTML.

**Related WebLogic Technologies**

- Programming WebLogic JSP at http://e-docs.bea.com/wls/docs81b/jsp/index.html
- Programming WebLogic HTTP Servlets at http://e-docs.bea.com/wls/docs81b/servlet/index.html
- Programming WebLogic XML at http://e-docs.bea.com/wls/docs81b/xml/index.html
Using i-Mode with WebLogic Server

Web Applications Basics at
http://e-docs.bea.com/wls/docs81b/webapp/basics.html