

IBM Content Manager OnDemand for Multiplatforms



# Web Enablement Kit Implementation Guide

*Version 7.1*



IBM Content Manager OnDemand for Multiplatforms



# Web Enablement Kit Implementation Guide

*Version 7.1*

**Note**

Before using this information and the product it supports, read the information in Appendix M, "Notices", on page 209.

**Third Edition (April 2003)**

This edition replaces and obsoletes *IBM Content Manager OnDemand for Multiplatforms Version 7.1 Web Enablement Kit Installation and Configuration Guide*, SC27-1000-01. This edition applies to IBM Content Manager OnDemand for Multiplatforms Version 7 Release 1 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 1996, 2003. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>About this publication . . . . .</b>	<b>vii</b>
How this publication is organized . . . . .	vii
Who should use this publication . . . . .	viii
What you should already know . . . . .	viii
Where to find more information . . . . .	ix
Product support . . . . .	ix
How to send your comments . . . . .	x
<b>  Summary of changes . . . . .</b>	<b>xi</b>
<b>Chapter 1. Product overview . . . . .</b>	<b>1</b>
About the programming interfaces . . . . .	3
About the viewers . . . . .	5
Using ODWEK . . . . .	6
Product functions . . . . .	7
Add Annotation . . . . .	7
Change Password . . . . .	7
Document Hit List . . . . .	8
Logoff . . . . .	8
Logon . . . . .	8
Retrieve Document . . . . .	8
Search Criteria . . . . .	8
Server Print Document . . . . .	8
Update Document . . . . .	8
View Annotations . . . . .	9
Server and data security . . . . .	9
<b>  Chapter 2. Implementation overview . . . . .</b>	<b>11</b>
Possible HTTP server and Web application server scenarios . . . . .	11
Implementation steps . . . . .	11
Your next step . . . . .	13
<b>  Chapter 3. Installation requirements . . . . .</b>	<b>15</b>
Installation requirements . . . . .	15
Disk space requirements . . . . .	16
Transform requirements . . . . .	17
Your next step . . . . .	18
<b>Chapter 4. Installing ODWEK . . . . .</b>	<b>19</b>
Before you begin . . . . .	19
Your next step . . . . .	19
Installing on AIX . . . . .	19
Your next step . . . . .	20
Installing on HP-UX . . . . .	20
Your next step . . . . .	20
Installing on Linux . . . . .	21
Your next step . . . . .	21
Installing on Solaris . . . . .	21
Your next step . . . . .	22
Installing on Windows servers . . . . .	22
Your next step . . . . .	22
<b>  Chapter 5. Deploying the CGI program . . . . .</b>	<b>25</b>
Before you begin . . . . .	25
Copying GGI program files . . . . .	25
Your next step . . . . .	25
<b>  Chapter 6. Deploying the Java servlet . . . . .</b>	<b>27</b>
Before you begin . . . . .	27
Copying files . . . . .	27
Your next step . . . . .	28
Deploying the servlet using WebSphere tools . . . . .	29
Assembling the application . . . . .	29
Installing the application . . . . .	31
Your next step . . . . .	33
<b>  Chapter 7. Specifying the ARSWWW.INI file . . . . .</b>	<b>35</b>
[@SRV@_DEFAULT] . . . . .	35
PORT . . . . .	35
PROTOCOL . . . . .	36
[@SRV@_server]. . . . .	36
HOST . . . . .	36
PORT . . . . .	37
PROTOCOL . . . . .	37
[CONFIGURATION] . . . . .	37
APPLETCACHEDIR . . . . .	38
APPLETDIR . . . . .	38
CACHEDIR . . . . .	39
CACHEDOCS . . . . .	39
CACHEMAXTHRESHOLD . . . . .	40
CACHEMINTHRESHOLD . . . . .	40
CACHESIZE . . . . .	40
CACHEUSERIDS . . . . .	41
CODEPAGE . . . . .	42
DOCSIZE . . . . .	42
IMAGEDIR . . . . .	42
LANGUAGE . . . . .	43
TEMPDIR . . . . .	44

TEMPLATEDIR . . . . .	45	VIEWNOTES . . . . .	66
[SECURITY] . . . . .	45	[browser] . . . . .	66
REPORTSERVERTIMEOUT . . . . .	45	[DEBUG] . . . . .	67
SERVERACCESS . . . . .	46	LOG . . . . .	68
[AFP2HTML] . . . . .	46	LOGDIR . . . . .	68
CONFIGFILE . . . . .	47	Example ARSWWW.INI file . . . . .	69
INSTALLDIR. . . . .	47	Your next step . . . . .	70
USEEXECUTABLE . . . . .	48		
[AFP2PDF] . . . . .	48	<b>Chapter 8. Configuring the sample applications</b> . . . . .	71
CONFIGFILE . . . . .	49	LOGON.HTM . . . . .	71
INSTALLDIR. . . . .	49	CREDIT.HTM . . . . .	72
USEEXECUTABLE . . . . .	49	TEMPLATE.HTM . . . . .	73
[MIMETYPES] . . . . .	50	Your next step . . . . .	74
AFP. . . . .	51		
BMP . . . . .	52	<b>Chapter 9. Installing the Web viewers</b> . . . . .	75
GIF . . . . .	52	Overview . . . . .	75
EMAIL. . . . .	52	Requirements . . . . .	76
JFIF. . . . .	53	Installation . . . . .	77
LINE . . . . .	53	Your next step . . . . .	78
PCX. . . . .	54		
PDF. . . . .	54	<b>Chapter 10. Verifying the installation</b> . . . . .	79
TIFF . . . . .	54	Verifying the CGI program . . . . .	79
[ATTACHMENT IMAGES] . . . . .	55	Verifying the servlet . . . . .	81
BMP . . . . .	55	Troubleshooting . . . . .	82
GIF . . . . .	56	Your next step . . . . .	83
TXT. . . . .	56		
[NO HTML] . . . . .	56	<b>Appendix A. CGI API reference</b> . . . . .	85
BEGIN. . . . .	56	Add Annotation . . . . .	86
END . . . . .	57	Change Password . . . . .	89
SEPARATOR. . . . .	57	Document Hit List . . . . .	91
[DEFAULT BROWSER] . . . . .	57	Logoff . . . . .	95
ADDEXTENSION . . . . .	58	Logon . . . . .	97
ADDFIELDSTODOCID . . . . .	58	Print Document (Server) . . . . .	99
ADDNOTES . . . . .	59	Retrieve Document . . . . .	103
AFPVIEWING . . . . .	59	Search Criteria . . . . .	107
AUTODOCRETRIEVAL . . . . .	60	Update Document . . . . .	110
EMAILVIEWING . . . . .	61	View Annotations . . . . .	112
ENCRYPTCOOKIES . . . . .	61		
ENCRYPTURL . . . . .	62	<b>Appendix B. Java servlet reference</b> . . . . .	115
FOLDERDESC . . . . .	62		
LINEVIEWING . . . . .	62	<b>Appendix C. Java API reference</b> . . . . .	117
MAXHITS . . . . .	63		
NOLINKS. . . . .	64	<b>Appendix D. Java API programming guide</b> . . . . .	119
ODApplet.jre.path.IE . . . . .	64	Client/server architecture . . . . .	119
ODApplet.jre.path.NN . . . . .	64	Packaging for the Java environment . . . . .	119
ODApplet.jre.version . . . . .	64	Programming tips . . . . .	120
ODApplet.version . . . . .	64	Setting up the system environment . . . . .	121
SERVERPRINT . . . . .	64	Setting environment variables . . . . .	121
SERVERPRINTERS. . . . .	65		
SHOWDOCLOCATION . . . . .	65		

Tracing and diagnostic information . . . . .	122	Example of a script file . . . . .	181
Tracing . . . . .	122	Changes to the Retrieve Document API . . . . .	183
Exception handling . . . . .	123		
Constants . . . . .	124	<b>Appendix G. AFP to HTML transform . . . . .</b>	<b>185</b>
Running an ODWEK application. . . . .	124	Format of the AFP2HTML.INI file . . . . .	186
Connecting to an OnDemand server . . . . .	124	Options for the AFP2WEB Transform . . . . .	187
Establishing a connection . . . . .	125	Viewing converted documents . . . . .	188
Setting and getting passwords . . . . .	125		
Working with an OnDemand server . . . . .	125		
Listing application groups in a folder . . . . .	128	<b>Appendix H. AFP to PDF transform . . . . .</b>	<b>189</b>
Searching a folder. . . . .	130	Specifying the AFP2PDF.INI file . . . . .	189
Cancelling a search . . . . .	135	Viewing converted documents . . . . .	191
Listing search criteria . . . . .	137		
Listing folders and folder information . . . . .	141	<b>Appendix I. Distributing user-defined files . . . . .</b>	<b>193</b>
Displaying a list of documents . . . . .	143	Installing the AFP Web Viewer files . . . . .	194
Retrieving a document . . . . .	145	Adding subdirectories . . . . .	195
Printing a document . . . . .	148	Storing user-defined files . . . . .	196
Listing information about notes . . . . .	151	Configuring font files . . . . .	196
Adding a note . . . . .	153	Building the AFP Web Viewer installation	
Updating a document . . . . .	155	file. . . . .	197
Changing a password . . . . .	158	Installing the AFP Web Viewer on a user's	
		workstation. . . . .	198
<b>Appendix E. Java line data viewer . . . . .</b>	<b>161</b>		
<b>Appendix F. Xenos transforms. . . . .</b>	<b>167</b>	<b>Appendix J. Mapping AFP fonts . . . . .</b>	<b>199</b>
Converting data streams . . . . .	168		
AFP documents to HTML . . . . .	168	<b>Appendix K. No HTML output . . . . .</b>	<b>201</b>
AFP data to PDF . . . . .	168	Delimited ASCII output. . . . .	201
AFP documents to XML . . . . .	169	Logon. . . . .	202
Metacode to AFP . . . . .	169	Notes . . . . .	202
Metacode documents to HTML . . . . .	169	Search Criteria. . . . .	202
Metacode to PDF . . . . .	169	Notes . . . . .	202
Metacode documents to XML. . . . .	170	Document Hit List . . . . .	203
Viewing and printing with the Web . . . . .	170	Notes . . . . .	204
Enablement Kit . . . . .	170	View Annotations. . . . .	204
Configuring the ARSWWW.INI file . . . . .	173	Error Message . . . . .	205
[XENOS]. . . . .	173	Notes . . . . .	205
Specifying the AFP transform option . . . . .	174		
Specifying the Metacode transform option .	175		
Configuring the ARSXENOS.INI file . . . . .	176	<b>Appendix L. Problem determination tools . . . . .</b>	<b>207</b>
Specifying the ARSXENOS.INI file . . . . .	176		
Viewing converted documents . . . . .	178	<b>Appendix M. Notices . . . . .</b>	<b>209</b>
Example of a parameter file . . . . .	178	Trademarks and service marks . . . . .	211
		<b>Index . . . . .</b>	<b>213</b>



---

# About this publication

This publication provides information that you can use to plan for, install, configure, and use the IBM® Content Manager OnDemand for Multiplatforms Version 7.1 (OnDemand) Web Enablement Kit.

---

## How this publication is organized

This publication provides the information that you need to install and configure the OnDemand Web Enablement Kit (ODWEK) and to configure the Web server software so that users can access data from an OnDemand server with a Web browser. This publication contains the following sections:

- Chapter 1, “Product overview”, on page 1 provides general information about the ODWEK system, programming interfaces, viewers, and functions and server and data security.
- Chapter 2, “Implementation overview”, on page 11 provides important information about the installation and configuration process and contains an implementation checklist.
- Chapter 3, “Installation requirements”, on page 15 lists the software requirements, disk space requirements, and transform requirements.
- Chapter 4, “Installing ODWEK”, on page 19 describes how to install the ODWEK software on the system.
- Chapter 5, “Deploying the CGI program”, on page 25 describes how to deploy the CGI version of ODWEK on the system.
- Chapter 6, “Deploying the Java servlet”, on page 27 describes how to deploy the Java™ servlet on the system.
- Chapter 7, “Specifying the ARSWWW.INI file”, on page 35 contains detailed information about the ODWEK configuration file.
- Chapter 8, “Configuring the sample applications”, on page 71 explains how to customize the sample applications that are provided with ODWEK.
- Chapter 9, “Installing the Web viewers”, on page 75 describes the viewers, lists the requirements for the viewers, and describes how to install the viewers.
- Chapter 10, “Verifying the installation”, on page 79 explains how to verify the installation, contains troubleshooting help, and provides a road map to information that you may need after you have finished installing ODWEK.
- Appendix A, “CGI API reference”, on page 85 contains reference information for programming with the CGI version of ODWEK.
- Appendix B, “Java servlet reference”, on page 115 contains reference information for programming with the Java servlet.

- Appendix C, “Java API reference”, on page 117 explains where to locate reference information for programming with the Java API.
- Appendix D, “Java API programming guide”, on page 119 describes how to set up the system environment and run Java on an ODWEK application. This section is also a guide to programming with the Java API and contains examples.
- Appendix E, “Java line data viewer”, on page 161 describes how to configure the system to use the Java line data viewer.
- Appendix F, “Xenos transforms”, on page 167 provides information about the optional Xenos transform. Note: In this publication, the term Xenos transform refers to the Xenos d2e Platform. Xenos d2e Platform is a trademark of Xenos Group Inc.
- Appendix G, “AFP to HTML transform”, on page 185 provides information about the optional AFP<sup>TM</sup> to HTML transform.
- Appendix H, “AFP to PDF transform”, on page 189 provides information about the optional AFP to PDF transform.
- Appendix I, “Distributing user-defined files”, on page 193 describes how to include user-defined files (such as fonts and mapping files) in the AFP Web Viewer installation file and distribute them to your users.
- Appendix J, “Mapping AFP fonts”, on page 199 explains where to find information about mapping AFP fonts for use with the AFP Web Viewer.
- Appendix K, “No HTML output”, on page 201 describes the delimited ASCII output that can be generated by ODWEK.
- Appendix L, “Problem determination tools”, on page 207 lists the tools that you can use to gather information about the system and documents to aid in troubleshooting.

---

## Who should use this publication

This publication is for administrators that need to configure ODWEK for an organization. Application programmers who need to create Internet applications that access OnDemand servers should read Chapter 8, “Configuring the sample applications”, on page 71, Appendix A, “CGI API reference”, on page 85, Appendix B, “Java servlet reference”, on page 115, Appendix C, “Java API reference”, on page 117, Appendix D, “Java API programming guide”, on page 119, and Appendix K, “No HTML output”, on page 201.

---

## What you should already know

The documentation for ODWEK assumes that you understand the Internet, Web servers and browsers, Transmission Control Protocol/Internet Protocol (TCP/IP) networking, and OnDemand. This book assumes that you are familiar with Hypertext Markup Language (HTML), Common Gateway

Interface (CGI) and Java programming, that you provide content for Web pages, that know how to configure and operate an Hypertext Transfer Protocol (HTTP) server, a Java-enabled Web server, and a Java application server, and that you can administer an OnDemand server.

If you plan to retrieve AFP documents or Metacode/DJDE documents from the server and process them with the Xenos transforms, see Appendix F, “Xenos transforms”, on page 167. Contact your IBM representative for information about how to obtain the Xenos transform programs, license, and documentation. Your IBM representative can also provide information about education and other types of help and support for installing and configuring the transform programs and processing input files with the transform programs.

If you plan to use the Java AFP2HTML Viewer, you must obtain the AFP2WEB Transform service offering from IBM and install and configure it on the Web server. See your IBM representative for more information about the AFP2WEB Transform service offering. You must also provide configuration options for the Advanced Function Presentation<sup>TM</sup> (AFP) documents and resources that you plan to process with the AFP2WEB Transform. See Appendix G, “AFP to HTML transform”, on page 185 for more information about the configuration file.

If you plan to convert AFP documents retrieved from OnDemand into PDF documents that can be viewed with the Adobe Acrobat viewer, then you must obtain the AFP2PDF Transform service offering from IBM and install and configure it on the Web server. See your IBM representative for more information about the AFP2PDF Transform service offering. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2PDF Transform. See Appendix H, “AFP to PDF transform”, on page 189 for more information about the configuration file.

---

## Where to find more information

Product documentation is available on the Web. Click **Library** from the product Web site at:

<http://www.ibm.com/software/data/ondemand/mp>

---

## Product support

Appendix L, “Problem determination tools”, on page 207 describes some of the ways that you can solve problems or get more information when you have problems using or running ODWEK.

Product support is available on the Web. Click **Support** from the product Web site at:

## How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this publication or other OnDemand documentation. You can use either of the following methods to provide comments:

- Send your comments from the Web. Visit the IBM Data Management Online Reader's Comment Form (RCF) page at:  
<http://www.ibm.com/software/data/rcf>
- Send your comments by e-mail to:  
ondemand@us.ibm.com

Be sure to include the name of the product, the version number of the product, and the name of the book. If you are commenting on specific text, please include the location of the text (for example, a chapter and section title, a table number, a page number, or a help topic title).

## Summary of changes

This edition of *IBM Content Manager OnDemand for Multiplatforms Version 7.1: Web Enablement Kit Installation and Configuration Guide* contains significant new technical information and editorial changes. There may be some instances where changes were made, but change bars are missing. Significant changes to note are:

The AFP plug-in was renamed to the AFP Web Viewer. The Image plug-in was renamed to the Image Web Viewer. In addition, when using Internet Explorer 5.5 with Service Pack 2 or later or Internet Explorer 6, the viewers will be installed as ActiveX controls rather than plug-ins. The installation section for the viewers was updated with hardware and software requirements and other installation information.

Documented the limitations when using ODWEK to access a Content Manager OnDemand for OS/390® V2.1 server. These include Add Annotation, Update Document, and View Annotation functions and the ADDNOTES, CACHEUSERIDS, SHOWDOCLOCATION and VIEWNOTES parameters in the ARSWWW.INI file.

A new section titled Implementation Overview was added.

Added an important note to point out that the CGI, Java API and Java servlet are mutually exclusive and cannot be mixed in a given instance or application of ODWEK.

Updated hardware and software requirements.

Updated the section titled Deploying the CGI Program.

Added a section titled Deploying the Java API.

Updated the section titled Deploying the Servlet.

Changes to the ARSWWW.INI file, including

- New and updated parameters, including: PORT, PROTOCOL, CACHEDIR, DOCSIZE, TEMPDIR, REPORTSERVERTIMEOUT, APPLETCACHEDIR, AUTODOCRETRIEVAL, ENCRYPTCOOKIES, ENCRYPTURL, FOLDERDESC, and SERVERPRINTERS
- The MIMETYPES section was updated with a list of the MIME content types for typical PC applications.

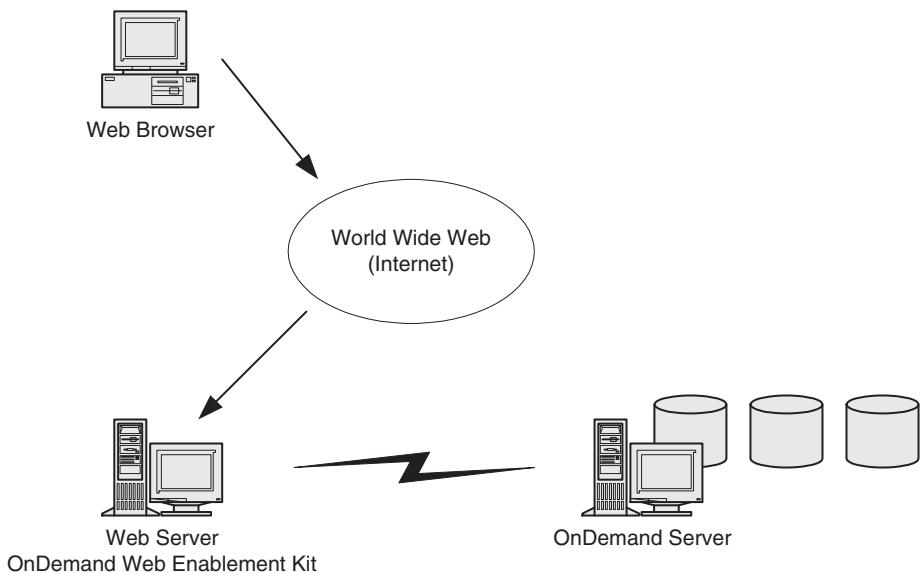
- | Added a section on Verifying the Installation.
- | Updates to all of the APIs in the CGI API Reference section, including adding the \_logoff parameter to the Add Annotation, Change Password, Document Hit List, Print Document, Retrieve Document, Update Document, and View Annotations functions and updating the function of the \_sort\_field and \_sort\_order parameters in the Document Hit List function.
- | Updates to the Search Criteria and Document Hit List parts of the No HTML Output section.
- | Added the ALLOBJECTS parameter to the AFP2HTML.INI file.
- | Added a section titled Java API Reference.
- | Added a section titled Java API Programming Guide.
- | Added a section titled Java Line Data Viewer.
- | Added a section titled Java Servlet Reference.
- | A new section titled Xenos Transforms was added, including a sample parameter file and a sample script file.

---

## Chapter 1. Product overview

ODWEK allows users to access data that is stored in an IBM Content Manager OnDemand server by using a Web browser. For example, you can provide some users with the Uniform Resource Locator (URL) of a Web page that permits them to log on to an OnDemand server; you can provide other users with the URL of a Web page that permits them to search a specific folder. ODWEK verifies that the user information is valid on the OnDemand server, such as permission to access the server and data stored in an application group. After the user submits a search, ODWEK displays a Web page that contains a list of the documents that match the query. The user selects a document to view and ODWEK sends the document to the browser.

Figure 1 shows a Web browser that is being used to access data from an OnDemand server.



*Figure 1. Accessing data stored in OnDemand using ODWEK.*

ODWEK can search for and retrieve documents from OnDemand servers that are running IBM Content Manager OnDemand for iSeries<sup>TM</sup> Common Server Version 5 Release 2, IBM Content Manager OnDemand for Multiplatforms Version 7.1, and IBM Content Manager OnDemand for z/OS<sup>TM</sup> and OS/390, Version 7.1.

**Note:** In IBM Content Manager OnDemand for z/OS and OS/390, Version 7.1, IBM provided a migration (or coexistence) strategy that allowed all of the OnDemand Version 7.1 clients (Windows®, CICS®, ODWEK) to access OnDemand Version 7.1 servers (all platforms) and OnDemand Version 2.1 servers (all platforms). ODWEK Version 7.1.0.6 or later (all platforms) also supports accessing an OnDemand for OS/390 Version 2.1 server directly. However, because of differences in the OnDemand Version 2.1 architecture and the OnDemand Version 7.1 architecture, there are some functions in ODWEK Version 7.1.0.6 that are not currently supported when accessing an OnDemand for OS/390 Version 2.1 server. The known functions that are not supported are:

- The CGI and servlet do not support the caching of userids. The CACHEUSERIDS parameter in the ARSWWW.INI configuration file has no effect in this environment.
- The CGI, servlet, and Java API do not support display or retrieval of the document location. The SHOWDOCLOCATION parameter in the ARSWWW.INI configuration file has no effect in this environment.
- The CGI, servlet, and Java API do not support the Add Annotation function. The ADDNOTES parameter in the ARSWWW.INI configuration file has no effect in this environment.
- The CGI, servlet, and Java API do not support the View Annotations function. The VIEWNOTES parameter in the ARSWWW.INI configuration file has no effect in this environment.
- The CGI, Servlet, and Java API do not support the Update Document function.

ODWEK contains several components:

- OnDemand programming interface. The programming interface uses standard OnDemand interfaces and protocols to access data stored in an OnDemand server. No additional code is needed on the OnDemand server to support ODWEK. You may use one of the following OnDemand programming interfaces in your ODWEK application:
  - CGI program. The CGI program runs on a system that is running an HTTP server, such as the IBM HTTP Server.
  - Java servlet. The servlet runs on a system that is running a Java-enabled HTTP server with a Java application server, such as the IBM WebSphere® Application Server. The servlet requires Java version 1.2.2 or later. Customers that plan to use Java version 1.3.1 to support the Java servlet must install Java version 1.3.1 with Fix Pack 4 or later.
  - Java Application Programming Interface (API). The Java API requires Java version 1.2.2 or later.
- IBM OnDemand AFP Web Viewer. The AFP Web Viewer lets users search, retrieve, view, navigate, and print AFP documents from a Web browser.

- IBM OnDemand Image Web Viewer. The Image Web Viewer lets users search, retrieve, view, navigate, and print BMP, GIF, JPEG, PCX, and TIFF documents from a Web browser.
- Java Line Data Viewer. The Java Line Data Viewer lets users view line data documents from a Web browser. IBM now provides two versions of the Java Line Data Viewer. See Chapter 9, “Installing the Web viewers”, on page 75 and Appendix E, “Java line data viewer”, on page 161 for information about the Java Line Data Viewer.
- Java AFP2HTML Viewer. The Java AFP2HTML Viewer lets users view the output generated by the IBM AFP2WEB Transform service offering. The AFP2WEB Transform converts AFP documents and resources into HTML files that can be displayed with the Java AFP2HTML Viewer. After installing and configuring the AFP2WEB Transform, an administrator enables the use of the Java AFP2HTML Viewer by configuring the ARSWWW.INI file.

**Note:** To view other types of documents stored in OnDemand, you must obtain and install the appropriate viewer. For example, to view Adobe Portable Data Format (PDF) documents, IBM recommends that you obtain the Adobe Acrobat viewer for the browsers that are used in your organization.

---

## About the programming interfaces

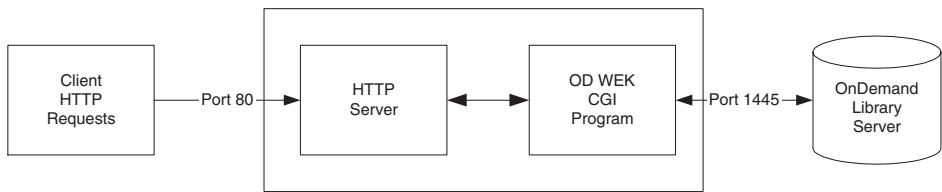
An *instance* of ODWEK (sometimes called an *application*) is ODWEK code that accesses data on an OnDemand server. An instance controls what can be done to the data, and manages the system resources that are assigned to it. Each instance is a complete environment. An instance has its own ASWWW.INI file and ODWEK programming interface, which other instances cannot access. There are three ODWEK programming interfaces:

- CGI program
- Java servlet
- Java API

It is very important to understand that an instance may use only one programming interface. The programming interfaces are mutually exclusive. They cannot be used in the same instance at the same time. However, it is possible to run multiple instances of ODWEK on a single machine and have each instance use a different programming interface by configuring each instance to use a different port number.

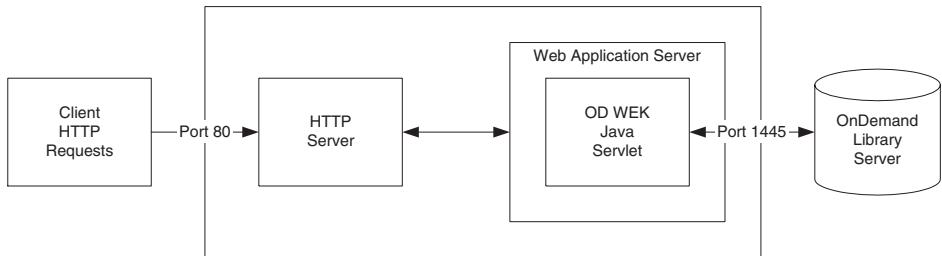
The most common implementation of ODWEK is a single instance on a system. The single instance configuration is typically for developer desktops or standalone production computing, which involve a single application server instance operating independently of any other applications.

Figure 2 shows an example of a single instance using the CGI interface.



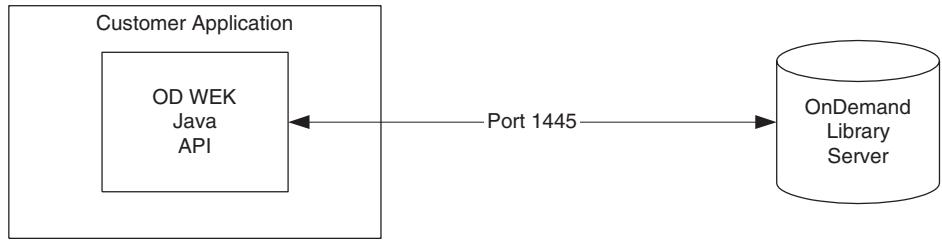
*Figure 2. Single instance using CGI interface*

Figure 3 shows an example of a single instance using the Java servlet interface.



*Figure 3. Single instance using Java interface*

Figure 4 shows an example of a single instance using the Java API interface.



*Figure 4. Single instance using Java API interface*

You can configure multiple instances of ODWEK on the same system. Each instance requires its own programming interface and ARSWWW.INI file, which specifies the unique port number over which communications between the programming interface and the OnDemand server take place. Each instance also requires its own storage and security. The multiple instance configuration is typically for customers that need to run one or more

developer, testing or production applications on the same system. The instances operate independently of each other.

Figure 5 shows an example of the multiple instance topology.

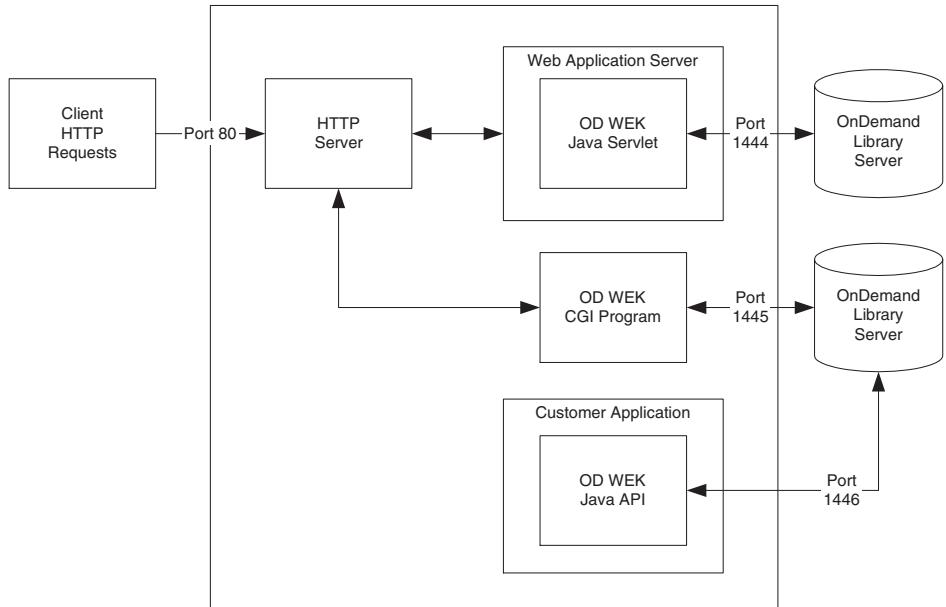


Figure 5. Multiple instance topology

## About the viewers

ODWEK provides the following viewers:

- AFP Web Viewer
- Image Web Viewer
- Java Line Data Viewer
- Java AFP2HTML Viewer

The AFP Web Viewer and the Image Web Viewer are software programs that extend the capabilities of a Web browser in a specific way. The AFP Web Viewer lets users view AFP documents. The Image Viewer lets users view BMP, GIF, JPEG, PCX, and TIFF documents. The viewers provide the capability to display documents in the browser window. Each viewer adds a toolbar to the top of the display window. The viewer toolbar may be in addition to the browser's toolbar. The viewer toolbar provides controls that can help users work with documents. Each user who plans to use the Web viewers to view documents must install them on their PC.

**Note:** The installation program will install the viewers as either plug-ins or ActiveX controls. If Internet Explorer is installed on the PC, then the installation program will install the ActiveX controls; if Netscape is installed on the PC, then the installation program will install the plug-ins. If both Internet Explorer and Netscape are installed on the workstation, then the installation program will install the ActiveX controls for Internet Explorer and the plug-ins for Netscape.

The Java Line Data Viewer is an applet that lets users view line data documents that are stored in OnDemand. The Java Line Data Viewer displays line data documents in the browser window and adds a toolbar to the top of the display window. The Java Line Data Viewer toolbar provides controls that can help users work with documents. An administrator enables the use of the Java Line Data Viewer by configuring the ARSWWW.INI file.

The Java AFP2HTML Viewer is an applet that lets users view the output generated by the IBM AFP2WEB Transform service offering. The AFP2WEB Transform converts AFP documents and resources into HTML documents. After installing and configuring the AFP2WEB Transform, an administrator enables the use of the Java AFP2HTML Viewer by configuring the ARSWWW.INI file. The Java AFP2HTML Viewer provides a toolbar with controls that can help users work with documents, including controls for large objects.

One advantage of the applets is that users never have to install or upgrade software on the PC to use them, unlike the AFP Web Viewer and the Image Web Viewer, which must be installed on the PC. Also, if IBM provides a new version of the AFP Web Viewer or the Image Web Viewer, you must distribute the updated software to all users.

When using the applets and viewers that are provided by IBM, the documents that are retrieved from an OnDemand server remain compressed until reaching the viewer. The viewer uncompresses the documents and displays the pages in a Web browser window. If a document was stored in OnDemand as a large object, then the viewer retrieves and uncompresses segments of the document, as needed, when the user moves through pages of the document.

---

## Using ODWEK

The most common method of using ODWEK is by customizing the sample HTML applications that are provided by IBM. The LOGON.HTM sample application supports users that are permitted access to several folders. To use the sample application, first modify the LOGON.HTM page with information about your OnDemand server. Then publish the URL of the LOGON.HTM file. Your users can then link to the URL and log on to the specified server. ODWEK automatically displays a series of Web pages for users to search for,

retrieve, and display OnDemand documents. The CREDIT.HTM sample application supports casual use of OnDemand by providing a Web page that contains search criteria for a specific folder. After you customize the sample, the user links to the URL, completes the search criteria, and presses the Submit button. ODWEK displays a Web page that lists the documents that match the query.

**Important:** ODWEK requires the ability to write cookie data on the PC. Users must configure their browsers to accept cookies.

Most customers define one OnDemand userid to access a server with ODWEK. This is common in environments with many casual users of OnDemand who will be accessing the same folder. You can also provide each user with their own OnDemand userid. Regardless of how you decide to access OnDemand with ODWEK, you must manage the userids in OnDemand: you must add them to the server and set application group and folder permissions for the users.

---

## Product functions

The following OnDemand functions are supported by ODWEK. You typically invoke the functions by creating Web pages that contain links to the ODWEK server program. Each link invokes a specific function. The output of one function is another Web page with links that lead the user to the next logical function. For example, the initial Web page may invoke the Logon function. The Logon function generates a Web page with a link to the Search Criteria function. Each function can be called with an API. See Appendix A, “CGI API reference”, on page 85 for details.

### Add Annotation

**Note:** The Add Annotation function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

The Add Annotation function enables users to add an annotation to the specified document. To add an annotation, the user must be given the Add permission under Annotation in the OnDemand application group. (A user is automatically given the Add permission when they are given permission to access an application group.)

### Change Password

The Change Password function allows users to change their OnDemand passwords.

## **Document Hit List**

The Document Hit List function builds the list of items that match the search criteria. The list is presented in an HTML table. Each item that matches the search is stored in a table cell and contains a link to the Retrieve Document function.

## **Logoff**

The Logoff function allows users to log off an OnDemand server.

## **Logon**

The Logon function allows the users to logon to an OnDemand server. If the Logon function is successful, the user is presented with a Web page that contains the list of folders that the user is authorized to open.

## **Retrieve Document**

The Retrieve Document function retrieves a document from OnDemand. The data stream returned from the server includes the document, and depending on the data type, the resources required to view the document. The data stream must not be modified in any way. The browser, along with the viewer, interpret and decode the data stream and display the document. If the document is stored in OnDemand as a large object, then only the first segment of the document is returned. Subsequent segments of the document are retrieved and displayed as needed.

## **Search Criteria**

After a successful logon, the user is presented with the list of folders that the user is authorized to open. The user selects a folder to open. Upon opening a folder, a Web page is displayed that contains the search fields for the folder. The user can accept the default search criteria or enter search criteria to search for specific documents. When the user presses the Submit button, the search request is sent to the OnDemand server.

## **Server Print Document**

The Server Print Document function sends copies of documents to an OnDemand server printer. To use server print, the user must be given the Print permission under Document in the OnDemand application group. (A user is automatically given the Print permission when they are given permission to access an application group.) At least one server printer must be defined on the OnDemand server.

## **Update Document**

**Note:** The Update Document function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

The Update Document function allows users to update the database. The Update Document function updates one or more database fields for a specific

document. To update a document, the user must be given the Update permission under Document in the OnDemand application group.

## View Annotations

**Note:** The View Annotations function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

The View Annotations function enables users to view the annotations attached to the specified document. To view annotations, the user must be given the View permission under Annotation in the OnDemand each application group. (A user is automatically given the View permission when they are given permission to access an application group.)

---

## Server and data security

There are two levels of security that you need to consider before you use ODWEK:

- Who can access the ODWEK programs and the Web pages
- Who can access data on the OnDemand server

Any user that can access your Web server and the programs and Web pages that comprise the front-end to ODWEK can potentially access data stored in OnDemand. IBM strongly encourages you to limit access to the programs and Web pages. There are many ways that you can limit access to programs and Web pages on your Web server. For example, many Web servers provide a system of security to sensitive Web pages by allowing you to restrict access to directories. You can also use a password file on the Web server, that requires users to enter a userid and password before accessing the Web pages. However, even though Web server userids and passwords are similar to operating system userids and passwords, there is no correspondence between them and operating system userids and passwords. There is also no correspondence between Web server userids and passwords and OnDemand userids and passwords.

ODWEK provides access to OnDemand servers and data using standard OnDemand APIs. The APIs verify that the OnDemand userid can access the server and the requested data. Someone in your organization must administer user and data security on the OnDemand server.

There's one other security-related detail that you might want to consider: the method used to transfer form parameters and values between the browser and the server. The forms provided with ODWEK use the POST method to transfer parameters and values within the body of the HTTP request. With the POST method, the parameters and values do not appear in the Location field of the browser. For example, a typical function call appears as follows:

`http://www.company.com/cgi-bin/arswww.cgi`

However, if you do not specify a method when you create a form, then the default method is GET, which transfers parameters and values within the URL itself. With the GET method, a typical function call appears as follows:

`http://www.company.com/cgi-bin/arswww.cgi?_function=logon  
&_user=bob&_password=secret`

The parameters and values appear as clear text in the Location field of the browser window. If you create your own forms, IBM strongly encourages you to use the POST method. To change the default method from GET to POST, you must code the METHOD attribute on the form tag.

## Chapter 2. Implementation overview

This section provides an overview of the installation and configuration process:

- "Possible HTTP server and Web application server scenarios".
- "Implementation steps".

### Possible HTTP server and Web application server scenarios

ODWEK can be installed on a system that is running AIX<sup>®</sup>, HP-UX, Linux, Solaris, Windows NT<sup>®</sup> 4.0 with Service Pack 5 or later, Windows 2000 Server, or Windows 2000 Advanced Server. These operating systems support many different HTTP servers and Web application servers. This book describes how to install and configure ODWEK with the following:

- IBM HTTP Server Version 1.3 or later.
- IBM WebSphere Application Server Version 4.0.3 or later.

If you need to install and configure ODWEK with some other HTTP server or Web application server, you must get specific instructions and support from the product vendor.

**Note:** IBM HTTP Server Version 1.3 or later and IBM WebSphere Application Server Version 4.0.3 or later must be installed, configured and operational before you install and configure ODWEK. The WebSphere Application Server information center at [www.ibm.com/software/webservers/appserv/library/](http://www.ibm.com/software/webservers/appserv/library/) contains complete installation and configuration instructions.

### Implementation steps

**Important:** An *instance* of ODWEK (sometimes called an *application*) is ODWEK code that accesses data on an OnDemand server. An instance controls what can be done to the data, and manages the system resources that are assigned to it. Each instance is a complete environment. An instance has its own ASWWW.INI file and ODWEK programming interface, which other instances cannot access. There are three ODWEK programming interfaces:

- CGI program
- Java servlet
- Java API

It is very important to understand that an instance may use only one programming interface. The programming interfaces are mutually exclusive. They cannot be used in the same instance at the same time. However, it is possible to run multiple instances of ODWEK on a single machine and have each instance use a different programming interface by configuring each instance to use a different port number.

Complete the following steps to perform a basic installation of the ODWEK software on the system:

1. Review the hardware and software requirements for ODWEK. See Chapter 3, "Installation requirements", on page 15.
2. Install the ODWEK software on the system. See Chapter 4, "Installing ODWEK", on page 19.

**Note:** After installing the ODWEK software, if you plan to use the Java API, see Appendix D, "Java API programming guide", on page 119 for information about setting up the system environment and running ODWEK applications.

3. If you plan to use the CGI version of ODWEK, deploy the CGI program. See Chapter 5, "Deploying the CGI program", on page 25.
4. If you plan to use the Java servlet, deploy the servlet. See Chapter 6, "Deploying the Java servlet", on page 27.
5. Verify and configure the ODWEK configuration file (ARSWWW.INI). See Chapter 7, "Specifying the ARSWWW.INI file", on page 35.
6. Modify the `logon.htm` sample application. See Chapter 8, "Configuring the sample applications", on page 71.
7. Install the viewers. See Chapter 9, "Installing the Web viewers", on page 75.
8. Verify the basic installation. See Chapter 10, "Verifying the installation", on page 79.

Depending on the types of documents that you plan to retrieve from the OnDemand server and how you plan to present them to your users, you may need to complete the following optional steps:

- If you plan to retrieve AFP documents or Metacode/DJDE documents from the server and process them with the Xenos transforms, see Appendix F, "Xenos transforms", on page 167. Contact your IBM representative for information about how to obtain the Xenos transform programs, license, and documentation. Your IBM representative can also provide information about education and other types of help and support for installing and configuring the transform programs and processing input files with the transform programs.

- If you plan to retrieve AFP documents from the server and convert them to HTML files with the AFP2WEB Transform, see Appendix G, “AFP to HTML transform”, on page 185. You must obtain the AFP2WEB Transform service offering from IBM and install and configure it on the system. See your IBM representative for more information about the AFP2WEB Transform service offering. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2WEB Transform. See Appendix G, “AFP to HTML transform”, on page 185 for more information about the configuration file.
- If you plan to retrieve AFP documents from the server and convert them to PDF files with the AFP2PDF Transform, see Appendix H, “AFP to PDF transform”, on page 189. You must obtain the AFP2PDF Transform service offering from IBM and install and configure it on the system server. See your IBM representative for more information about the AFP2WEB Transform service offering. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2PDF Transform. See Appendix H, “AFP to PDF transform”, on page 189 for more information about the configuration file. To view converted documents, you must obtain the Adobe Acrobat viewer for the browsers used by your organization.

---

## Your next step

After you have reviewed the implementation steps, verified that the HTTP server and the Web application server are installed, configured and operational, and verified that any optional transforms that you may require are installed and configured, you can now install the ODWEK software on the system. For installation procedures, see the following sections:

- Chapter 3, “Installation requirements”, on page 15.
- Chapter 4, “Installing ODWEK”, on page 19.



## Chapter 3. Installation requirements

Before you install the ODWEK software, you should make sure that your system meets ODWEK's hardware and software requirements.

This section describes the following requirements that you should take into consideration before installing ODWEK:

- "Installation requirements".
- "Disk space requirements" on page 16.
- "Transform requirements" on page 17.

### Installation requirements

You can install ODWEK on a system that is running AIX 4.3.3 or later, HP-UX 11.0 or later, Linux kernel 2.2 or higher, Solaris 8 or later, Windows NT Server 4.0, Windows 2000 Server, and Windows 2000 Advanced Server.

**Note:** IBM recommends that you install ODWEK on a different system than the one on which the OnDemand library server is running.

**Important:** An *instance* of ODWEK (sometimes called an *application*) is ODWEK code that accesses data on an OnDemand server. An instance controls what can be done to the data, and manages the system resources that are assigned to it. Each instance is a complete environment. An instance has its own ASWWW.INI file and ODWEK programming interface, which other instances cannot access. There are three ODWEK programming interfaces:

- CGI program
- Java servlet
- Java API

It is very important to understand that an instance may use only one programming interface. The programming interfaces are mutually exclusive. They cannot be used in the same instance at the same time. However, it is possible to run multiple instances of ODWEK on a single machine and have each instance use a different programming interface by configuring each instance to use a different port number.

ODWEK requires:

- For the CGI program, an HTTP server that supports the CGI protocol, such as the IBM HTTP Server. The HTTP server must be installed, configured and operational. This publication describes how to configure the CGI program with IBM HTTP Server Version 1.3.
- For the Java servlet, a Java-enabled HTTP server with a Java application server, such as the IBM WebSphere Application Server. The HTTP server and the Web application server must be installed, configured and operational. The servlet requires Java version 1.2.2 or later. Customers that plan to use Java version 1.3.1 to support the Java servlet must install Java version 1.3.1 with Fix Pack 4 or later. This publication describes how to configure the servlet with IBM WebSphere Application Server Version 4.0.3.
- For the Java API, Java version 1.2.2 or later.
- Appropriate media type for installation.
- Adequate disk space for installation files: approximately 25 MB on the Web server.

---

## Disk space requirements

You must allocate disk space for temporary files. Start with a minimum of 10 MB. The name of the temporary directory is specified with the TEMPDIR parameter in the ARSWWW.INI file. See “TEMPDIR” on page 44 for more information. For better performance, IBM recommends that you specify a directory and mount point that is different from those specified for the CACHEDIR and LOGDIR parameters.

If you plan to enable logging, you must allocate disk space for the log files and reports that are created by the HTTP server, (optionally) the Web application server, and ODWEK. Start with a minimum of 10 MB, although the exact size is a function of the number and type of logs and the activity on the system. For ODWEK, the name of the log directory is specified with the LOGDIR parameter in the ARSWWW.INI file. See “LOGDIR” on page 68 for more information. For the HTTP server, the name of the log directory is specified in the httpd.conf file. See your HTTP server information for directions. For the WebSphere, the name of the log directory is specified in the was.conf file. See your WebSphere information for directions. For better performance, IBM recommends that you specify a directory and mount point that is different from those specified for the CACHEDIR and TEMPDIR parameters.

ODWEK can *cache* (temporarily store) documents on the system. This can increase the speed with which previously viewed documents can be sent to users. To enable cache storage for documents:

- Configure the CACHEDOCS parameter in the ARSWWW.INI file. See “CACHEDOCS” on page 39 for more information.

- Specify a cache directory with the CACHEDIR parameter in the ARSWWW.INI file. For better performance, IBM recommends that you specify a directory and mount point that is different from those specified for the LOGDIR and TEMPDIR parameters. See “CACHEDIR” on page 39 for more information.
- Specify the amount of disk space (in MB) that should be reserved for cache storage. Start with a minimum of 10 MB, although the exact size will depend on the number documents that are retrieved from the system. See “CACHESIZE” on page 40 for more information.

You must make sure that the processes that run ODWEK programs can read from the directory that contains the programs and can write to the cache, log, and temporary directories. For example, an AIX system is typically configured to run ODWEK processes with the user identification (UID) of “nobody”, because “nobody” has minimal privileges. Verify that the permissions for ODWEK programs allow “nobody” to execute them. Verify that the permissions for the directory that contains the ODWEK programs allow “nobody” to read from the directory. Verify that the permissions for the cache, log, and temporary directories allow “nobody” to write to the directories.

---

## Transform requirements

If you plan to retrieve AFP documents or Metacode/DJDE documents from the server and process them with the Xenos transforms, see Appendix F, “Xenos transforms”, on page 167. Contact your IBM representative for information about how to obtain the Xenos transform programs, license, and documentation. Your IBM representative can also provide information about education and other types of help and support for installing and configuring the transform programs and processing input files with the transform programs.

If you plan to use the Java AFP2HTML Viewer, then you must obtain the AFP2WEB Transform service offering from IBM and install and configure it on the Web server. See your IBM representative for more information about the AFP2WEB Transform service offering. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2WEB Transform. See Appendix G, “AFP to HTML transform”, on page 185 for more information about the configuration file.

If you plan to convert AFP documents stored in OnDemand to PDF documents that can be viewed with the Adobe Acrobat viewer, then you must obtain the AFP2PDF Transform service offering from IBM and install and configure it on the Web server. See your IBM representative for more information about the AFP2PDF Transform service offering. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2PDF Transform. See Appendix H, “AFP to PDF

transform”, on page 189 for more information about the configuration file. To view converted documents, you must obtain the Adobe Acrobat viewer for the browsers used by your organization.

---

## Your next step

After you have determined that your system meets all hardware and software requirements, and after you have reviewed the implementation overview, you can now install the ODWEK software. For installation procedures, see Chapter 4, “Installing ODWEK”, on page 19.

---

## Chapter 4. Installing ODWEK

This section explains how to install the ODWEK software on the system.

---

### Before you begin

Before you begin the installation, make sure that your system meets all of the hardware and software requirements to install the ODWEK product. For more information, see Chapter 3, “Installation requirements”, on page 15.

---

### Your next step

To install ODWEK, go to the appropriate chapter:

- “Installing on AIX”.
- “Installing on HP-UX” on page 20.
- “Installing on Linux” on page 21.
- “Installing on Solaris” on page 21.
- “Installing on Windows servers” on page 22.

---

### Installing on AIX

**Note:** You can install ODWEK on an AIX Version 4.3.3 or later system.

To install the ODWEK software on the system:

1. Login with root privileges.
2. Use SMIT to install ODWEK. At the prompt, type `smitty install_latest`.
3. Click the INPUT device/directory for software field and press F4.
4. Select the installation media type and press Enter.
5. Click the Specify Software field and press F4.
6. Select the ODWEK software component and press F7.
7. Press Enter.
8. Type `no` in the COMMIT software entry field.
9. Type `yes` in the SAVE replaced files? field.
10. Press Enter.
11. Press Enter to confirm.
12. When the installation completes, press F10 to exit SMIT.

## Your next step

If you plan to use the CGI program, go to Chapter 5, “Deploying the CGI program”, on page 25.

If you plan to use the Java servlet, go to Chapter 6, “Deploying the Java servlet”, on page 27.

If you plan to use the Java API, go to Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.

---

## Installing on HP-UX

**Note:** You can install ODWEK on an HP-UX 11.0 or later system.

To install the ODWEK software on the system:

1. Login with root privileges.
2. Insert the ODWEK CD-ROM into the drive. The steps that follow assume that the CD-ROM drive is mounted on the directory /cdrom.
3. Install the ODWEK files using SWINSTALL. Use the following format of the command:

```
swinstall -s /cdrom/ODWEK/hpxx/OD71 -x mount_all_filesystems=FALSE
```

Where /ODWEK/hpxx/OD71 is the name of the installation image. (The name of the installation image is listed in the README file on the CD-ROM.)

4. From the options pull-down, review the installation options and change any that are required for your particular installation.
5. At the Software Selection Screen, select ODWEK.
6. From the Actions pull-down, select Mark for Install.
7. From the Actions pull-down, select Install.
8. Select OK.
9. Select Logfile to view the log for messages.
10. When the installation completes, remove the CD-ROM from the drive.

## Your next step

If you plan to use the CGI program, go to Chapter 5, “Deploying the CGI program”, on page 25.

If you plan to use the Java servlet, go to Chapter 6, “Deploying the Java servlet”, on page 27.

If you plan to use the Java API, go to Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.

---

## Installing on Linux

**Note:** To install ODWEK on a Linux system, you need the following:

- Linux kernel 2.2 or higher
- *glibc* Version 2.1 or higher
- *libstdc++* Version 2.9.0 or higher
- *rpm*

To install the ODWEK software on the system:

1. Login with root privileges.
2. Insert the ODWEK CD-ROM into the drive. The steps that follow assume that the CD-ROM drive is mounted on the directory */cdrom*.
3. Install the ODWEK files using *rpm*. Use the following format of the command:

```
rpm -iv /cdrom/ODWEK/linux/OD71.rpm
```

Where */ODWEK/linux/OD71.rpm* is the name of the *rpm* package. (The name of the *rpm* package is listed in the *README* file on the CD-ROM.)

4. When the installation completes, eject the CD-ROM from the drive.

### Your next step

If you plan to use the CGI program, go to Chapter 5, “Deploying the CGI program”, on page 25.

If you plan to use the Java servlet, go to Chapter 6, “Deploying the Java servlet”, on page 27.

If you plan to use the Java API, go to Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.

---

## Installing on Solaris

**Note:** You can install ODWEK on a Solaris 8 or later system.

To install the ODWEK software on the system:

1. Login with root privileges.

2. Insert the ODWEK CD-ROM into the drive. The steps that follow assume that the CD-ROM drive is mounted on the directory /cdrom.
3. Install the ODWEK files using PKGADD. Use the following format of the command:

```
/usr/sbin/pkgadd -d /cdrom/ODWEK/sun/OD71
```

Where /ODWEK/sun/OD71 is the name of the installation image. (The name of the installation image is listed in the README file on the CD-ROM.)

4. Select ODWEK.
5. Enter Y to continue.
6. When the installation completes, eject the CD-ROM from the drive.

### Your next step

If you plan to use the CGI program, go to Chapter 5, “Deploying the CGI program”, on page 25.

If you plan to use the Java servlet, go to Chapter 6, “Deploying the Java servlet”, on page 27.

If you plan to use the Java API, go to Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.

---

## Installing on Windows servers

To install the ODWEK software on the system:

1. Log on to Windows with a user account that has administrator privileges.
2. Insert the ODWEK installation media into the drive.
3. Choose Run from the Start menu.
4. In the Open box, type D:\odwek71.exe, where D represents the drive that contains the installation media and odwek71 is the name of the installation program. (The name of the installation program is listed in the README file on the CD-ROM.)
5. Click OK.
6. The installation program starts. Follow the instructions on the screen. You may need to select check boxes and choose the Continue button on successive screens to complete the installation.

### Your next step

If you plan to use the CGI program, go to Chapter 5, “Deploying the CGI program”, on page 25.

If you plan to use the Java servlet, go to Chapter 6, “Deploying the Java servlet”, on page 27.

If you plan to use the Java API, go to Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.



# Chapter 5. Deploying the CGI program

This section explains how to deploy the CGI version of ODWEK on the system:

- “Before you begin”.
- “Copying GGI program files”.

## Before you begin

Before you begin deploying the CGI program:

- Make sure that you have completed the software installation. See Chapter 4, “Installing ODWEK”, on page 19.
- Make sure that you have the current version of the IBM HTTP server installed, configured, and operating on the system.

## Copying GGI program files

Complete the following steps:

1. Copy the ARSWWW.CGI file to a directory on the system that has been designated as executable and that can contain CGI programs. For example:

<server\_root>/cgi-bin

2. For Windows systems, copy the following files to the directory in which you copied the ARSWWW.CGI file:

ARSSCKNT.DLL  
ARSCT32.DLL

3. Copy the following files to the directory in which you copied the ARSWWW.CGI file:

ARSWWW.INI  
AFP2HTML.INI  
AFP2PDF.INI

## Your next step

After you have deployed the CGI program, you can now configure the ODWEK initialization file for your operating environment. See Chapter 7, “Specifying the ARSWWW.INI file”, on page 35.



# Chapter 6. Deploying the Java servlet

This section explains how to deploy the Java servlet version of ODWEK on the system.

## Before you begin

Before you begin deploying the servlet:

- Make sure that you have completed the software installation. See Chapter 4, "Installing ODWEK", on page 19.
- Make sure that you have the current version of the IBM HTTP server installed, configured and operating on the system.
- Make sure that you have the current version of the IBM WebSphere Application Server installed, configured and operating on the system.

IBM recommends that you use the WebSphere tools to deploy the servlet. The WebSphere tools automatically configure the HTTP server and Web application server configuration files. An experienced Web server administrator may choose not to use the WebSphere tools, and deploy the servlet by manually configuring the HTTP server and Web application server configuration files. See the HTTP server and WebSphere information for directions.

To use the WebSphere tools to deploy the servlet, follow these steps:

1. "Copying files".
2. "Deploying the servlet using WebSphere tools" on page 29.

## Copying files

Complete the following steps:

1. In most installations, the Web application server locates servlet class files in the servlet root directory:

`<server_root>/lib`

For example: `c:\WebSphere\AppServer\lib` or  
`/opt/WebSphere/AppServer/lib`

Copy the `ArsWWWServlet.jar` file to this directory.

2. Copy the `ArsWWWInterface.class` file to a directory that is set by the CLASSPATH variable for the Web application server. Because this file is

part of a package (`com.ibm.edms.od`), you must mirror the package structure as subdirectories under this directory.

For example, if the directory `<server_root>/classes` is set by the `CLASSPATH` variable, then you must copy the `ArsWWWInterface.class` file to the `<server_root>/classes/com/ibm/edms/od` directory.

3. Copy the shared library to a directory that is set by the shared library path variable:

*Table 1. Shared Library Directory*

Operating System	Shared Library Path Variable	Shared Library
AIX	LIBPATH	libarswwwsl.so
HP-UX	SHLIB_PATH	libarswwwsl.so
Linux	LD_LIBRARY_PATH	libarswwwsl.so
Solaris	LD_LIBRARY_PATH	libarswwwsl.sl
Windows	PATH	arswwwsl.dll

4. For Windows systems, copy these files to the directory in which you copied the shared library:

`ARSSCKNT.DLL`  
`ARSCT32.DLL`

5. Copy these files:

`ARSSWW.INI`  
`AFP2HTML.INI`  
`AFP2PDF.INI`

to the HTTP server bin directory:

*Table 2. HTTP Server Directory*

Operating System	HTTP Server Directory
AIX	/usr/lpp/IBM HTTP Server/bin
HP-UX	/opt/IBM HTTP Server/bin
Linux	/opt/IBM HTTP Server/bin
Solaris	/opt/IBM HTTP Server/bin
Windows	c:\IBM HTTP Server\bin

## Your next step

To use the WebSphere tools to deploy the servlet, go to “Deploying the servlet using WebSphere tools” on page 29.

## Deploying the servlet using WebSphere tools

The WebSphere tools are the recommended method for deploying the servlet. The WebSphere tools can perform all of the tasks required to deploy the servlet.

There are two steps in deploying the servlet using the WebSphere tools:

1. Assembling the application with the WebSphere Application Assemble Tool (AAT). See "Assembling the application".
2. Installing the application from the WebSphere administration console. See "Installing the application" on page 31.

### Assembling the application

1. Start the WebSphere Application Assembly Tool (AAT).

*Table 3. Starting AAT*

Operating System	Start Command
AIX	/usr/lpp/WebSphere/AppServer/bin/assembly.sh
HP-UX	/opt/WebSphere/AppServer/bin/assembly.sh
Linux	/opt/WebSphere/AppServer/bin/assembly.sh
Solaris	/opt/WebSphere/AppServer/bin/assembly.sh
Windows	Start -> Programs -> IBM WebSphere -> Application Server v4.0 -> Administrator's Console

2. Click Create Application Wizard.
3. In the Display Name field, type a name for your ODWEK application. For example: OnDemand WEK.
4. In the File Name field, type a name for the configuration files. For example: odwek.ear.
5. Click Next.
6. Click Next at the Adding Supplementary Files window.
7. Click Next at the Choosing Application Icons window.
8. Click Next at the Adding EJB Modules window.
9. Click Next at the Adding Web Modules window
10. Click Next at the Adding Application Client Modules
11. Click Finish. A window will open in the AAT with the details of the application that you are creating.
12. Right click on Web Modules and select New from the pop-up list.

13. Complete the fields listed in Table 4.

*Table 4. Assembling the application*

Field	Value
<b>File Name</b> Specifies the name of the configuration file.	odwek.war
<b>Context Root</b> Specifies the name of the virtual root directory for this module.	/od
<b>Classpath</b> Specifies the location of the JAR file, from Step 1 on page 27.	/usr/lpp/WebSphere/AppServer/lib (AIX) /opt/WebSphere/AppServer/lib (HP-UX) /opt/WebSphere/AppServer/lib (Linux) /opt/WebSphere/AppServer/lib (Solaris) c:\websphere\appserver\lib (Windows)
<b>Display Name</b> Specifies the name of this module.	OD WEK Module

14. Click OK.
15. Expand OD WEK Module (**Display Name**) by clicking on the +
16. Right click on Web Components and select New from the pop-up list.
17. In the Component Name field, type ArsWWWServlet, which is case sensitive.
18. In the Component Type field, click Servlet.
19. For Class Name, click Browse.
20. Locate the ArsWWWServlet.jar file. The location of the JAR file was determined in Step 1 on page 27.
21. Expand the path COM -> IBM -> EDMS -> OD and click on the ArsWWWServlet.class file.
22. Click OK.
23. Expand the ArsWWWServlet Web component.
24. Right click on Initialization Parameters and select New from the pop-up menu.

25. Complete the fields listed in Table 5.

*Table 5. Assembling the application*

Field	Value
<b>Parameter Name</b> Specifies the parameter to be passed to this module. Important: The value is <b>case sensitive</b> .	ConfigDir
<b>Parameter Value</b> Specifies the location of the ARSWWW.INI file. See Step 5 on page 28.	/usr/lpp/IBM HTTP Server/bin (AIX) /opt/IBM HTTP Server/bin (HP-UX) /opt/IBM HTTP Server/bin (Linux) /opt/IBM HTTP Server/bin (Solaris) c:\IBM HTTP Server\bin (Windows)

26. Click OK.

27. Right click on Servlet Mapping and select New from the pop-up menu. The servlet name will already be filled in. You need to specify the URL pattern that you would like to use when calling the servlet from within the Web browser. The URL pattern includes a user-defined name, for example: arswww.

**Important:** You must specify the URL pattern in the following format:  
`/arswww/`, where arswww is the user-defined name.

28. Click Apply.
29. Click File from the top of the ATT task bar and select Save As from the menu.
30. Choose a location to save the configuration (such as the directory from Step 1 on page 27) and specify a name for the file (such as odwek.ear).

Next, install the application. See “Installing the application”.

## Installing the application

**Note:** You must assemble the application before you proceed. See “Assembling the application” on page 29

1. Start the WebSphere administration console.

*Table 6. Starting WebSphere administration console*

Operating System	Start Command
AIX	/usr/lpp/WebSphere/AppServer/bin/adminclient.sh
HP-UX	/opt/WebSphere/AppServer/bin/adminclient.sh
Linux	/opt/WebSphere/AppServer/bin/adminclient.sh

*Table 6. Starting WebSphere administration console (continued)*

Operating System	Start Command
Solaris	/opt/WebSphere/AppServer/bin/adminclient.sh
Windows	Start -> Programs -> IBM WebSphere -> Application Server v4.0 -> Administrator's Console (Or run adminclient.bat from the WebSphere bin directory.)

2. From the WebSphere administration console, expand WebSphere Admin Domain.
3. Right click on Enterprise Applications and selected Install Enterprise Application from the pop-up menu.
4. Choose the Node for this application.
5. Click Browse to specify the location of the odwek.ear file.
6. Type an Application Name. For example: ODWEK.
7. Click Next.
8. Click Next at Mapping User to Roles.
9. Click Next at Mapping EJB RunAs.
10. Click Next at Binding Enterprise Beans.
11. Click Next at Mapping EJB References.
12. Click Next at Mapping Resource References.
13. Click Next at Specify the Default Datasource.
14. Click Next at Specifying datasources for CMP Beans.
15. Specify the Virtual Host for your Web Module. For example, Default Host.
16. Click Next.
17. Select the Application Server for your Application. For example, Default Server.
18. Click Next.
19. Click Finish.
20. Expand Nodes.
21. Right Click on your node and select Regen WebSphere Plugin from the pop-up menu.
22. Expand Enterprise Applications.
23. Right click on your OD WEK application and select Start from the pop-up menu.
24. Stop and restart the Web application server.

## Your next step

After you have deployed the Java servlet using the WebSphere tools, you can now configure the ODWEK initialization file for your operating environment. See Chapter 7, “Specifying the ARSWWW.INI file”, on page 35.



---

## Chapter 7. Specifying the ARSWWW.INI file

The ARSWWW.INI file is an ASCII text file that contains parameters that are read by ODWEK programs (such as the CGI program or the Java servlet). You specify each parameter on a separate line using the following format: PARAMETER=value. For example:

```
AFPVIEWING=plugin  
CACHEDIR=/ars/www/cache  
LANGUAGE=ENU
```

The parameters in the ARSWWW.INI file are grouped into sections. You specify the beginning of a section using a section header, in the following format: [sectionHeader]. You specify the parameters for a section after the section header. For example:

```
[@SRV@_odserver1]  
HOST=odserver1.xyz.com  
PORT=1445  
PROTOCOL=0
```

An example ARSWWW.INI configuration file is provided with the product. The example configuration file provides a set of the most commonly used values. “Example ARSWWW.INI file” on page 69 shows the example.

The sections and parameters for the ARSWWW.INI file are as follows:

---

### **[@SRV@\_DEFAULT]**

The default server section. You can use the default server section to specify parameters that are common to the OnDemand servers with which ODWEK will communicate. The parameters and values that you specify in this section will be used unless you specify them in a server section.

This section has a global scope for all servers, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section may contain the following parameters:

#### **PORT**

The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the

| OnDemand service is not specified in the SERVICES file, then port number  
| 1445 will be used. Before using any port number, verify with your TCP/IP  
| administrator that the port is available.

| You can specify this parameter once in the default section. When using the  
| Logon API, you can override the specified port number with the \_port  
| parameter.

| This parameter is optional.

| Example:

```
[@SRV@_DEFAULT]  
PORT=1445
```

## PROTOCOL

The networking protocol that is used to communicate between the OnDemand library server and ODWEK. The only valid value is 0 (zero), for TCP/IP.

You must specify this parameter once in the default section.

This parameter is optional. If not specified, a value of 0 (zero) is used.

Example:

```
[@SRV@_DEFAULT]  
PROTOCOL=0
```

---

## [@SRV@\_server]

A server section. You must specify one server section for each OnDemand server with which ODWEK will communicate. A server section contains the parameters and values for a specific server. The section header must include the string that identifies the server. The parameters specified in a server section override the parameters found in the default server section.

You must specify one server section for each server.

This section is optional. However, if not specified, the system will use the values that are specified (or defaulted to) in the [@SRV@\_DEFAULT] section.

This section may contain the following parameters:

## HOST

The name of the OnDemand server. You can specify the TCP/IP address, host name alias, or fully-qualified host name of the server.

You must specify this parameter once in the server section.

This parameter is required.

Example:

```
[@SRV@_odserver1]  
HOST=odserver1.xyz.com
```

## PORT

The TCP/IP port number that ODWEK and the OnDemand library server use to communicate.

You can specify this parameter once in the server section. When using the Logon API, you can override the specified port number with the `_port` parameter.

This parameter is optional. If not specified, the value that is specified (or defaulted to) in the `[@SRV@_DEFAULT]` section is used.

Example:

```
[@SRV@_odserver1]  
PORT=1445
```

## PROTTOCOL

The networking protocol that is used to communicate between the OnDemand library server and ODWEK. The only valid value is 0 (zero), for TCP/IP.

You can specify this parameter once in the server section.

This parameter is optional. If not specified, the value that is specified (or defaulted to) in the `[@SRV@_DEFAULT]` section is used.

Example:

```
[@SRV@_odserver1]  
PROTOCOL=0
```

---

## [CONFIGURATION]

The CONFIGURATION section contains parameters that are used by ODWEK on the Web server.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is required.

This section can contain the following parameters:

## APPLETCACHEDIR

Specifies the directory in which the Line Data applet and the AFP2HTML applet temporarily store documents. The directory can be local to the user's workstation or on a network drive. All users must have write access to the specified directory.

**Example:**

```
[Configuration]
APPLETCACHEDIR=n:\temp
```

**Notes:**

1. The APPLETCAHEDIR parameter has a global scope.
2. The APPLETCAHEDIR parameter is optional. However, if this parameter is not specified, the applets will attempt to store documents in the Java working directory.
3. If the specified directory does not exist, the applets will attempt to store documents in the Java working directory.
4. The applet removes a document from the cache directory when the user leaves the applet (for example, closes the document).

## APPLETDIR

Identifies the directory that contains the Line Data and AFP2HTML applets.

**Notes:**

1. You can specify a directory name or an *alias*:
  - If you specify a directory name, the directory must be relative to the ServerRoot directory. For example, if you specify appletdir=applets and the ServerRoot directive is set to /usr/lpp/ars/www, then the applets must exist in the /usr/lpp/ars/www/applets directory. The ServerRoot directive is defined in the Web server configuration file.
  - If you specify an alias, then it must be defined in the Web server configuration file. For example, if you specify appletdir=/applets/, then the Web server configuration file must have an alias for /applets/. The alias must be set to the full path name of the directory on the server. For example:

```
Alias      /applets/      /usr/lpp/ars/www/applets
```

**Note:** Chapter 5, "Deploying the CGI program", on page 25 and Chapter 6, "Deploying the Java servlet", on page 27 contain examples of Web server configuration files.

2. Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the applet directory.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is required.

Example:

```
[CONFIGURATION]
APPLETDIR=applets
```

## CACHEDIR

Specifies the directory on the Web server in which ODWEK temporarily stores (*caches*) server data and optionally, documents (see “CACHEDOCS”).

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is required.

Example:

```
[CONFIGURATION]
CACHEDIR=/ars/www/cache
```

### Notes:

1. See Chapter 3, “Installation requirements”, on page 15 for information about the cache directory.
2. Verify the permissions of the directory that you specify. The processes that run ODWEK programs must write to and read from the cache storage directory.
3. For better performance, the directory and mount point that you specify for the CACHEDIR parameter should be different than those that you specify for the TEMPDIR parameter.

## CACHEDOCS

Determines whether ODWEK temporarily stores (*caches*) documents on the Web server. Cache storage can increase the speed with which previously viewed documents are retrieved from the server. The default value is 0 (zero), which means that cache storage for documents is not enabled. Specify a 1 (one) to enable cache storage for documents. If you enable cache storage for documents, verify the directory in which ODWEK caches documents (see “CACHEDIR”) and the amount of disk space reserved for cache storage (see “CACHESIZE” on page 40).

**Note:** IBM recommends that you always enable cache storage for documents when you use the Microsoft® Internet Explorer browser and the AFP Web Viewer or the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

| This parameter is optional. **Note:** If usage patterns is such that a document is  
| viewed only once by a single user, then caching may degrade the performance  
| of the system.

Example:

```
[CONFIGURATION]  
CACHEDOCS=1
```

## CACHEMAXTHRESHOLD

Determines when ODWEK begins deleting data and documents from cache storage. ODWEK begins deleting data and documents when the percentage of disk space used in cache storage is equal to or greater than the value specified. The default value is 80 (eighty percent). ODWEK deletes items from cache storage until a threshold is reached (see "CACHEMINTHRESHOLD").

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional.

Example:

```
[CONFIGURATION]  
CACHEMAXTHRESHOLD=80
```

## CACHEMINTHRESHOLD

Determines when ODWEK stops deleting data and documents from cache storage. ODWEK stops deleting data and documents when the percentage of disk space used in cache storage is less than or equal to the value specified. The default value is 40 (forty percent). ODWEK begins deleting items from cache storage when a threshold is reached (see "CACHEMAXTHRESHOLD").

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional.

Example:

```
[CONFIGURATION]  
CACHEMINTHRESHOLD=40
```

## CACHESIZE

The amount of disk space that ODWEK can use to temporarily store (*cache*) data and documents on the Web server. Specify the value in megabytes. The default value is 10 (ten megabytes).

**Note:** To enable cache storage for documents, see "CACHEDOCS" on page 39.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional. However, when caching documents, the more disk space that you allocate, the more documents ODWEK can store on the Web server. Generally, this can increase the speed with which ODWEK sends previously viewed documents to users. **Note:** If usage patterns is such that a document is viewed only once by a single user, then caching may degrade the performance of the system.

Example:

```
[CONFIGURATION]
CACHESIZE=1024
```

## CACHEUSERIDS

Specifies a comma separated list of OnDemand userids for which ODWEK uses data from cache storage to complete the logon process. For the specified userids, multiple logon attempts will bypass the standard OnDemand logon processing, except if the data is not in cache storage or if the Inactivity Time Out value (see the system parameters on the OnDemand server) is reached. Separate each userid with the comma character.

### Notes:

1. The CACHEUSERIDS parameter is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 library server.
2. If the userid is case sensitive on the server (see the system parameters on the OnDemand server), then you must specify the userid exactly as it was defined to OnDemand.
3. The userids listed in the CACHEUSERIDS list can access only those folders whose names and other information are currently in cache storage. The users will not be able to access folders created after they log on to an OnDemand server. To allow a userid listed in the CACHEUSERIDS list to access a new folder, either delete the user's name from the CACHEUSERIDS list or purge the cache.
4. To specify that ODWEK should use data from cache storage for all OnDemand users, specify CACHEUSERIDS=\*.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional.

Example:

```
[CONFIGURATION]
CACHEUSERIDS=oduser1,oduser2,oduser3
```

## **CODEPAGE**

Identifies the code page of the OnDemand database. The default code page is the code page of the Web server.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section. When using the Logon API, you can override the specified code page with the \_codepage parameter.

This parameter is optional. However, if the Web server is running in a different code page than the database, then you must specify the CODEPAGE parameter.

Example:

```
[CONFIGURATION]
CODEPAGE=943
```

## **DOCSIZE**

When retrieving documents from the OnDemand server, determines the maximum size (in bytes) of a document that can be written directly to memory instead of first writing the document to disk. Any document that is less than or equal to the value specified will be written directly to memory. Any document that exceeds the specified value will first be written to disk and then read from disk into memory before it is delivered to the browser. A lower value may conserve system resources, while a higher value will improve viewing performance. The range is from 0 (zero) to *n* bytes, where *n* is the amount of available memory on the system. A value of zero defaults the size to 1 MB. If this parameter is not specified or if the value is not defined or recognized, the size defaults to 1 MB.

This parameter has a global scope, and you specify it only once in the CONFIG section.

This parameter is optional.

Example:

```
[CONFIGURATION]
DOCSIZE=524287
```

## **IMAGEDIR**

Identifies the directory that contains image files used by ODWEK. The default value is /usr/lpp/ars/www/images.

### **Notes:**

1. ODWEK concatenates the value that you specify with the file names found on HTML image tags. For example, if you specify:

```
imagedir=images
```

Then the HTML image tag for the View Document function will appear in the output as follows:

```
<IMG SRC="images/odic_vd.gif">
```

2. You can specify a directory name or an *alias*:

- If you specify a directory name, then the directory must be relative to the ServerRoot directory. For example, if you specify imagedir=images and the ServerRoot directory is set to /usr/lpp/ars/www, then the images must exist in the /usr/lpp/ars/www/images directory. The ServerRoot directory is set in the Web server configuration file.
  - If you specify an alias, then the alias must be defined in the Web server configuration file. For example, if you specify imagedir=/images/, then the Web server configuration file must have an alias for /images/. The alias must be set to the full path name of the directory on the server.
- For example:

```
Alias      /images/      /usr/lpp/ars/www/images
```

**Note:** Chapter 5, “Deploying the CGI program”, on page 25 and Chapter 6, “Deploying the Java servlet”, on page 27 contain examples of Web server configuration files.

3. Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the image directory.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is required.

Example:

```
[CONFIGURATION]
IMAGEDIR=/usr/lpp/ars/www/images
```

## LANGUAGE

Identifies the language in which ODWEK displays messages. The default language is English (ENU). ODWEK supports the following languages:

Value	Country or Region
ARA	Egypt
CHS	China
CHT	Taiwan
DAN	Denmark
DEU	Germany
ENU	US/English

Value	Country or Region
ESP	Spain
FIN	Finland
FRA	France
FRC	Canada
ITA	Italy
JPN	Japan
KOR	Korea
NLD	Netherlands
NOR	Norway
PTB	Brazil
SVE	Sweden

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional.

Example:

```
[CONFIGURATION]
LANGUAGE=JPN
```

## TEMPDIR

Use to specify the directory in which ODWEK will store temporary files.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is required.

Example:

```
[CONFIGURATION]
TEMPDIR=/ars/www/tmp
```

### Notes:

1. See Chapter 3, “Installation requirements”, on page 15 for information about the temporary directory.
2. Verify the permissions of the directory that you specify. The processes that run ODWEK programs must write to and read from the cache storage directory.

- 3. For better performance, the directory and mount point that you specify for the TEMPDIR parameter should be different than those that you specify for the CACHEDIR parameter.

## TEMPLATEDIR

Identifies the directory that contains the HTML template files. ODWEK uses the template files to generate Web pages in response to the various product functions (such as Logon, Search, Retrieve Document, and so forth). By default, ODWEK retrieves the template files from the `usr/lpp/ars/www/samples` directory.

**Note:** Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the template directory.

This parameter has a global scope, and you specify it only once in the CONFIGURATION section.

This parameter is optional.

Example:

```
[CONFIGURATION]
TEMPLATEDIR=/usr/lpp/ars/www/samples
```

---

## [SECURITY]

The SECURITY section contains the security parameters that are used by ODWEK on the Web server.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## REPORTSERVERTIMEOUT

Use to specify that ODWEK should use the Inactivity Time Out parameter from the OnDemand library server. The Inactivity Time Out parameter determines when the server can terminate a session with an inactive user. To specify that ODWEK should use the Inactivity Time Out parameter, set the REPORTSERVERTIMEOUT parameter to 1 (one).

This parameter has a global scope, and you specify it only once in the SECURITY section.

This parameter is optional. If not specified or set to 0 (zero), ODWEK will not use the Inactivity Time Out parameter, meaning that ODWEK will not terminate a session with an inactive user. For more information about the Inactivity Time Out parameter, see the online help for the administrative client.

Example:

```
[SECURITY]  
REPORTSERVERTIMEOUT=1
```

## SERVERACCESS

Specifies a comma separated list of the OnDemand library servers that ODWEK can access. If specified, the users that use ODWEK and the programs that use the APIs are permitted to access only those servers that are listed. You can specify the host name alias of the server, or the TCP/IP address or fully qualified host name of the server.

This parameter has a global scope, and you specify it only once in the SECURITY section.

This parameter is optional. If you do not specify this parameter (or you specify a blank list), then ODWEK can access any OnDemand library server.

Example:

```
[SECURITY]  
SERVERACCESS=odserver1,odserver2
```

---

## [AFP2HTML]

The AFP2HTML section contains the parameters that are used by the AFP2WEB Transform. The AFP2WEB Transform converts AFP documents and resources into HTML documents that can be displayed with the AFP2HTML applet.

### Notes:

1. To convert AFP documents to HTML documents, an administrator must obtain the AFP2WEB Transform service offering from IBM and install and configure it on the server. See your IBM representative for more information about the AFP2WEB Transform service offering. Someone in your organization must also provide configuration options for the AFP2WEB Transform. See Appendix G, "AFP to HTML transform", on page 185 for more information about the configuration file.
2. To convert documents with the AFP2WEB Transform, you must specify the AFPVIEWING=HTML parameter in the DEFAULT BROWSER section (or other browser sections). See "AFPVIEWING" on page 59 for details. (If you plan

- to use the Retrieve Document API, then you should specify the `_afp=HTML` parameter. See “Retrieve Document” on page 103 for details.)
3. By default, ODWEK uses the AFP2HTML applet to view converted documents. If a converted document was stored in OnDemand as a large object, then the AFP2HTML applet provides controls to help users easily move to any page in the document.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## CONFIGFILE

The configuration file that contains the options used by the AFP2WEB Transform to convert AFP documents and resources into HTML data, fonts, and images that can be viewed with the AFP2HTML applet. Appendix G, “AFP to HTML transform”, on page 185 shows the sample configuration file provided with OnDemand. See the AFP2WEB Transform documentation for details about the options that you can specify in the configuration file.

This parameter has a global scope, and you specify it only once in the AFP2HTML section.

This parameter is optional.

Example:

```
[AFP2HTML]
CONFIGFILE=afp2html.ini
```

## INSTALLDIR

The directory that contains the AFP2WEB Transform programs, configuration files, and mapping files. Specify the full path name of the directory on the Web server.

**Note:** Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the install directory.

This parameter has a global scope, and you specify it only once in the AFP2HTML section.

This parameter is optional.

Example:

```
[AFP2HTML]
INSTALLDIR=/usr/lpp/ars/www/bin/afp2web
```

## USEEXECUTABLE

Determines whether ODWEK starts the AFP2WEB Transform by using the shared library or the executable. The default value is 0 (zero) and means that ODWEK uses the shared library. If you experience problems running the transform from the shared library, then set this parameter to 1 (one).

This parameter has a global scope, and you specify it only once in the AFP2HTML section.

This parameter is optional.

Example:

```
[AFP2HTML]
USEEXECUTABLE=1
```

---

## [AFP2PDF]

The AFP2PDF section contains the parameters that are used by the IBM AFP2PDF Transform. The AFP2PDF Transform converts AFP documents and resources into PDF documents that can be viewed with the Adobe Acrobat viewer.

### Notes:

1. To convert AFP documents to PDF documents, an administrator must obtain the AFP2PDF Transform service offering from IBM and install and configure it on the Web server. See your IBM representative for more information about the AFP2PDF Transform service offering. Someone in your organization must also provide configuration options for the AFP2PDF Transform. See Appendix H, "AFP to PDF transform", on page 189 for more information about the configuration file.
2. To convert documents with the AFP2PDF Transform, you must specify the AFPVIEWING=PDF parameter in the DEFAULT BROWSER (or other browser sections). See "AFPVIEWING" on page 59 for details. (If you plan to use the Retrieve Document API, then you should specify the \_afp=PDF parameter. See "Retrieve Document" on page 103 for details.)
3. By default, ODWEK uses the Adobe Acrobat viewer to view converted documents. You must obtain the Adobe Acrobat viewer for the browsers that are used in your organization.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## CONFIGFILE

The configuration file that contains the options used by the AFP2PDF Transform to convert AFP documents and resources into PDF documents that can be viewed with the Adobe Acrobat viewer. Appendix H, “AFP to PDF transform”, on page 189 shows the sample configuration file provided with OnDemand. See the AFP2PDF Transform documentation for details about the options that you can specify in the configuration file.

This parameter has a global scope, and you specify it only once in the AFP2PDF section.

This parameter is optional.

Example:

```
[AFP2PDF]
CONFIGFILE=afp2pdf.ini
```

## INSTALLDIR

The directory that contains the AFP2PDF Transform programs, configuration files, and mapping files. Specify the full path name of the directory on the Web server.

**Note:** Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the install directory.

This parameter has a global scope, and you specify it only once in the AFP2PDF section.

This parameter is optional.

Example:

```
[AFP2PDF]
INSTALLDIR=/usr/lpp/ars/www/bin/afp2pdf
```

## USEEXECUTABLE

Determines whether ODWEK starts the AFP2PDF Transform by using the shared library or the executable. The default value is 0 (zero) and means that ODWEK uses the shared library. If you experience problems running the transform from the shared library, then set this parameter to 1 (one).

This parameter has a global scope, and you specify it only once in the AFP2PDF section.

This parameter is optional.

Example:

```
[AFP2PDF]  
USEEXECUTABLE=1
```

---

## [MIMETYPES]

The MIMETYPES section identifies the Multipurpose Internet Mail Extension (MIME) content type for documents that will be retrieved from the OnDemand server. The browser uses the MIME content type to format and display the document, to choose the correct applet or viewer to open the document, or to start a user-defined program to open the document.

### Notes:

1. The MIMETYPES section should contain a parameter=*value* pair for each type of document that you plan to retrieve from the OnDemand server. The parameter identifies the data type of the document in OnDemand. (This is the data type that is assigned to the OnDemand application on the View Information page.) The *value* determines the program that is started to open the document. The *value* is case sensitive.
2. In the example ARSWWW.INI file (see “Example ARSWWW.INI file” on page 69), the MIMETYPES section contains a parameter for each of the standard data types supported by OnDemand (AFP, BMP, EMAIL, GIF, JFIF, LINE, PCX, PDF, and TIFF).
3. In addition to the standard data types, OnDemand also supports user-defined data types. A user-defined data type can identify any other type of data that you want to store on the system. Before users can view documents that have a user-defined data type, you must add a parameter to the MIMETYPE section. The parameter must identify the MIME content type of the data and the file extension that was specified for the OnDemand application on the View Information page. The file extension must also be registered with the operating system on the PC. For example, suppose you define an application to store Lotus® WordPro documents in OnDemand. You specify the file extension as LWP on the application View Information page. To configure the system to recognize documents retrieved from the application, add the following parameter to ARSWWW.INI file:

```
[MIMETYPES]  
LWP=application/lwp
```

Then, when a user retrieves a document from the application, ODWEK sets the MIME content type to application/lwp and the system starts Lotus WordPro to open the document. For Netscape, the MIME content type must be defined in Preferences->Navigator->Applications.

Table 7 on page 51 lists the MIME content types for several PC applications:

*Table 7. MIME content types for several PC applications*

Lotus Applications	WK1=application/lotus-1-2-3 WK3=application/lotus-1-2-3 WK4=application/lotus-1-2-3 123=application/lotus-1-2-3 APR=application/lotus-approach VEW=application/lotus-approach LWP=application/lotus-wordpro SAM=application/lotus-wordpro MWP=application/lotus-wordpro SMM=application/lotus-wordpro PRE=application/lotus-freelance PRZ=application/lotus-freelance
Microsoft Applications	DOC=application/ms-word XLS=application/ms-excel PPS=application/ms-powerpoint PPT=application/ms-powerpoint MPD=application/ms-project MPP=application/ms-project MPT=application/ms-project MPD=application/ms-project
HTML Applications	HTML=application/html HTM=application/htm

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## AFP

The MIME content type for AFP documents, when AFPVIEWING=NATIVE is specified in the [DEFAULT BROWSER] section. See “[AFPVIEWING](#)” on page 59 for more information. This specifies the MIME type for the document that the browser then uses to determine what program should be used process the document.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional.

Example:

```
[MIMETYPES]
AFP=application/afp
```

## BMP

The MIME content type for BMP documents. By default, BMP documents are displayed using the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then ODWEK sets the MIME content type to `image/bmp` and starts the program that is associated with the BMP file type on the PC.

Example:

```
[MIMETYPES]
BMP=image/IBM-OnDemand
```

## GIF

The MIME content type for GIF documents. By default, GIF documents are displayed using the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then ODWEK sets the MIME content type to `image/gif` and uses the browser's built-in viewer to display GIF documents.

Example:

```
[MIMETYPES]
GIF=image/IBM-OnDemand
```

## EMAIL

The MIME content type for EMAIL documents. See "EMAILVIEWING" on page 61 for more information about processing EMAIL documents before sending them to the viewer.

### Notes:

1. If you convert EMAIL documents to HTML, ODWEK sets the MIME content type to `text/html`. ODWEK ignores the value of the EMAIL parameter, if specified.

2. If you extract and uncompress EMAIL documents from OnDemand, ODWEK uses the value of the EMAIL parameter to determine the program to open the document.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional.

Example:

```
[MIMETYPES]
EMAIL=text/plain
```

## JFIF

The MIME content type for JFIF (JPEG) documents. By default, JFIF documents are displayed using the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then ODWEK sets the MIME content type to `image/jpeg` and starts the program that is associated with the JPEG file type on the PC.

Example:

```
[MIMETYPES]
JFIF=image/IBM-OnDemand
```

## LINE

The MIME content type for line data documents. See “LINEVIEWING” on page 62 for more information about processing line data documents before sending them to the viewer.

This is used when LINEVIEWING=NATIVE is specified in the [DEFAULT BROWSER] section. If you extract and uncompress line data documents from OnDemand, ODWEK uses the value of the LINE parameter to determine the program to start to open the document.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional.

Example:

```
[MIMETYPES]
LINE=text/html
```

## **PCX**

The MIME content type for PCX documents. By default, PCX documents are displayed using the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then ODWEK sets the MIME content type to `image/pcx` and starts the program that is associated with the PCX file type on the PC.

Example:

```
[MIMETYPES]  
PCX=image/IBM-OnDemand
```

## **PDF**

The MIME content type for PDF documents.

**Notes:**

1. ODWEK uses the value of the PDF parameter to determine the program to start to open PDF documents. By default, PDF documents are opened with the Adobe Acrobat viewer.
2. To view PDF documents, you should obtain and install the Adobe Acrobat viewer for the browsers that are used by your organization.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional.

Example:

```
[MIMETYPES]  
PDF=application/pdf
```

## **TIFF**

The MIME content type for TIFF documents. By default, TIFF documents are displayed using the Image Web Viewer.

This parameter has a global scope, and you specify it only once in the MIMETYPES section.

This parameter is optional. However, if you do not specify this parameter, then ODWEK sets the MIME content type to `image/tiff` and starts the program that is associated with the TIFF file type on the PC.

Example:

```
[MIMETYPES]
TIFF=image/IBM-OnDemand
```

---

## [ATTACHMENT IMAGES]

The ATTACHMENT IMAGES section identifies the image files that ODWEK uses to display attachments to a document. Each image file should contain an icon that represents a specific type of attachment. For example, you can identify an image file that contains an icon for a text attachment, a bitmap attachment, and so forth.

### Notes:

1. Each parameter that you specify must identify the file type that the operating system associates with the type of attachment. The file type determines the program that the operating system starts to process the attachment. For example, if the operating system associates the file type TXT with text file attachments, add a `TXT=value` parameter to the ATTACHMENT IMAGES section. As the `value`, specify the name of the file that contains the icon that you want to use to indicate a text attachment to a document. When the user clicks on the icon, the operating system starts the program that is registered to open TXT documents.
2. By default, all attachments to a document are indicated by the `odic_att.gif` file (which is located in the directory that is specified by the `IMAGEDIR` parameter in the CONFIGURATION section). OnDemand also uses the `odic_att.gif` file for any file types for which a parameter is not specified in the ATTACHMENT IMAGES section.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

### BMP

The parameter identifies the bitmap data type. The value identifies the file that contains the icon to represent a bitmap image attached to the document.

This parameter has a global scope, and you specify it only once in the ATTACHMENT IMAGES section.

This parameter is optional.

Example:

```
[ATTACHMENT IMAGES]
BMP=userBitMap.gif
```

## **GIF**

The parameter identifies the GIF data type. The value identifies the file that contains the icon to represent a GIF image attached to the document.

This parameter has a global scope, and you specify it only once in the ATTACHMENT IMAGES section.

This parameter is optional.

Example:

```
[ATTACHMENT IMAGES]
GIF=userGIF.gif
```

## **TXT**

The parameter identifies the TXT data type. The value identifies the file that contains the icon to represent a text file attached to the document.

This parameter has a global scope, and you specify it only once in the ATTACHMENT IMAGES section.

This parameter is optional.

Example:

```
[ATTACHMENT IMAGES]
TXT=userText.gif
```

---

## **[NO HTML]**

The NO HTML section contains the parameters that are used to override the default characters that delimit strings and separate a list of values in the delimited ASCII output. A function generates delimited ASCII output when you set its \_nohtml parameter to 1 (one). See Appendix K, “No HTML output” , on page 201 for details about the delimited ASCII output.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## **BEGIN**

The character that ODWEK uses to delimit the beginning of a string or a string of values. You must change the BEGIN delimiter if a string contains the default character (the [ character).

This parameter has a global scope, and you specify it only once in the NO HTML section.

This parameter is optional.

Example:

```
[NO HTML]  
BEGIN=<
```

## END

The character that ODWEK uses to delimit the end of a string or a string of values. You must change the END delimiter if a string contains the default character (the ] character).

This parameter has a global scope, and you specify it only once in the NO HTML section.

This parameter is optional.

Example:

```
[NO HTML]  
END=>
```

## SEPARATOR

The character that ODWEK uses to separate a string of values. You must change the SEPARATOR delimiter if a string contains the default character (the ^character ).

This parameter has a global scope, and you specify it only once in the NO HTML section.

This parameter is optional.

Example:

```
[NO HTML]  
SEPARATOR=;
```

---

## [DEFAULT BROWSER]

You can use the DEFAULT BROWSER section to specify parameters for the browsers used by your organization. The parameters that you specify will be used unless you specify them in a browser section. (The parameters specified in a browser section override those from the DEFAULT BROWSER section. See “[browser]” on page 66.)

This section has a global scope for all browsers, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## **ADDEXENSION**

Determines whether the three-character file extension of the document is added to the extra path information of the URL that is returned to the browser. Adding the file extension to the URL can help browsers determine the correct viewer to start for the document. The default value is 0 (zero) and means that the file extension is not added to the URL.

**Note:** If you use the Microsoft Internet Explorer browser, then IBM recommends that you specify ADDEXENSION=1 so that the file extension is added to the URL.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
ADDEXENSION=1
```

## **ADDFIELDSTODOCID**

Determines whether the field values are added to the document identifiers. (The document identifiers are returned by the Document Hit List function.) The default value is 0 (zero) and means that the field values are not added to the document identifiers. If you enable ODWEK to add the field values to the document identifiers, then they will also appear in the system log, provided that you have configured the system to save application group messages in the system log.

**Notes:**

1. If you use the Update Document function, then you must specify ADDFIELDSTODOCID=1.
2. If the Annotation Flags in the document database table field is set to Yes, then you must specify ADDFIELDSTODOCID=1. You can set the Annotations Flags in document database table field on the Database Information dialog box, from the General page in the OnDemand application group definitions. (Click Advanced to open the Database Information dialog box.)

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
ADDFIELDSTODOCID=1
```

## ADDNOTES

Determines whether annotations can be added to documents. If enabled, ODWEK puts a control for adding annotations next to each document in the document list. The default value is 0 (zero) and means that annotations cannot be added to documents.

Notes:

1. Users are permitted or denied the ability to add annotations to documents based on the Annotation permissions in the OnDemand application group.
2. The ADDNOTES parameter is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK V7.1.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
ADDNOTES=1
```

## AFPVIEWING

When a user retrieves an AFP document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer. For example, some customers convert AFP documents to HTML with the AFP2WEB Transform and use the AFP2HTML applet to view the HTML output. Those customers should specify AFPVIEWING=HTML so that ODWEK will convert the AFP document before sending it to the viewer.

You can set the parameter to one of the following values:

- |              |  |
|--------------|--|
| <b>ASCII</b> | ODWEK converts AFP documents to ASCII text.                                |
| <b>HTML</b>  | ODWEK converts AFP documents to HTML documents with the AFP2WEB Transform. |

<b>NATIVE</b>	ODWEK extracts and uncompresses AFP documents and their resources from OnDemand.
	<b>Note:</b> If you specify AFPVIEWING=NATIVE, verify that the MIME content type for AFP documents identifies the viewer that you want to use. See “[MIMETYPES]” on page 50 for details.
<b>PDF</b>	ODWEK converts AFP documents to PDF documents with the AFP2WEB Transform.
	<b>Note:</b> If you specify AFPVIEWING=PDF, verify that the MIME content type for PDF documents identifies the viewer that you want to use. See “[MIMETYPES]” on page 50 for details.
<b>PLUGIN</b>	ODWEK does not convert AFP documents (the default).

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the \_afp parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
AFPVIEWING=PLUGIN
```

## AUTODOCRETRIEVAL

Specifies whether the viewer automatically displays a document when one and only one document matches the query. This capability means that, for queries that you know will match only one document, you can set up the system to bypass the document list Web page and display the document without the user taking action. The default value is 0 (zero) and means that ODWEK will display the document list Web page, even if only one document matches the query.

**Important:** If you are using Internet Explorer, you must specify AUTODOCRETRIEVAL=0 to disable the automatic document retrieval function. Beginning with Version 7.1.0.11, IBM has set AUTODOCRETRIEVAL=0 as the default value in the [IE] section of the ARSWWW.INI file that is distributed with ODWEK.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
AUTODOCRETRIEVAL=1
```

## EMAILVIEWING

When a user retrieves an EMAIL document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer.

You can set this parameter to one of the following values:

**NATIVE**      ODWEK extracts and uncompresses EMAIL documents from OnDemand.

**Note:** If you specify EMAIL=NATIVE, verify that the MIME content type identifies the viewer that you want to use. See “[MIMETYPES]” on page 50 for details.

**HTML**      ODWEK converts EMAIL documents to HTML documents.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the \_email parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
EMAILVIEWING=HTML
```

## ENCRYPTCOOKIES

Determines whether ODWEK encrypts cookies that are sent to the browser. The default value is 0 (zero), meaning that cookies will not be encrypted. Specify 1 (one) to encrypt all cookies that are sent to the browser.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
ENCRYPTCOOKIES=1
```

## **ENCRYPTURL**

Determines whether ODWEK encrypts the server, userid, password, and docid values that are contained in the URL that is sent to the browser. The default value is 0 (zero), meaning that these values will not be encrypted. Specify 1 (one) to encrypt these values.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional. However, if you must use the GET method to transfer form parameters and values between the browser and the Web server, then you can encrypt these values by specifying ENCRYPTURL=1. See “Server and data security” on page 9 for more information about the method attribute of the form tag.

Example:

```
[DEFAULT BROWSER]  
ENCRYPTURL=1
```

## **FOLDERDESC**

Specifies whether the folder description is displayed to the right of the folder name on the folder selection page. The default value is 0 (zero), meaning that the folder description will not be displayed. Specify 1 (one) to display the folder description. If this parameter is not specified or if the value is not defined or recognized, the folder description will not be displayed.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
FOLDERDESC=1
```

## **LINEVIEWING**

When a user retrieves a line data document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer.

You can set this parameter to one of the following values:

**APPLET**      ODWEK converts line data documents for viewing with the Line Data applet (the default).

<b>ASCII</b>	ODWEK converts line data documents to ASCII text.
<b>NATIVE</b>	ODWEK extracts and uncompresses line data documents from OnDemand.

**Note:** If you specify LINEVIEWING=NATIVE, verify that the MIME content type identifies the viewer that you want to use. See “[MIMETYPES]” on page 50 for details.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the \_line parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
LINEVIEWING=APPLET
```

## MAXHITS

The maximum number of items returned to the document list, regardless of the number of items that match the query.

**Notes:**

1. The document list is filled with items that match a query in the order in which the items were loaded into the database.
2. ODWEK uses one of the following values to determine the number of items to return to the document list:
  - For the Document Hit List function, the value of the Maximum Hits field (specified on the folder Permissions page). This value overrides all other values.
  - For the Document Hit List and Print Document functions, the value of the \_max\_hits parameter, if specified for a function. The value of the \_max\_hits parameter overrides the MAXHITS parameter.
  - The value of the MAXHITS parameter, if specified.
  - If none of the above are specified, ODWEK returns a maximum of 200 items to the document list.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
MAXHITS=200
```

## NOLINKS

Determines whether the document list contains controls for viewing documents. If enabled, ODWEK adds a control next to each document. To view a document, the user must use the control. The default value is 0 (zero) and means that the user must use a text link to view a document.

**Important:** You must set NOLINKS=0 if you are using the Microsoft Internet Explorer browser. IBM suggests that you specify NOLINKS=0 in any browser sections that you define for Internet Explorer.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
NOLINKS=1
```

## ODApplet.jre.path.IE

See Appendix E, “Java line data viewer”, on page 161.

## ODApplet.jre.path.NN

See Appendix E, “Java line data viewer”, on page 161.

## ODApplet.jre.version

See Appendix E, “Java line data viewer”, on page 161.

## ODApplet.version

See Appendix E, “Java line data viewer”, on page 161.

## SERVERPRINT

Determines whether the document list contains controls for sending documents to a server printer. If enabled, ODWEK adds a control next to each document. The default value is 0 (zero) and means that users must open a document before they can send it to a server printer.

### Notes:

1. To use server print, at least one server printer must be defined to the OnDemand server.
2. Users are permitted or denied the ability to print documents based on the Print permissions in the OnDemand application group.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
SERVERPRINT=1
```

## SERVERPRINTERS

Use to specify the type of server print devices that the user can select. There are three types of server print devices:

- | P Server Printer
- | I Server Printer with Information
- | F Server Fax

You can specify from zero to three types, in a comma-separated list.

The following example shows how to specify that the user can select server printer and server fax devices:

```
[DEFAULT BROWSER]
SERVERPRINTERS=P,F
```

## SHOWDOCLOCATION

When generating delimited ASCII output rather than HTML (see Appendix K, “No HTML output”, on page 201), determines whether the storage location of the document will appear in the output. See “Document Hit List” on page 203 for details. The default value is 0 (zero) and means that the storage location will not appear in the output.

**Notes:**

- | 1. The SHOWDOCLOCATION parameter is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK V7.1.
- | 2. To display the storage location, you must also set the Display Document Location property in the OnDemand folder.

This parameter has a global scope, unless overridden in a browser section (see “[browser]” on page 66). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
SHOWDOCLOCATION=1
```

## VIEWNOTES

Determines whether annotations to documents can be viewed. If enabled, ODWEK puts a control for viewing annotations next to each document in the document list. The default value is 0 (zero) and means that annotations cannot be viewed.

### Notes:

1. Users are permitted or denied the ability to view annotations to documents based on the Annotation permissions in the OnDemand application group.
2. The VIEWNOTES parameter is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK V7.1.

This parameter has a global scope, unless overridden in a browser section (see "[browser]"). You specify this parameter only once in the DEFAULT BROWSER section.

This parameter is optional.

### Example:

```
[DEFAULT BROWSER]
VIEWNOTES=1
```

---

## [browser]

You can specify options for the specific browsers used by your organization. Except for the parameters noted in Item 1 below, the parameters that you specify in a browser section override the parameters from the DEFAULT BROWSER section of the ARSWWW.INI file. (The parameters that you specify in the DEFAULT BROWSER section will be used unless you specify them in a browser section.)

### Notes:

1. The following parameters have a global scope and may only be specified in the DEFAULT BROWSER section. (If these parameters are specified in some other browser section, they will be ignored.)
  - ODApplet.jre.path.IE
  - ODApplet.jre.path.NN
  - ODApplet.jre.version
  - ODApplet.version
2. The section header must contain a string that identifies the browser for which you want to specify the options. ODWEK extracts the value of the

HTTP\_USER\_AGENT environment variable to determine the browser being used. ODWEK then searches the ARSWWW.INI file for a browser section that matches the value. If no browser section is found, ODWEK then searches the ARSWWW.INI file for one of the following sections:

```
[browser version(major.minor)/platform]  
[browser version(major.minor)]  
[browser version(major)]  
[browser]  
[DEFAULT BROWSER]
```

ODWEK uses the options from the first section that matches the value.

3. For the browser, you can specify IE or Netscape.
4. For the platform, you can specify Win95, Win98 or WinNT.

A browser section has a global scope for the specified browser. Specify only one browser section for each browser. You should specify only the parameters that you need to override from the DEFAULT BROWSER section.

This section is optional.

This section can contain the same parameters that are defined for the default browser. See “[DEFAULT BROWSER]” on page 57.

Examples:

```
[IE 5]  
AUTODOCRETRIEVAL=0  
NOLINKS=0  
  
[Netscape 4.6]  
AUTODOCRETRIEVAL=1  
NOLINKS=1
```

---

## [DEBUG]

The DEBUG section contains options that you can use to help solve problems that you and others in your organization are having using ODWEK.

The DEBUG section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## LOG

Enables ODWEK to write messages and other program information to a log file. (The log file is named ARSWWW.LOG.)

This parameter has a global scope, and you specify it only once in the DEBUG section.

This parameter is optional. By default, ODWEK does not write messages to a log file. Specify a value of 1 (one) to log messages.

Example:

```
[DEBUG]  
LOG=1
```

## LOGDIR

Determines the directory in which ODWEK writes the ARSWWW.LOG file, if logging is enabled using the LOG parameter.

This parameter has a global scope, and you specify it only once in the DEBUG section.

This parameter is optional. By default, if logging is enabled, ODWEK writes the log file to the /tmp directory.

Example:

```
[DEBUG]  
LOGDIR=/ars/www/log
```

### Notes:

1. See Chapter 3, “Installation requirements”, on page 15 for information about the log file directory.
2. Verify the permissions of the directory that you specify. The processes that run ODWEK programs must write to and read from the log directory.
3. For better performance, the directory and mount point that you specify for the LOGDIR parameter should be different than those that you specify for the CACHEDIR and TEMPDIR parameters.
4. When logging is enabled, the system appends new information to the existing ARSWWW.LOG file. If you enable logging and leave it on for extended periods of time, not only will the performance of the system suffer, but the log file can grow very large, very quickly.

---

## Example ARSWWW.INI file

An example ARSWWW.INI configuration file is provided with the product. The example configuration file sets the most commonly used default values for servers, browsers (Netscape Navigator), and viewers.

```
[@SRV@_<DEFAULT]
PORT=0
PROTOCOL=0

;[@SRV@_odserver1]
;HOST=odserver1.mycompany.com
;PORT=1445
;PROTOCOL=0

[CONFIGURATION]
APPLETDIR=applets
CACHEDIR=/ars/www/cache
CACHEDOCS=1
CACHEMAXTHRESHOLD=80
CACHEMINTHRESHOLD=40
CACHESIZE=1024
CACHEUSERIDS=*
IMAGEDIR=images
LANGUAGE=ENU
TEMPDIR=/ars/www/tmp
TEMPLATEDIR=templates

[SECURITY]
SERVERACCESS=

[AFP2HTML]
CONFIGFILE=afp2html.ini
INSTALLDIR=
USEEXECUTABLE=0

[AFP2PDF]
CONFIGFILE=afp2pdf.ini
INSTALLDIR=
USEEXECUTABLE=0
```

(Continued on next page.)

(Example ARSWWW.INI file, continued.)

```
[MIMETYPES]
AFP=application/afp
BMP=image/IBM_OnDemand
EMAIL=text/plain
GIF=image/IBM_OnDemand
JFIF=image/IBM_OnDemand
LINE=text/html
PCX=image/IBM_OnDemand
PDF=application/pdf
TIFF=image/IBM_OnDemand

[ATTACHMENT IMAGES]
BMP=
GIF=
TXT=

[NO HTML]
BEGIN>[
END=]
SEPARATOR=\n

[DEFAULT BROWSER]
ADDEXTION=0
ADDFIELDSTODOCID=0
ADDNOTES=1
AFPVIEWING=plugin
AUTODOCRETRIEVAL=1
EMAILVIEWING=native
FolderDesc=0
LINEVIEWING=applet
MAXHITS=200
NOLINKS=1
ODApplet.version=1
SERVERPRINT=1
SHOWDOCLOCATION=0
VIEWNOTES=1

[IE]
AUTODOCRETRIEVAL=0

[DEBUG]
LOG=0
LOGDIR=/ars/www/log
```

---

## Your next step

After you have verified the ARSWWW.INI file, go to Chapter 8, “Configuring the sample applications”, on page 71

---

## Chapter 8. Configuring the sample applications

**Note:** If you are using the Java API to control ODWEK, see Appendix D, “Java API programming guide”, on page 119 for information about setting up the system environment and running ODWEK applications.

This chapter explains how to customize the sample applications that are provided with ODWEK for the CGI program and the Java servlet.

- LOGON.HTM. This application supports users that are permitted to access several folders. Each user is defined to the OnDemand library server. After logging on to the server, ODWEK shows the user the list of folders that the user is permitted to open. “LOGON.HTM” contains instructions for customizing this application.
- CREDIT.HTM. This application supports casual use of OnDemand. The user is presented with search criteria for a specific folder. The OnDemand server name, userid and password, folder name, and folder fields are coded in the application. “CREDIT.HTM” on page 72 contains instructions for customizing this application.
- FCREDIT.HTM. A version of the CREDIT application that demonstrates the use of HTML frames.

After you modify the sample applications, publish the URL of each file so that users can link to them and access OnDemand. Each sample requires a different level of customization. There are complete instructions for customizing the CREDIT.HTM sample application. Use the instructions as a guide for customizing other applications that you may need.

**Note:** In addition to modifying the sample applications, IBM recommends that you customize the TEMPLATE.HTM file for your organization. The TEMPLATE.HTM file contains user-defined content that ODWEK uses to display Web pages. See “TEMPLATE.HTM” on page 73 for important information about modifying this file.

---

### LOGON.HTM

1. Copy the logon.htm file from the installation directory to the document root directory of the HTTP server.

*Table 8. Caption*

Operating system	Installation Directory
AIX	/usr/lpp/ars/www/samples

*Table 8. Caption (continued)*

Operating system	Installation Directory
HP-UX	/opt/ondemand/www/samples
Linux	/opt/ondemand/www/samples
Solaris	/opt/ondemand/www/samples
Windows	X:\installation directory\samples, where X is the installation drive and installation directory is the directory in which you installed the ODWEK software

2. For the CGI program, verify that the logon.htm file contains the following lines:

```
<h4>Please enter your logon information:</h4>
<FORM METHOD=POST ACTION="/arswww.cgi">
```

3. For the servlet, verify that the logon.htm file contains the following line:

```
<FORM METHOD=POST ACTION="/ArsWWWServlet">
```

---

## CREDIT.HTM

Customize the CREDIT.HTM sample application by making a copy of the file for each folder that you want users to access. The name of the file should be the same as the name of the folder.

1. Edit the CREDIT.HTM file. (By default, this file is located in the /usr/lpp/ars/www/samples directory.)
2. Change or delete the background image specified in the <body> statement (line 11).
3. Optionally change the background color specified in the <body> statement (line 11).
4. Change or delete the product image specified in the <img> statement (line 12).
5. Replace the folder name specified in the <h1> statement (line 15).
6. Replace the text specified in the <p> statements (lines 17 through 25). Enter general instructions to the user.
7. Replace the CGI-BIN directory name specified in the <FORM> statement (line 29). Enter the name of the CGI-BIN directory that contains the ODWEK programs and files on the Web server.
8. Replace the value specified in the <input> statement (line 30). This is a comma separated string that contains the names of the folder display fields.
9. Replace the value specified in the <input> statement (line 31). This is the name of the folder.

10. Replace the value specified in the <input> statement (line 33). This is the maximum number of items displayed in the document list, regardless of the number of items that match the query.
11. Replace the server name specified in the <input> statement (line 35). This is the name of the OnDemand server with which ODWEK is to communicate. The supplied server name is gunnar.
12. If you want to sort items in the document list, verify the value specified in the <input> statement (line 36). Otherwise, delete line 36.
13. If you want to sort items in the document list, verify the value specified in the <input> statement (line 37). Otherwise, delete line 37.
14. Replace the value specified in the <input> statement (line 38). This is the OnDemand userid. The userid that you specify must have permission to open the folder and access application group data.
15. Optionally change the name of the template file specified in the <input> statement (line 39). OnDemand uses the template file to generate subsequent Web pages. The supplied template name is template.htm.
16. Modify lines 40 through 43 for the first folder search field.
  - a. Type a name for the folder field in the <font> statement.
  - b. Replace the value specified in the name field of the <input> statement with the actual folder field name.
  - c. Replace the value specified in the value field of the <input> statement with the default search value.
17. Copy lines 40 through 43 and repeat step 16 for each additional folder search field.
18. Save your changes and close the text editor.

---

## TEMPLATE.HTM

The TEMPLATE.HTM file is the default template file used by ODWEK to generate Web pages in response to the various product functions (such as Logon). You should replace this file with one that contains user-defined content. However, the template file must contain the following HTML comment line:

```
<!-- - - AOI# Marker - - -->
```

The location of comment line determines where ODWEK program places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.

By default, the template file is located in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. See “TEMPLATEDIR” on page 45 for details.

---

## Your next step

After you have configured the sample applications, go to Chapter 9, “Installing the Web viewers”, on page 75

---

# Chapter 9. Installing the Web viewers

---

## Overview

IBM provides viewers for the standard types of documents that can be retrieved from OnDemand. The installation requirements vary, depending on the viewers that users need to use.

- To view line data documents, IBM recommends that you use the Java Line Data Viewer. The Java Line Data Viewer is an applet that is stored on the Web server. After an administrator enables the use of the Java Line Data Viewer, it is automatically loaded into memory on the workstation when the user selects to view a line data document. Verify that the LINEVIEWING parameter in the ARSWWW.INI file specifies the viewer that your users will be using.
- To view AFP documents, you can use the IBM OnDemand AFP Web Viewer, the Adobe Acrobat viewer, or the Java AFP2HTML Viewer.
  - To view AFP documents with the IBM OnDemand AFP Web Viewer, users must install it on the PC.
  - To view AFP documents with the Adobe Acrobat viewer, an administrator must install and configure either the Xenos transform or the AFP2PDF transform on the system and configure the ARSWWW.INI file. After an administrator enables the use of the transform, by default, the browser will attempt to start the Adobe Acrobat viewer when the user selects to view an AFP document. The user must obtain and install the Adobe Acrobat viewer on the PC.
  - To view AFP documents with the Java AFP2HTML Viewer, an administrator must install and configure the AFP2WEB transform on the system and configure the ARSWWW.INI file. The Java AFP2HTML Viewer is stored on the Web server. After an administrator enables the use of the AFP2HTML applet, it is automatically loaded into memory on the PC when the user selects to view an AFP document.

Verify that the AFPVIEWING parameter in the ARSWWW.INI file specifies the viewer that your users will be using.

- To view BMP, GIF, JPEG, PCX, and TIFF documents, IBM recommends that your users install the IBM OnDemand Image Web Viewer on their PCs; otherwise, they should use some other viewer that handles these types of documents. (For example, most browsers have built-in viewers capable of viewing GIF and JPEG.) If your users decide to use some other viewer, make sure that an administrator changes the default MIME content type for

these types of documents. Verify that the parameters in the MIMETYPES section of the ARSWWW.INI file specify the viewers that your users will be using.

**Notes:**

1. To view other types of data, you may need to install other viewers. For example, to view PDF documents that are retrieved from the OnDemand server, IBM recommends that you obtain and install the Adobe Acrobat viewer for the browsers used in your organization.
2. The nppdf32.dll file is required in the browser plugin directory to view PDF documents. For Internet Explorer, it should be in the \Program Files\Internet Explorer\PLUGINS directory. For Netscape, it should be in the \Program Files\Netscape\Communicator\Program\Plugins directory. If the file is not in the browser directory, you will need to reinstall the Adobe software.

---

## Requirements

The viewers that are provided by IBM require Netscape Navigator 4.06 or later or Microsoft Internet Explorer 4.01 or later.

ODWEK requires the ability to write cookie data on the PC. Make sure that your users configure their browsers to accept cookies.

IBM provides two versions of the Java line data viewer in the applets directory:

**ODLineDataViewer.jar** is the old Java line data viewer, which requires Java support in the browser. Java support is most likely provided by a Java Virtual Machine (JVM). If you are using Microsoft Internet Explorer Version 5.5 or later, then when you install the browser, you must use the Custom installation option to select and install the JVM.

**ODLineDataViewer2.jar** is the new Java line data viewer, which requires Version 1.4.1 or later of the Java plugin. The new Java line data viewer does not use the Java support in the browser. The user must install the Java plugin on the PC to use the new Java line data viewer. See Appendix E, "Java line data viewer", on page 161 for important configuration information.

**Note:** The default product installation will use the old Java line data viewer.

The browser must run under Windows 2000, Windows 98, Windows NT 4.0 with SP5 or later, or Windows XP and requires the following hardware and software:

- Physical connection to the network, such as a Token Ring or Ethernet network adapter

- TCP/IP
- A minimum of 32 MB of RAM
- An IBM-compatible PC with a 166 MHz or faster processor
- A super VGA display and adapter with a minimum resolution of 800 x 600
- A minimum of 20 MB of free disk space to view documents
- Approximately 3 MB on each workstation that needs the IBM OnDemand AFP Web Viewer and 2 MB on each workstation that needs the IBM OnDemand Image Web Viewer.

---

## Installation

**Note:** If you plan to distribute user-defined files with the AFP Web Viewer, then you should configure the AFP Web Viewer installation file to hold the user-defined files before your users begin installing the AFP Web Viewer. See Appendix I, "Distributing user-defined files", on page 193 for more information.

The viewers that are provided by IBM are installed using self extracting files. These files should be downloaded to the user's Windows 98, Windows NT, Windows 2000, or Windows XP system and run to install the appropriate viewer. If the user is running a browser while the installation is in progress, then the user must stop and restart the browser before the viewer can be used. The following viewer files can be found in the /usr/lpp/ars/www/plugins directory:

- afpplgus.exe - IBM OnDemand AFP Web Viewer - English only
- afpplgin.exe - IBM OnDemand AFP Web Viewer - All languages including DBCS support
- afpplgin.zip - IBM OnDemand AFP Web Viewer - Zip format for all languages include DBCS support
- imgplgin.exe - IBM OnDemand Image Web Viewer - All languages

The installation process copies the viewer and its associated files to directories of the user's choice. The AFP Web Viewer requires approximately 3 MB of space on the PC. The Image Web Viewer requires approximately 2 MB of space on the PC. Remind your users to restart their browser if it is active during the installation process.

**Note:** The installation program will install the viewers as either plug-ins or ActiveX controls. If Internet Explorer is installed on the PC, then the installation program will install the ActiveX controls; if Netscape is installed on the PC, then the installation program will install the plug-ins. If the user has both Internet Explorer and Netscape installed

| on the workstation, then the installation program will install the  
| ActiveX controls for Internet Explorer and the plug-ins for Netscape.

---

## **Your next step**

After you have installed the ODWEK software, configured the HTTP server, configured the Web application server, verified the ARSWWW.INI file, configured the sample applications, and installed the Web viewers, you should verify the installation before you begin using ODWEK. For verification procedures, see Chapter 10, “Verifying the installation”, on page 79.

## Chapter 10. Verifying the installation

At this point, you should have completed all of the steps in the basic installation for ODWEK.

You can verify that ODWEK is installed correctly by logging on to an OnDemand library server and opening a folder. If you are using the CGI program, go to “Verifying the CGI program”. If you are using the Java servlet, go to “Verifying the servlet” on page 81. **Note:** If you are using the Java API, see Appendix D, “Java API programming guide”, on page 119 for information about configuring the system and using the Java interpreter to run an ODWEK application.

### Verifying the CGI program

You can verify the installation by performing the following steps:

**Important:** Before you begin, restart the HTTP server to initialize the system with the changes that you have made to the configuration files.

1. Verify the HOST, PORT, and PROTOCOL parameters in the `[@SRV@_default]` section of the `arswww.ini` file. The default location of the `arswww.ini` file is:

AIX	<code>/usr/lpp/ars/www/cgi-bin</code>
HP-UX	<code>/opt/ondemand/www/cgi-bin</code>
Linux	<code>/opt/ondemand/www/cgi-bin</code>
Solaris	<code>/opt/ondemand/www/cgi-bin</code>
Windows	<code>x:\yyyyyyyy\BIN</code> , where x is the installation drive and yyyyyyyy is the directory in which you installed the ODWEK software

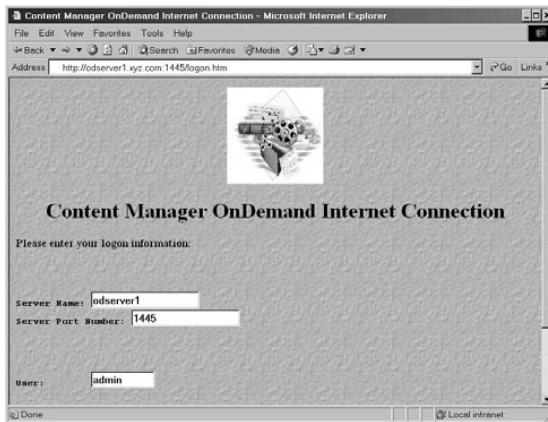
See “[@SRV@\_DEFAULT]” on page 35. **Note:** The value of the PORT parameter in the ARSWWW.INI file is the port number on which the OnDemand library server is running, not the port number that the IBM HTTP Server listens for requests from the client (browser).

2. Start a browser.
3. On the Address line of the browser, type a URL that includes the OnDemand library server, HTTP port, and logon function. For example:

`http://odserver1.xyz.com:80/logon.htm`

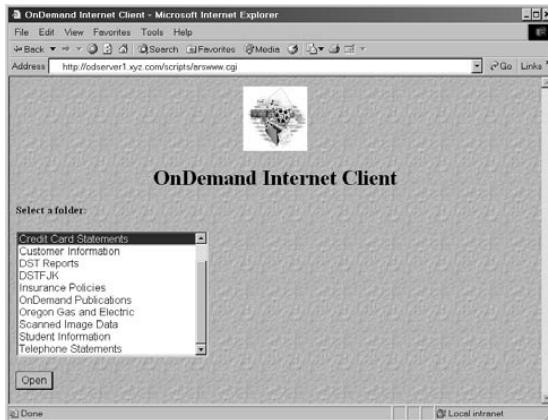
Where `odserver1.xyz.com` is the value of the HOST parameter in the ARSWWW.INI file, 80 is the HTTP port, and `logon.htm` specifies the function that ODWEK should invoke. In this example, ODWEK will

- invoke the logon function to log on to the specified OnDemand library server. (The `logon.htm` file is one of the sample applications that are provided with ODWEK. See Chapter 8, “Configuring the sample applications”, on page 71 for instructions about how to deploy the sample applications.)
- If the system is configured correctly, ODWEK should display the Logon screen. Figure 6 shows an example.



*Figure 6. Logon Screen in ODWEK.*

- If the Logon screen did not appear, see “Troubleshooting” on page 82.
- From the Logon screen, type a userid and password that is valid on the OnDemand library server. Click Submit to proceed to the Open a Folder screen. Figure 7 shows an example.



*Figure 7. Logon Screen in ODWEK.*

- At this point, the basic installation is successful. However, you may need to continue the validation process by retrieving different types of documents to test any transforms that you may have integrated with ODWEK.

---

## Verifying the servlet

**Note:** Before you proceed, if you have not already done so, stop and restart the HTTP server and the Web application server.

To verify that the servlet works correctly, start a Web browser and open the servlet. Specify the location of the servlet. For example:

`http://server/od/arswww`

Where server is the hostname of the system on which you are deploying the servlet, od is the Content Root that you specified in Step 13 on page 30 and arswww is the Servlet Mapping that you specified in Step 27 on page 31.

If you see a Web page with the text Internet Connection Version 7.1.0.x and The argument '\_function' was not specified, then the deployment was successful.

## Troubleshooting

This section describes the typical errors when attempting to verify the installation and contains possible solutions for those errors.

*Table 9. Troubleshooting ODWEK Installation.*

Problem	Solution																				
The Logon screen did not appear.	<p>If the Logon screen did not appear, the most likely cause is that the mapping rules for the <code>logon.htm</code> file are incorrect. The mapping rules are specified in the <code>httpd.conf</code> file. See your HTTP server information for directions.</p> <p>If the mapping rules are correct:</p> <ol style="list-style-type: none"><li>1. Verify that the permissions on the samples directory:<table><tr><td>AIX</td><td>/usr/lpp/ars/www/samples</td></tr><tr><td>HP-UX</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Linux</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Solaris</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Windows</td><td>x:\yyyyyyy\SAMPLES, where x is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software</td></tr></table></li><li>2. Verify the permissions on the <code>logon.htm</code> file in the samples directory:<table><tr><td>AIX</td><td>/usr/lpp/ars/www/samples</td></tr><tr><td>HP-UX</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Linux</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Solaris</td><td>/opt/ondemand/www/samples</td></tr><tr><td>Windows</td><td>x:\yyyyyyy\SAMPLES, where X is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software</td></tr></table></li><li>3. Verify that the HTTP server is operational.</li></ol> <p>Make the necessary corrections, then restart the HTTP server and the Web application server and try the logon process again.</p>	AIX	/usr/lpp/ars/www/samples	HP-UX	/opt/ondemand/www/samples	Linux	/opt/ondemand/www/samples	Solaris	/opt/ondemand/www/samples	Windows	x:\yyyyyyy\SAMPLES, where x is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software	AIX	/usr/lpp/ars/www/samples	HP-UX	/opt/ondemand/www/samples	Linux	/opt/ondemand/www/samples	Solaris	/opt/ondemand/www/samples	Windows	x:\yyyyyyy\SAMPLES, where X is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software
AIX	/usr/lpp/ars/www/samples																				
HP-UX	/opt/ondemand/www/samples																				
Linux	/opt/ondemand/www/samples																				
Solaris	/opt/ondemand/www/samples																				
Windows	x:\yyyyyyy\SAMPLES, where x is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software																				
AIX	/usr/lpp/ars/www/samples																				
HP-UX	/opt/ondemand/www/samples																				
Linux	/opt/ondemand/www/samples																				
Solaris	/opt/ondemand/www/samples																				
Windows	x:\yyyyyyy\SAMPLES, where X is the installation drive and yyyyyyy is the directory in which you installed the ODWEK software																				
Error 404, file not found	<p>If the Logon screen appeared, but you received an Error 404 message when you tried to log on to the server, verify the file mappings in the <code>httpd.conf</code> file. See your HTTP server information for directions.</p> <p>Make the necessary corrections, then restart the HTTP server and the Web application server and try the logon process again.</p>																				

*Table 9. Troubleshooting ODWEK Installation. (continued)*

Problem	Solution
Error 500, Server Error	If the Logon screen appeared successfully, but you received an Error 500 message, then you are most likely running the Java servlet version of ODWEK. Verify that the ServerInit statement in the Servlet Support section of the httpd.conf file identifies the name and location of the was.conf file. In most installations, the was.conf file will be in the IBM HTTP server root directory. See your HTTP server information for directions.  Make the necessary corrections, then restart the HTTP server and the Web application server and try the logon process again.

## You next step

This section provides a road map to information that you may need after you have finished installing ODWEK. It includes a list of optional configuration tasks that are covered in this book, information for administrators who are responsible for distributing the ODWEK client software and who work with AFP fonts, information for programmers who need to integrate business applications with ODWEK, and some hints and tips on problem determination.

These sections provide information about optional configuration tasks:

- Appendix F, “Xenos transforms”, on page 167.
- Appendix G, “AFP to HTML transform”, on page 185.
- Appendix H, “AFP to PDF transform”, on page 189.

These sections provide information for administrators:

- Appendix E, “Java line data viewer”, on page 161.
- Appendix I, “Distributing user-defined files”, on page 193.
- Appendix J, “Mapping AFP fonts”, on page 199.

These sections provide information for programmers:

- Appendix A, “CGI API reference”, on page 85.
- Appendix B, “Java servlet reference”, on page 115.
- Appendix C, “Java API reference”, on page 117.
- Appendix D, “Java API programming guide”, on page 119.
- Appendix K, “No HTML output”, on page 201.

| For information on problem determination tools and hints and tips, see  
| Appendix L, "Problem determination tools", on page 207.

---

## Appendix A. CGI API reference

This chapter contains information about the programming functions that are available with ODWEK. This chapter is of primary interest to programmers responsible for integrating ODWEK with Web browsers.

**Note:** Parameter values are standard text. Is it possible that the text could consist of characters that will confuse browsers. To prevent possible errors, you must code all special characters in their corresponding hexadecimal codes. These special characters include control characters and certain alphanumeric symbols. For example, the string:

The post date is 12/31/95

would be converted to:

The%20post%20date%20is%2012%2f31%2f95

Parameter values include folder names, folder field names, and search criteria.

## Add Annotation

Add an annotation to the specified document

### Purpose

The Add Annotation function enables users to add an annotation to the specified document. To add an annotation, the user must be given permission to add annotations in the OnDemand application group.

**Note:** The Add Annotation function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

### Parameters

*Table 10. Add Annotation function*

Name=Value	Purpose
<code>_function=addnote</code>	Add an annotation.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid. The user must be given permission to add annotations to the OnDemand application group(s).
<code>_password=value</code>	The password for the user.
<code>_folder=value</code>	The name of the folder.
<code>_perm=value</code>	Determines whether the annotation is Public (0), Private (1), or Private for Group (2). Public annotations can be viewed by any user who has been given the View permission under Annotation in the application group. Private annotations can be viewed by the user that created the annotation, application group administrators, and system administrators. Private for Group annotations can be viewed by users in the specified group, application group administrators, and system administrators. The <code>_group</code> parameter contains the name of the group. The default value is 0 (Public).
<code>_group=groupName</code>	If the <code>_perm</code> parameter is set to 2 (Private for Group), names the group.
<code>_copy=value</code>	Determines whether the annotation should remain attached to the document if the document is exported to another server. The default value is off, meaning the annotation is not attached to the document. A value of on means the annotation is attached to the document if the document is exported to another server.
<code>_text=value</code>	The text of the annotation.

Table 10. Add Annotation function (continued)

Name=Value	Purpose
<code>_html=value</code>	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, ODWEK uses the ADDNOTE.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line:  <code>&lt;!-- -AOI# Marker--&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the add an annotation function.</p>
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_docid=documentID</code>	The identifier of the document to which the annotation is to be attached. The document identifier is returned by the Document Hit List function.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.

*Table 10. Add Annotation function (continued)*

Name=Value	Purpose
_logoff=1	Automatically disconnects the user from the OnDemand server after adding the annotation. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

### **Sample Function Call**

```
http://www.company.com/cgi-bin/arswww.cgi?_function=addnote  
&_server=myzos&_user=web&_password=web  
&_folder=credit%20card%20statements  
&_text=Test%20note%20from%20the%20OnDemand%20Internet%20Client  
&_docid=6850-6851-SUA17-1FAAA-225712-1634-132014-132172-89-76-11-25-0  
&_perm=1&_logoff=1
```

## Change Password

Change the OnDemand logon password

### Purpose

The Change Password function permits users to change their OnDemand passwords.

### Parameters

*Table 11. Change Password function*

Name=Value	Purpose
<code>_function=chgpassword</code>	Change the OnDemand password for the userid.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid.
<code>_password=value</code>	The password for the userid.
<code>_new_password=value</code>	The new password for the userid.
<code>_html=value</code>	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the CHGPASSWORD.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- - -AOI# Marker- - --&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the change password function.</p>
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.

Table 11. Change Password function (continued)

Name=Value	Purpose
_port= <i>value</i>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
_codepage= <i>value</i>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
_cgibin= <i>program</i>	Used by the CGI program when generating the next output page. If specified, then the page will contain a call to the specified program instead of the default program (ARSWWW.CGI). This parameter is primarily used by programmers who are creating a front-end CGI program or servlet to the CGI program or servlet provided by IBM.  The <i>program</i> can name a directory that is relative to the ServerRoot directive or name an <i>alias</i> that is defined in the Web server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.
_logoff=1	Automatically disconnects the user from the OnDemand server after changing the password. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

### Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=chgpassword
&_server=myzos&_user=web&_password=web
&_newpassword=newpw&_html=template.htm&_logoff=1
```

## Document Hit List

Display the list of documents that match the search criteria

### Purpose

The Document Hit List function displays the list of documents that match the search criteria for a specific folder. Each document is represented by a link to the document on the OnDemand server. After clicking on a link, ODWEK retrieves the document from the server and displays it in the browser window using the appropriate viewer.

### Parameters

Table 12. Document Hit List function

Name=Value	Purpose
<code>_function=dochitlist</code>	Display list of documents that match the search criteria.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid.
<code>_password=value</code>	The password for the userid.
<code>_folder=value</code>	The name of the folder.
<code>folder field name=value</code>	The name of a folder search field and the search value. You can specify one or more sets of field names and search values, up to the number of fields defined for the folder.
<code>folder field name2=value</code>	For folder search fields that use the BETWEEN or NOT BETWEEN search operators, the upper value with which to search the field.
<code>_display_fields=value[,value,...]</code>	A comma separated list that contains the names of the folder display fields. You can specify one or more field names. If you do not specify this parameter, the output page contains all of the folder display fields.
<code>_sort_field=value[,value,...]</code>	Determines the folder search field or fields that OnDemand uses to sort items in the document list. If you specify more than one field, separate the field names with a comma. For example: <code>_sort_field=Account,Account+Balance,Date</code> . The default sort fields are defined on the Field Information page for the folder.
<code>_sort_order=value[,value,...]</code>	For each folder search field that is specified in the <code>sort_field</code> parameter, determines whether OnDemand sorts items first to last or last to first. Specify an A (ascending) to sort items first to last. Specify any other character to sort items last to first (descending). For example: <code>_sort_order=A,D,A</code> . The default sort order is determined by the sort order that is defined on the Field Information page for the folder.

Table 12. Document Hit List function (continued)

Name=Value	Purpose
_max_hits=value	<p>Determines the maximum number of items that ODWEK returns to the document list, regardless of the number of items that match the query. ODWEK fills the document list with items that match a query in the order in which matching items were loaded into the database.</p> <p>ODWEK uses one of the following values to determine the number of items to return to the document list:</p> <ul style="list-style-type: none"> <li>• The value of the Maximum Hits field (specified on the Permissions page in the OnDemand folder). This value overrides all other values.</li> <li>• The value of the <code>_max_hits</code> parameter, if specified. This value overrides the MAXHITS parameter from the ARSWWW.INI file.</li> <li>• The value of the MAXHITS parameter, if specified.</li> <li>• If none of the above are specified, ODWEK returns a maximum of 200 items to the document list.</li> </ul>
_html=value	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the DOCHITLIST.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line:  <code>&lt;!-- - -AOI# Marker--&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the document hit list function.</p>
_frame=value	The output of this command will include a <code>target=value</code> attribute. This parameter makes building HTML frames simpler. This is an optional parameter.

Table 12. Document Hit List function (continued)

Name=Value	Purpose
<code>_datefmt=value</code>	Determines the format of date values used by ODWEK to search the database and display items that match a query. The default date format is set on the folder Field Information page. See the <i>IBM Content Manager OnDemand for Multiplatform Version 7.1 Administration Guide</i> , SC27-0840 for details about date formats that are supported by OnDemand.
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_sql=string</code>	<p>Specifies the SQL query that OnDemand uses to search the folder. If you specify this parameter, the SQL query is used to search the folder rather than any folder field name / value pairs that may be specified. OnDemand does not validate the query string.</p> <p>When using an SQL string, you must specify application group database field names and values. If you plan to query date fields, you must specify OnDemand internal date values. For example, the date January 1, 1999 would be specified as 10593. You can use the ARSDATE program to list the internal date value for a given date.</p> <p>The SQL string is used to search all of the application groups contained in the folder. If the SQL string contains a database field name that is in one application group but not in another application group, then the query will fail.</p>
<code>_date1=value</code>	Use to specify the beginning date in a range of dates to search. If you specify the <code>_date1</code> and <code>_date2</code> parameters, OnDemand limits the query to the table or tables that contain one or both of the specified dates. The format of the date string that you specify must match the display format of the folder field. (You can use the administrative client to list the display format of the folder field.)

Table 12. Document Hit List function (continued)

Name=Value	Purpose
<code>_date2=value</code>	Use to specify the ending date in a range of dates to search. If you specify the <code>_date1</code> and <code>_date2</code> parameters, OnDemand limits the query to the table or tables that contain one or both of the specified dates. The format of the date string that you specify must match the display format of the folder field. (You can use the administrative client to list the display format of the folder field.)
<code>_cgibin=program</code>	Used by the CGI program when generating the next output page. If specified, then the page will contain a call to the specified program instead of the default program (ARSWWW.CGI). This parameter is primarily used by programmers who are creating a front-end CGI program or servlet to the CGI program or servlet provided by IBM. The <i>program</i> can name a directory that is relative to the ServerRoot directive or name an <i>alias</i> that is defined in the Web server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.
<code>_or=value</code>	Specify a 1 (one) to connect search fields using the OR logical operator; an item must match at least one of the specified search values. The default value is 0 (zero), which means that OnDemand connects search fields with the AND logical operator (an item must match all of the specified search values).
<code>_logoff=1</code>	Automatically disconnects the user from the OnDemand server after creating the document list. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

## Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=dochitlist
&_server=myzos&_user=web&_password=web
&_folder=credit%20card%20statements
&account%20number=1000100010009999&date=1%2f1%2f96&date2=12%2f31%2f96
&_sort_field=Account,Account%20Balance,Date&_sort_order=A,D,A
&_logoff=1
&_html=template.htm
```

## Logoff

Logoff an OnDemand server

### Purpose

The Logoff function attempts to log a user off an OnDemand server. The name of the server and the userid to log off are stored in a browser cookie on the client by the Logon function. If the server is not a valid OnDemand server, an error message is returned. If the userid is not logged on to the specified server, an error message is returned.

### Parameters

Table 13. Logoff function

Name=Value	Purpose
_function=logoff	Log off an OnDemand server.
_html=value	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the LOGOFF.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line:   &lt;!-- -AOI# Marker--&gt; The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the logoff function.</p>
_nohtml=value	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.

*Table 13. Logoff function (continued)*

Name=Value	Purpose
_port=value	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.

### **Sample Function Call**

```
http://www.company.com/cgi-bin/arswww.cgi?_function=logoff  
&_html=template.htm
```

## Logon

Logon to an OnDemand server

### Purpose

The Logon function attempts to access an OnDemand server using the values of the server, user, and password parameters. The Logon function verifies that the specified user is authorized to logon to the specified server and verifies the password. If the user is not authorized to logon to the server, an error message is returned. If the server is not a valid OnDemand server, an error message is returned. If the password is not valid for the user, an error message is returned. After a successful logon, the Logon function displays a Web page that contains a list of the folders that the user is authorized to access.

### Parameters

Table 14. Logon function

Name=Value	Purpose
<code>_function=logon</code>	Logon to an OnDemand server.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid.
<code>_password=value</code>	The password for the userid.
<code>_new_password=value</code>	The new password for the userid. Allows the password to be changed after successfully logging on to the OnDemand server. This is an optional parameter.
<code>_html=value</code>	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the LOGON.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- - -AOI# Marker- - --&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the logon function.</p>

Table 14. Logon function (continued)

Name=Value	Purpose
<code>_frame=value</code>	The output of this command will include a <code>target=value</code> attribute. This parameter makes building HTML frames simpler. This is an optional parameter.
<code>_datefmt=value</code>	Determines the format of date values used by ODWEK to search the database and display items that match a query. The default date format is set on the folder Field Information page. See the <i>IBM Content Manager OnDemand for Multiplatforms Version 7.1 Administration Guide</i> , SC27-0840 for details about date formats supported by OnDemand.
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_cgibin=program</code>	Used by the CGI program when generating the next output page. If specified, then the page will contain a call to the specified program instead of the default program (ARSWWW.CGI). This parameter is primarily used by programmers who are creating a front-end CGI program or servlet to the CGI program or servlet provided by IBM.  The <code>program</code> can name a directory that is relative to the ServerRoot directive or name an <i>alias</i> that is defined in the Web server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.

## Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=logon
&_server=myzos&_user=web&_password=web
&_html=template.htm
```

---

## Print Document (Server)

Sends one or more documents to the specified server printer

### Purpose

The Print Document function sends copies of documents to an OnDemand server printer. To use the server print facility, the user must have the Print permission under Document in the OnDemand application group. At least one server printer must be defined on the specified OnDemand server.

### Parameters

*Table 15. Print Document function*

Name=Value	Purpose
<code>_function=printdocs</code>	Print documents.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid. The user must have the Print permission under Document in the application group.
<code>_password=value</code>	The password for the user.
<code>_folder=value</code>	The name of the folder.

Table 15. Print Document function (continued)

Name=Value	Purpose
<code>_printer=value</code>	<p>The name of the OnDemand server printer.</p> <p>When the specified printer is a FAX or a Printer with Information, then you can specify the following additional parameters:</p> <ul style="list-style-type: none"> <li><code>_recv_name=value</code> The receiver's name.</li> <li><code>_recv_comp=value</code> The receiver's company name.</li> <li><code>_recv_fax=value</code> The receiver's fax number.</li> <li><code>_send_name=value</code> The sender's name.</li> <li><code>_send_comp=value</code> The sender's company name.</li> <li><code>_send_tel=value</code> The sender's telephone number.</li> <li><code>_send_fax=value</code> The sender's fax number.</li> <li><code>_send_cover=value</code> A user-defined overlay that the Header Page Exit program merges with the values of the other parameters to produce a cover page for the document.</li> <li><code>_subject=value</code> A string that represents the subject of the document.</li> <li><code>_notes=value</code> A string that represents a note about the document.</li> </ul>

Table 15. Print Document function (continued)

Name=Value	Purpose
<code>_html=value</code>	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the PRINTDOCS.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line:  <code>&lt;!-- -AOI# Marker--&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the print documents function.</p>
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_docids=documentIDList</code>	A list of document identifiers for the documents to be printed. The document identifiers are returned by the Document Hit List function. If you specify more than one document identifier, then you must separate the document identifiers with the \003 character. <b>Note:</b> If the number of document identifiers exceeds 200, then you must specify the <code>_max_hits</code> parameter.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.

Table 15. Print Document function (continued)

Name=Value	Purpose
_max_hits=value	<p>Use this parameter to specify the number document identifiers to process. Specify a value that is equal to or greater than the number of document identifiers specified with the <code>_docids</code> parameter.</p> <p><b>Note:</b> If the number of document identifiers exceeds the value specified by the MAXHITS parameter in the ARSWWW.CGI file (or 200, if not specified), then you must specify the <code>_max_hits</code> parameter. If you do not specify the <code>_max_hits</code> parameter (or you do not specify a value for the MAXHITS parameter), a maximum of 200 document identifiers will be processed, regardless of the number of document identifiers that you specified with the <code>_docids</code> parameter.</p> <p>ODWEK uses one of the following values to determine the number of document identifiers to process:</p> <ul style="list-style-type: none"> <li>The value of the <code>_max_hits</code> parameter, if specified. This value overrides the value of the MAXHITS parameter.</li> <li>The value of the MAXHITS parameter, if specified.</li> <li>If none of the above are specified, ODWEK processes a maximum of 200 document identifiers.</li> </ul>
_logoff=1	Automatically disconnects the user from the OnDemand server after printing the document. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

### Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=printdocs
&_server=myzos&_user=web&_password=web
&_folder=credit%20card%20statements
&_printer=infoprint60
&_docids=6850-6851-SUA17-1FAAA-225712-1634-132014-132172-89-76-11-25-0
&_logoff=1
```

## Retrieve Document

Retrieves the selected document from OnDemand

### Purpose

The Retrieve Document function retrieves the selected document from the OnDemand server. ODWEK displays the document in the browser window using the applet, plug-in, or other program that is associated with the document type.

### Parameters

*Table 16. Retrieve Document function*

Name=Value	Purpose
<code>_function=retrieve</code>	Retrieve the selected document.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid.
<code>_password=value</code>	The password for the userid.
<code>_folder=value</code>	The name of the folder.
<code>folder field name=value</code>	The name of a folder search field and the search value. You can specify one or more sets of field names and search values, up to the number of fields defined for the folder.
<code>_html=value</code>	<p>When an error occurs retrieving a document, determines the HTML file that ODWEK uses as a template to generate the (error) output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the RETRIEVE.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- - -AOI# Marker--&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the retrieve function.</p>

Table 16. Retrieve Document function (continued)

Name=Value	Purpose
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, “No HTML output”, on page 201 for details about the delimited ASCII output.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_cgibin=program</code>	Used by the CGI program when generating the next output page. If specified, then the page will contain a call to the specified program instead of the default program (ARSWWW.CGI). This parameter is primarily used by programmers who are creating a front-end CGI program or servlet to the CGI program or servlet that is provided by IBM.  The <i>program</i> can name a directory that is relative to the ServerRoot directive or name an <i>alias</i> that is defined in the Web server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.
<code>_or=value</code>	Specify a 1 (one) to connect search fields using the OR logical operator; an item must match at least one of the specified search values. The default value is 0 (zero), which means that OnDemand connects search fields with the AND logical operator (an item must match all of the specified search values).

Table 16. Retrieve Document function (continued)

Name=Value	Purpose
<code>_afp=value</code>	<p>When you retrieve an AFP document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer. For example, some customers convert AFP documents to HTML with the AFP2WEB Transform and use the AFP2HTML applet to view the HTML output. Those customers should specify <code>_afp=HTML</code> so that ODWEK will convert the AFP document before sending it to the viewer.</p> <p>The <i>value</i> can be:</p> <ul style="list-style-type: none"> <li><b>ASCII</b>      ODWEK converts the AFP document to ASCII text.</li> <li><b>HTML</b>      ODWEK converts the AFP document to HTML with the AFP2WEB Transform.</li> <li><b>NATIVE</b>    ODWEK extracts and uncompresses the AFP document and its resources from OnDemand. <b>Note:</b> If you specify <code>_afp=NATIVE</code>, verify that the MIME content type identifies the viewer that you want to use (see “[MIMETYPES]” on page 50 for more information).</li> <li><b>PDF</b>       ODWEK converts the AFP document to PDF with the AFP2WEB Transform.</li> <li><b>PLUGIN</b>   ODWEK does not convert the AFP document (the default).</li> </ul>
<code>_email=value</code>	<p>When you retrieve an EMAIL document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer. The <i>value</i> can be:</p> <ul style="list-style-type: none"> <li><b>NATIVE</b>    ODWEK extracts and uncompresses the EMAIL document from OnDemand. <b>Note:</b> If you specify <code>_email=NATIVE</code>, verify that the MIME content type identifies the viewer that you want to use (see “[MIMETYPES]” on page 50 for more information).</li> <li><b>HTML</b>       ODWEK converts the EMAIL document to HTML.</li> </ul>

Table 16. Retrieve Document function (continued)

Name=Value	Purpose						
_line=value	<p>When you retrieve a line data document from the OnDemand server, the value of this parameter determines what action, if any, ODWEK takes before sending the document to the viewer. The <i>value</i> can be:</p> <table> <tr> <td><b>APPLET</b></td><td>ODWEK converts the line data document for viewing with the Line Data applet (the default).</td></tr> <tr> <td><b>ASCII</b></td><td>ODWEK converts the line data document to ASCII text.</td></tr> <tr> <td><b>NATIVE</b></td><td>ODWEK extracts and uncompresses the line data document from OnDemand. <b>Note:</b> If you specify _line=NATIVE, verify that the MIME content type identifies the viewer that you want to use (see “[MIMETYPES]” on page 50 for more information).</td></tr> </table>	<b>APPLET</b>	ODWEK converts the line data document for viewing with the Line Data applet (the default).	<b>ASCII</b>	ODWEK converts the line data document to ASCII text.	<b>NATIVE</b>	ODWEK extracts and uncompresses the line data document from OnDemand. <b>Note:</b> If you specify _line=NATIVE, verify that the MIME content type identifies the viewer that you want to use (see “[MIMETYPES]” on page 50 for more information).
<b>APPLET</b>	ODWEK converts the line data document for viewing with the Line Data applet (the default).						
<b>ASCII</b>	ODWEK converts the line data document to ASCII text.						
<b>NATIVE</b>	ODWEK extracts and uncompresses the line data document from OnDemand. <b>Note:</b> If you specify _line=NATIVE, verify that the MIME content type identifies the viewer that you want to use (see “[MIMETYPES]” on page 50 for more information).						
_docid=documentID	The identifier of the document to be retrieved. The document identifier is returned by the Document Hit List function.						
_logoff=1	Automatically disconnects the user from the OnDemand server after retrieving the document. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).						

### Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=retrieve
&_server=myzos&_user=web&_password=web
&_folder=credit%20card%20statements
&account%20number=1000100010009999&date=1%2f1%2f96
&_html=template.htm&_logoff=1
```

## Search Criteria

Display search criteria for a specific folder

### Purpose

The Search Criteria function displays the search criteria for a specific folder using a form. The user can accept the default search criteria or enter search criteria to search for a specific document. After clicking the Submit button, ODWEK displays a Web page that lists the documents that match the search criteria.

### Parameters

Table 17. Search Criteria function

Name=Value	Purpose
<code>_function=searchcrit</code>	Display search criteria for a specific folder.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid.
<code>_password=value</code>	The password for the userid.
<code>_folder=value</code>	The name of the folder to search.
<code>_html=value</code>	<p>Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the SEARCHCRIT.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR variable.</p> <p>The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- - -AOI# Marker--&gt;</code></p> <p>The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.</p> <p>The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the search criteria function.</p>
<code>_frame=value</code>	The output of this command will include a <code>target=value</code> attribute. This parameter makes building HTML frames simpler. This is an optional parameter.

Table 17. Search Criteria function (continued)

Name=Value	Purpose
<code>_datefmt=value</code>	Determines the format of date values used by ODWEK to search the database and display items that match a query. The default date format is set on the folder Field Information page. See the <i>IBM Content Manager OnDemand for Multiplatforms Version 7.1 Administration Guide</i> , SC27-0840 for details about date formats supported by OnDemand.
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_cgibin=program</code>	Used by the CGI program when generating the next output page. If specified, then the page will contain a call to the specified program instead of the default program (ARSWWW.CGI). This parameter is primarily used by programmers who are creating a front-end CGI program or servlet to the CGI program or servlet provided by IBM.  The <i>program</i> can name a directory that is relative to the ServerRoot directive or name an <i>alias</i> that is defined in the Web server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.
<code>_logoff=1</code>	Automatically disconnects the user from the OnDemand server after displaying the search criteria. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

## Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=searchcrit  
&_server=myzos&_user=web&_password=web  
&_folder=credit%20card%20statements&_html=template.htm  
&_logoff=1
```

## Update Document

Updates one or more database values for the specified document

### Purpose

The Update Document function allows authorized users to update documents. The Update Document function updates one or more database values for a specific document.

**Note:** The Update Document function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

### Parameters

*Table 18. Update Document function*

Name=Value	Purpose
<code>_function=updatedoc</code>	Update the database.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid. The user must have the Update permission under Document in the application group.
<code>_password=value</code>	The password for the user.
<code>_folder=value</code>	The name of the folder.
<code>folder field name=value</code>	The name of the field that you want to update and the value that you want put in the field. You can specify one or more sets of field names and values, up to the number of fields defined for the folder.
<code>_html=value</code>	Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the UPDATE.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.  The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- -AOI# Marker--&gt;</code> The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.  The TEMPLATE.HTM file is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the update function.

Table 18. Update Document function (continued)

Name=Value	Purpose
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_docid=documentID</code>	The identifier of the document to be updated. The document identifier is returned by the Document Hit List function.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_logoff=1</code>	Automatically disconnects the user from the OnDemand server after updating the document. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

### Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=updatedoc
&_server=myzos&_user=web&_password=web
&_folder=credit%20card%20statements
&account%20number=1000100010009999
&_docid=6850-6851-SUA17-1FAAA-225712-1634-132014-132172-89-76-11-25-0
&_html=template.htm&_logoff=1
```

## View Annotations

View annotations attached to the specified document

### Purpose

The View Annotations function enables users to view the annotations attached to the specified document. To view annotations, the user must have the View permission under Annotation in the OnDemand application group.

**Note:** The View Annotations function is not supported when accessing a Content Manager OnDemand for OS/390 V2.1 server with ODWEK.

### Parameters

*Table 19. View Annotations function*

Name=Value	Purpose
<code>_function=getnotes</code>	View annotations.
<code>_server=value</code>	The name of the OnDemand server.
<code>_user=value</code>	The OnDemand userid. The user must be have the View permission under Annotation in the application group(s).
<code>_password=value</code>	The password for the user.
<code>_folder=value</code>	The name of the folder.
<code>html=value</code>	Determines the HTML file that ODWEK uses as a template to generate the output Web page. The value can be a file name or an * (asterisk). If the value is an asterisk, then ODWEK uses the GETNOTES.HTML file found in the directory named by the TEMPLATEDIR parameter in the ARSWWW.INI file. If the value is a file name without a path name, then the file must be located in the directory named by the TEMPLATEDIR parameter. If the value includes a path name, then the path should be relative to the directory named by the TEMPLATEDIR parameter.  The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <code>&lt;!-- - -AOI# Marker--&gt;</code> The location of the comment line determines where ODWEK places its output. All lines above the comment line will be written before the output generated by ODWEK. All lines below the comment line will be written after the output generated by ODWEK.  The TEMPLATE.HTM is the sample template file provided with ODWEK. You can use the sample template file to help create your own template file for the view annotations function.

Table 19. View Annotations function (continued)

Name=Value	Purpose
<code>_nohtml=value</code>	Determines the type of output generated by ODWEK. The default value is 0 (zero) and means that ODWEK generates HTML output. If you specify 1 (one), then ODWEK generates delimited ASCII output. See Appendix K, "No HTML output", on page 201 for details about the delimited ASCII output.
<code>_docid=documentID</code>	The identifier of the document that contains the annotations to be viewed. The document identifier is returned by the Document Hit List function.
<code>_port=value</code>	The TCP/IP port number that ODWEK and the OnDemand library server use to communicate. The default value, 0 (zero), means that the port number that is specified for the OnDemand service in the SERVICES file is used. If the OnDemand service is not specified in the SERVICES file, then port number 1445 will be used. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file. Before using any port number, verify with your TCP/IP administrator that the port is available.
<code>_codepage=value</code>	The code page of the OnDemand database. The default code page is the code page of the Web server. If the code page of the server is different than the code page of the database, then you must specify the code page. Any value that you specify overrides the value of the CODEPAGE parameter in the ARSWWW.INI file.
<code>_logoff=1</code>	Automatically disconnects the user from the OnDemand server after viewing the annotation. Specifying this parameter eliminates the need for your application to call the Logoff function to disconnect the user. The only valid value for this parameter is 1 (one).

## Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=getnotes
&_server=myzos&_user=web&_password=web
&_folder=credit%20card%20statements
&_docid=6850-6851-SUA17-1FAAA-225712-1634-132014-132172-89-76-11-25-0
&_logoff=1
```



---

## Appendix B. Java servlet reference

The Java servlet acts as a controller of your Web application, performing functions and common tasks before and after an action, such as management of the connection to the OnDemand server.

Functions are provided for typical application tasks:

- log on and log off
- search
- retrieve, print, and update documents
- add and view annotations
- change password

You use a set of application functions and parameters to use the servlet in your application.

The Java servlet uses the same functions as the CGI program. See Appendix A, “CGI API reference”, on page 85 for a reference of the functions, descriptions, and parameters.



---

## Appendix C. Java API reference

The documentation for the Java API is provided in HTML format with the ODWEK software.

Before you can view the documentation, you must install ODWEK software on the system and then extract the documentation files from the `ODApiDoc.zip` file in the `install/api` directory. (Where `install` is the installation directory for ODWEK software.) Use an extraction method that preserves the directory structure of the files in the archive.

To view the documentation, after extracting the files, open the `index.html` file with a Web browser.



## Appendix D. Java API programming guide

The Java application programming interfaces (APIs) are a set of classes that access and manipulate data on an OnDemand server. This section describes the Java APIs, the Java implementation of document functions, and Internet connectivity.

The Java APIs support:

- A common object model for data access
- Search and update across OnDemand servers. **Note:** See Chapter 1, "Product overview", on page 1 for limitations when accessing an OnDemand for OS/390 Version 2 server.
- Client/server implementation for Java application users

### Client/server architecture

The APIs provide a convenient programming interface for application users. APIs can reside on both the OnDemand server and the client (both provide the same interface), and the applications can be located locally or remotely. The client API communicates with the server to access data through the network. Communication between the client and the server is performed by classes; it is not necessary to add any additional programs.

The API classes consist of one package: `com.ibm.edms.od`.

### Packaging for the Java environment

The API classes are contained in one package: `com.ibm.edms.od`. The classes are:

#### **com.ibm.edms.od.ODCallback**

This class is used with all methods in which the server operation returns data while processing.

#### **com.ibm.edms.od.ODCriteria**

A class that represents the search criteria from an OnDemand folder. The criteria class contains methods to set a search operator and search values.

#### **com.ibm.edms.od.ODError**

This class represents the exceptions which may occur when using the APIs.

**com.ibm.edms.od.ODFolder**

A class that represents an OnDemand folder. This object is returned from a successful call to `ODServer.openFolder()`. This class contains folder criteria information. These criteria objects are what need to be modified in order to narrow the query on the server.

**com.ibm.edms.od.ODHit**

This class represents an OnDemand document.

**com.ibm.edms.od.ODNote**

This class represents an OnDemand annotation.

**com.ibm.edms.od.ODServer**

This class represents a connection to an OnDemand server. From this class you can logon, logoff and change the password. After a successful logon, this object will contain a list of all folders that the session has access to. **Note:** Access to this server object should be done in a single threaded environment. The only exception is when cancelling a server operation.

## Programming tips

You must import the `com.ibm.edms.od` package into your ODWEK application.

You do not need an HTTP server or a Web application server to run ODWEK applications that use the Java API. You can run the Java interpreter on ODWEK applications.

To run the Java interpreter on an ODWEK application:

1. Copy the `arswww.ini` file to a user-defined run time directory.
2. Copy the shared library to the directory in which you copied the `arswww.ini` file:

*Table 20. Shared Library Filename*

Operating System	Shared Library
AIX	<code>libarswwwsl.so</code>
HP-UX	<code>libarswwwsl.so</code>
Linux	<code>libarswwwsl.so</code>
Solaris	<code>libarswwwsl.sl</code>
Windows	<code>arswwwsl.dll</code>

3. For Windows systems, copy these files to the directory in which you copied the `arswww.ini` file:

`ARSSCKNT.DLL`  
`ARSCT32.DLL`

4. Specify the name of the user-defined directory when you run the Java interpreter on the application. See “Running an ODWEK application” on page 124 for an example.

---

## Setting up the system environment

When you set up your AIX, HP-UX, Linux, Solaris, or Windows environment, you must establish the following settings:

### package

Import for all ODWEK applications.

- com.ibm.edms.od

### Library files

**Shared objects for AIX, HP-UX, Linux, and Solaris**

**DLLs for Windows**

## Setting environment variables

When developing an ODWEK application, you must set up your environment.

### AIX

In the AIX environment, you must set the following environment variables to set up your development environment for developing ODWEK applications.

**PATH** Make sure your PATH contains /usr/lpp/ars/www

**LIBPATH** Make sure your LIBPATH contains /usr/lpp/ars/www

### LD\_LIBRARY\_PATH

Make sure your LD\_LIBRARY\_PATH contains /usr/lpp/ars/www

**CLASSPATH** Make sure your CLASSPATH contains /usr/lpp/ars/www/api/ODApi.jar , which is the class library.

### HP-UX

In the HP-UX environment, you must set the following environment variables to set up your development environment for developing ODWEK applications.

**PATH** Make sure your PATH contains /opt/ondemand/www

**LIBPATH** Make sure your LIBPATH contains /opt/ondemand/www

### LD\_LIBRARY\_PATH

Make sure your LD\_LIBRARY\_PATH contains /opt/ondemand/www

**CLASSPATH** Make sure your CLASSPATH contains /opt/ondemand/www/api/ODApi.jar , which is the class library.

## **Linux**

In the Linux environment, you must set the following environment variables to set up your development environment for developing ODWEK applications.

**PATH** Make sure your PATH contains /opt/ondemand/www

**LIBPATH** Make sure your LIBPATH contains /opt/ondemand/www

### **LD\_LIBRARY\_PATH**

Make sure your LD\_LIBRARY\_PATH contains /opt/ondemand/www

**CLASSPATH** Make sure your CLASSPATH contains /opt/ondemand/www/api/ODApi.jar , which is the class library.

## **Solaris**

In the Solaris environment, you must set the following environment variables to set up your development environment for developing ODWEK applications.

**PATH** Make sure your PATH contains /opt/ondemand/www

**LIBPATH** Make sure your LIBPATH contains /opt/ondemand/www

### **LD\_LIBRARY\_PATH**

Make sure your LD\_LIBRARY\_PATH contains /opt/ondemand/www

**CLASSPATH** Make sure your CLASSPATH contains /opt/ondemand/www/api/ODApi.jar , which is the class library.

## **Windows**

In the Windows environment, you must set the following environment variables to set up your development environment for developing ODWEK applications.

**PATH** Make sure your PATH contains x :\yyyyyyy \DLL; where x is the drive on which you installed ODWEK and yyyyyyy is the installation directory for the ODWEK software.

**CLASSPATH** Make sure your CLASSPATH contains x :\yyyyyyy \WWW\API\ODApi.jar, where x is the drive on which you installed ODWEK and yyyyyyy is the installation directory for the class library.

---

## **Tracing and diagnostic information**

To handle problems that arise in your Java API applications, you can use tracing and exception handling.

### **Tracing**

The following parameters in the ARSWWW.INI file write trace information to the arswww.log file in the specified directory:

```
[DEBUG]  
LOG=1  
LOGDIR=/ars/www/log
```

**Note:** Because a significant amount of information can be written to a log file, IBM recommends that you enable logging only when needed, such as when recreating a problem. If you need to enable logging for extended periods of time, make sure that the log file paths point to storage devices with plenty of free space. Remember to periodically delete old log files from the system.

See Appendix L, “Problem determination tools”, on page 207 for information about other tools that you can use to gather information about the system and documents.

## Exception handling

When the Java APIs encounter a problem, they throw an exception. Throwing an exception creates an exception object of `ODError` class or one of its subclasses.

When a `ODError` is created, the API logs diagnostic information into a log file, assuming that logging is enabled. See “Tracing” on page 122 for more information about the log file used by the Java APIs.

When a `ODError` is caught, it allows you to see any error messages, error codes, and error states that occurred while running. When an error is caught, an error message is issued along with the location of where the exception was thrown. The error ID and exception ID are also given. The code below shows an example of the throw and catch process:

```
try  
{  
    odServer = new ODServer( );  
    odServer.initialize( argv[9], "TcUpdate.java" );  
    System.out.println( "Logging on to " + argv[0] + "..." );  
    odServer.logon( argv[0], argv[1], argv[2] );  
    odServer.logoff( );  
    odServer.terminate( );  
}  
  
catch ( ODError e )  
{  
    System.out.println( "ODError: " + e );  
    System.out.println( "    id = " + e.getErrorId( ) );  
    System.out.println( "    msg = " + e.getErrorMsg( ) );  
    e.printStackTrace( );  
}
```

## Constants

The constants provided for use with the Java APIs are described in an online reference. See Appendix C, “Java API reference”, on page 117 for more information.

## Running an ODWEK application

You can use the Java interpreter to run an ODWEK application. Please keep the following points in mind when creating, compiling and running an ODWEK application:

1. Create your ODWEK application by using the methods that are available to you in the Java API. Import the Java API package in your ODWEK application file. For example:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class Logon
{
    public static void main ( String argv[] )
    {
        .
        .
        .
    }
}
```

2. Compile your ODWEK application file ( .java ) with javac to produce the .class file. See your Java reference publication for instructions on compiling Java applications.
3. Run the Java interpreter on your application (.class file). For example:

```
java Logon server userid passwd /tmp/ondemand/www
```

Where Logon is the name of the .class file, server, userid, and passwd are parameters for the application, and /tmp/ondemand/www is the user-defined run time directory that contains a copy of the arswww.ini file. **Note:** This example assumes that you have specified the path to the ODWEK class and servlet libraries by using system environment variables (see “Setting up the system environment” on page 121).

## Connecting to an OnDemand server

An object of the class ODServer represents and manages a connection to an OnDemand server, provides transaction support, and runs server commands. Appendix C, “Java API reference”, on page 117 explains where to find the online reference of methods and their descriptions.

When connecting to an OnDemand server, you must be aware of the requirements for the server; for example, the password for OnDemand can be no more than eight characters in length.

## Establishing a connection

The ODServer class provides methods for connecting to an OnDemand server and disconnecting from the server. The following example uses an OnDemand library server named LIBSRVR1 , the userid ADMIN and password PASSWD . The example creates an ODServer object for the OnDemand server, connects to it, works with it (not specified in the example), and then disconnects from it.

```
odServer = new ODServer( );
System.out.println( "Logging on to " + "LIBSRVR1" + "..." );
odServer.logon( "LIBSRVR1", "ADMIN", "PASSWD" );
.
.
.
odServer.logoff( );
odServer.terminate( );
```

See “Working with an OnDemand server” for the complete sample application from which this example was taken.

## Setting and getting passwords

You can access or set a user’s password on an OnDemand server by using the methods in ODServer. The following example shows how to set and get a user’s password on an OnDemand library server.

```
odServer = new ODServer( );
odServer.setServer( "LIBSRVR1" );
odServer.setUserId( "ADMIN" );
odServer.setPassword( "PASSWD" );

System.out.println( "Logging on to " + "LIBSRVR1" + "..." );

odServer.logon( odServer.getServerName( ),
                odServer.getUserId( ),
                odServer.getPassword( ),
                ODConstant.CONNECT_TYPE_LOCAL,
                0 );
```

See “Working with an OnDemand server” for the complete sample application from which this example was taken.

---

## Working with an OnDemand server

An object of the class ODServer represents and manages a connection to an OnDemand server, provides transaction support, and runs server commands.

The following example uses ODServer methods to prepare for logon, set the application name, (optionally) display the local directory, display the server

name, userid and password, display and set the connection type, display and set the port, and disconnect from the server.

This example demonstrates these ODServer methods:

- initialize
- logon
- logoff
- terminate
- getConnectType
- getLocalDir
- getPassword
- getPort
- getServerName
- getUserId
- setApplicationName
- setConnectType
- setLocalDir
- setPassword
- setPort
- setServer
- setUserId

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

Example of working with an OnDemand server:

```
*****  
import java.util.*;  
import java.io.*;  
import com.ibm.edms.od.*;  
  
public class TcServerMisc  
{  
    public static void main ( String argv[] )  
    {  
        ODServer odServer;  
        String str;  
        int j;  
  
        //-----  
        // If too few parameters, display syntax and get out  
        //-----  
        if ( argv.length < 4 )  
        {  
            System.out.println( "usage: java TcServerMisc <server> <userid> <password> <config dir> [<local server dir>]" );  
            return;  
        }  
  
        try  
        {  
            //-----  
            // Set the stage  
            //-----  
            System.out.println( "This testcase should:" );  
        }
```

```

System.out.println( " Use ODServer methods setServer, setId, and setPassword" );
System.out.println( " to prepare for logon" );
System.out.println( " Set the application name" );
System.out.println( " Display the" );
System.out.println( " Local Directory" );
System.out.println( " Server name" );
System.out.println( " User Id" );
System.out.println( " Password" );
System.out.println( " Connect Type" );
System.out.println( " Set and display the port" );
System.out.println( " Set the connect type" );
System.out.println( " Logoff" );
System.out.println( "" );
System.out.println( "Ensure that all information is correct." );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );

//-----
// Logon to specified server
//-----
odServer = new ODServer( );
odServer.initialize( argv[3], "TcServerMisc.java" );
odServer.setServer( argv[0] );
odServer.setId( argv[1] );
odServer.setPassword( argv[2] );

System.out.println( "Logging on to " + argv[0] + "..." );
if ( argv.length == 4 )
    odServer.logon( );
else
{
    if ( argv.length == 5 )
    {
        odServer.setLocalDir( argv[4] );
        odServer.logon( odServer.getServerName( ),
                        odServer.getId( ),
                        odServer.getPassword( ),
                        ODConstant.CONNECT_TYPE_LOCAL,
                        0,
                        odServer.getLocalDir( ) );
    }
}

//-----
// Test miscelaneous methods
//-----
System.out.println( "Setting application name to TcServerMisc.java..." );
odServer.setApplicationName( "TcServerMisc.java" );

System.out.println( "Local Dir: " + odServer.getLocalDir( ) );
System.out.println( "Server Name: " + odServer.getServerName( ) );
System.out.println( "User Id: " + odServer.getId( ) );
System.out.println( "Password: " + odServer.getPassword( ) );
System.out.println( "Connect Type: " + getConnectTypeName( odServer.getConnectType( ) ) );

j = odServer.getPort( );
System.out.println( "Setting port to " + j + "..." );
odServer.setPort( j );
System.out.println( "Port: " + j );

if ( argv.length == 4 )
{
    System.out.println( "Setting connect type to ODConstant.CONNECT_TYPE_TCPIP..." );
    odServer.setConnectType( ODConstant.CONNECT_TYPE_TCPIP );
}
else
{
    System.out.println( "Setting connect type to ODConstant.CONNECT_TYPE_LOCAL..." );
    odServer.setConnectType( ODConstant.CONNECT_TYPE_LOCAL );
}

//-----
// Cleanup
//-----
System.out.println( "Logging off..." );
odServer.logoff( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - analyze if required" );

```

```

        System.out.println( "" );
    }

    catch ( ODError e )
    {
        System.out.println( "ODError: " + e );
        System.out.println( "    id = " + e.getErrorCode( ) );
        System.out.println( "    msg = " + e.getErrorMessage( ) );
        e.printStackTrace( );
    }

    catch ( Exception e2 )
    {
        System.out.println( "exception: " + e2 );
        e2.printStackTrace( );
    }
}

static String getConnectTypeName( char type )
{
    String str;

    switch( type )
    {
        case ODConstant.CONNECT_TYPE_TCPIP:
            str = "TCPIP";
            break;
        case ODConstant.CONNECT_TYPE_LOCAL:
            str = "LOCAL";
            break;
        default:
            str = "*** Unknown connect type";
            break;
    }

    return str;
}
}

```

---

## Listing application groups in a folder

An object of the class `ODFolder` represents an OnDemand folder.

The following example uses `ODFolder` methods to display the number of application groups that can be searched from the folder and display the name of each application group.

This example demonstrates these `ODFolder` methods:

- `getNumAppGroups`
- `getApplGroups`
- `close`

This example also uses `ODServer` methods to prepare for logon, open the specified folder, and log off. This example demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `openFolder`
- `logoff`
- `terminate`

This example uses these run-time parameters:

- Server name

- User Id
- Password
- Folder name
- Configuration directory (location of arswww.ini file)
- (optional) Local server directory

Example of listing the application groups in a folder:

```
/*
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcApplGrp
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        Object[] appl_grps;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcApplGrp <server> <userid> <password> <folder> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Display the folder name" );
            System.out.println( " Display the number of application groups" );
            System.out.println( " Display the name of each application group" );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );

            //-----
            // Logon to the specified server
            //-----
            odServer = new ODServer( );
            odServer.initialize( argv[4], "TcListCriteria.java" );

            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 5 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                if ( argv.length == 6 )
                    odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

            //-----
            // Open the specified folder
            //-----
            System.out.println( "Opening " + argv[3] + " folder..." );
            odFolder = odServer.openFolder( argv[3] );

            //-----
            // Display number and names of application groups
            //-----
            System.out.println( "There is(are) " + odFolder.getNumApplGroups( ) + " application group(s) in the folder:" );
            appl_grps = odFolder.getApplGroups( );
            for ( j = 0; j < appl_grps.length; j++ )
                System.out.println( " " + appl_grps[j].toString( ) );

            //-----
            // Cleanup
            //-----
            odFolder.close( );
            odServer.logoff( );
            odServer.terminate( );
            System.out.println( "" );
            System.out.println( "-----" );
        }
    }
}
```

```

        System.out.println( "" );
        System.out.println( "Testcase completed - analyze results if required" );
        System.out.println( "" );
    }

    catch ( ODError e )
    {
        System.out.println( "ODError: " + e );
        System.out.println( " id = " + e.getErrorCode( ) );
        System.out.println( " msg = " + e.getErrorMessage( ) );
        e.printStackTrace( );
    }

    catch ( Exception e2 )
    {
        System.out.println( "exception: " + e2 );
        e2.printStackTrace( );
    }
}

```

---

## Searching a folder

An object of the class `ODFolder` represents an OnDemand folder. An object of the class `ODCriteria` represents the search criteria for an OnDemand folder. An object of the class `ODHit` represents an OnDemand document.

The following example uses `ODFolder` methods to open the specified folder, display the folder name, description, display order and search criteria, search the folder, and close the folder. This example uses `ODCriteria` methods to set the current search operand and search values. This example uses `ODHit` methods to get the display values for the document, get the document type, get a persistent identifier for the document, get the document location, and get the MIME content type for the document.

This example demonstrates these `ODFolder` methods:

- `getName`
- `getDescription`
- `getDisplayOrder`
- `getCriteria`
- `search`
- `getSearchMessage`
- `close`

This example demonstrates these `ODCriteria` methods:

- `getName`
- `setOperand`
- `setSearchValue`
- `setSearchValues`

This example demonstrates these `ODHit` methods:

- `getDisplayValue`
- `getDisplayValues`
- `getDocType`
- `getMimeType`

- getDocLocation
- getDocId

This example also uses ODServer methods to prepare for logon, open the specified folder, and log off. This example demonstrates these ODServer methods:

- initialize
- logon
- openFolder
- terminate

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Criteria name
- Operator (must be one of eq, ne, lt, le, gt, ge, in, ni, li, nl, be, nb)
- Search value 1
- (optional) Search value 2
- Configuration directory (location of arswww.ini file)

**Note:** The number of hits may be restricted by the MAXHITS parameter in the arswww.ini file.

Example of searching a folder:

```
/************************************************************************/
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcSearch
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODCriteria odCrit;
        ODHit odHit;
        Enumeration values_enum;
        Vector hits;
        String[] display_crit;
        String header, line1, line2, hit_value, useable_value;
        boolean mismatch_detected;
        int j, k, opr;

        //-----
        // If too few parameters, display syntax and get out
        //-
        if ( argv.length < 9 )
        {
            System.out.println( "usage: java TcSearch <server> <userid> <password> <folder> <criteria> <opr> <value1> <value2> <config dir>" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Display the folder name and description" );
            System.out.println( " Get the specified criteria" );
            System.out.println( " Set the operator" );
            System.out.println( " Set the operand(s)" );
        }
```

```

System.out.println( " Search the folder" );
System.out.println( " Display search message (if any)" );
System.out.println( " Display the number of hits" );
System.out.println( " Display the hitlist with each hit using 3 lines:" );
System.out.println( "   1. The hit values returned by the ODHit.getDisplayValue method" );
System.out.println( "   2. The hit values returned by the ODHit.getDisplayValues method" );
System.out.println( "   3. The doc type, mime type, doc location, and doc id values" );
System.out.println( " " );
System.out.println( "Ensure that lines 1 and 2 of the hitlist are the same and that the" );
System.out.println( "hitlist values are the same as those displayed using the Windows Client." );
System.out.println( "If arswww.ini is restricting the number of hits, there may be fewer" );
System.out.println( "hits than displayed using the Windows Client." );
System.out.println( " " );
System.out.println( "-----" );
System.out.println( " " );

//-----
// Logon to specified server
//-----
odServer = new ODServer( );
odServer.initialize( argv[8], "TcSearch.java" );
System.out.println( "Logging on to " + argv[0] + "..." );
odServer.logon( argv[0], argv[1], argv[2] );

//-----
// Open the specified folder and find the requested criteria
//-----
System.out.println( "Opening " + argv[3] + " folder..." );
odFolder = odServer.openFolder( argv[3] );
System.out.println( "Name=" + odFolder.getName( ) + " Desc=" + odFolder.getDescription( ) + " " );
System.out.println( "Getting " + argv[4] + " criteria..." );
odCrit = odFolder.getCriteria( argv[4] );

//-----
// Convert the operator parameter to the internal operator value and set
// the criteria operator
//-----
System.out.println( "Setting operator to " + argv[5] + "..." );
if ( argv[5].equals( "eq" ) )
    opr = ODConstant.OPEqual;
else if ( argv[5].equals( "ne" ) )
    opr = ODConstant.OPNotEqual;
else if ( argv[5].equals( "lt" ) )
    opr = ODConstant.OPLessThan;
else if ( argv[5].equals( "le" ) )
    opr = ODConstant.OPLessThanEqual;
else if ( argv[5].equals( "gt" ) )
    opr = ODConstant.OPGreaterThan;
else if ( argv[5].equals( "ge" ) )
    opr = ODConstant.OPGreaterThanOrEqual;
else if ( argv[5].equals( "in" ) )
    opr = ODConstant.OPIn;
else if ( argv[5].equals( "ni" ) )
    opr = ODConstant.OPNotIn;
else if ( argv[5].equals( "li" ) )
    opr = ODConstant.OPLike;
else if ( argv[5].equals( "nl" ) )
    opr = ODConstant.OPNotLike;
else if ( argv[5].equals( "be" ) )
    opr = ODConstant.OPBetween;
else if ( argv[5].equals( "nb" ) )
    opr = ODConstant.OPNotBetween;
else
    opr = -1;

System.out.println( "Setting operand(s)..." );
odCrit.setOperand( opr );

if ( opr == ODConstant.OPBetween || opr == ODConstant.OPNotBetween )
{
    odCrit.setSearchValues( argv[6], argv[7] );
    System.out.println( " " + odCrit.getName( ) + " " + getOperatorName( opr ) + " " + argv[6] + " and " + argv[7] );
}
else
{
    odCrit.setSearchValue( argv[6] );
    System.out.println( " " + odCrit.getName( ) + " " + getOperatorName( opr ) + " " + argv[6] );
}

//-----
// Search the folder
//-----
System.out.println( " Searching " + argv[3] + "..." );
hits = odFolder.search( );
System.out.println( "   Search message: " + odFolder.getSearchMessage( ) );
System.out.println( "   Number of hits: " + hits.size( ) );

//-----
// Display the hits
//-----
mismatch_detected = false;
if ( hits != null && hits.size( ) > 0 )
{
    display_crit = odFolder.getDisplayOrder( );
    header = " ";
    for( j = 0; j < display_crit.length; j++ )

```

```

        header = header + display_crit[j] + "--";
System.out.println( "-----" );
System.out.println( header + " (from ODHit.getDisplayValue method)" );
System.out.println( header + " (from ODHit.getDisplayValues method)" );
System.out.println( "----- DocType--MimeType--DocLocation--DocId" );
System.out.println( "-----" );
for ( j = 0; j < hits.size( ); j++ )
{
    odHit = (ODHit)hits.elementAt( j );
    line1 = "";
    for ( k = 0; k < display_crit.length; k++ )
    {
        hit_value = odHit.getDisplayValue( display_crit[k] );
        useable_value = ( hit_value.equals( "" ) ) ? " " : hit_value;
        line1 = line1 + useable_value + "--";
    }
    System.out.println( line1 );
    line2 = " ";
    for ( values_enum = odHit.getDisplayValues( ); values_enum.hasMoreElements( ); )
    {
        hit_value = (String)values_enum.nextElement( );
        useable_value = ( hit_value.equals( "" ) ) ? " " : hit_value;
        line2 = line2 + useable_value + "--";
    }
    System.out.println( line2 );
    System.out.println( "-----" );
    System.out.println( " " + getDocTypeString( odHit.getDocType( ) ) +
        "--" + odHit.getMimeType( ) +
        "--" + getLocationString( odHit.getDocLocation( ) ) +
        "--" + odHit.getDocId( ) );
    if ( !line1.equals( line2 ) )
        mismatch_detected = true;
}
}

//-----
// Cleanup
//-----
odFolder.close( );
odServer.logoff( );
odServer.terminate( );
System.out.println( "-----" );
System.out.println( " " );
System.out.println( "Testcase completed - analyze if required" );
System.out.println( " " );
if ( mismatch_detected )
{
    System.out.println( "*** At least one mismatch was found between" );
    System.out.println( "*** lines 1 and 2 of a hit" );
    System.out.println( " " );
}
}

catch ( ODEception e )
{
    System.out.println( "ODEception: " + e );
    System.out.println( " id = " + e.getId( ) );
    System.out.println( " msg = " + e.getMessage( ) );
    e.printStackTrace();
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace();
}
}

static String getOperatorName( int oper )
{
    String str;
    switch( oper )
    {
        case ODConstant.OPEqual:
            str = "Equals";
            break;
        case ODConstant.OPNotEqual:
            str = "Not Equal";
            break;
        case ODConstant.OPLessThan:
            str = "Less Than";
            break;
        case ODConstant.OPLessThanEqual:
            str = "Less Than or Equal";
            break;
        case ODConstant.OPGreaterThan:
            str = "Greater Than";
            break;
        case ODConstant.OPGreaterThanOrEqual:
            str = "Greater Than or Equal";
            break;
        case ODConstant.OPIn:
            str = "In";
            break;
        case ODConstant.OPNotIn:
            str = "Not In";
            break;
    }
}

```

```

        str = "Not In";
        break;
    case ODConstant.OPLike:
        str = "Like";
        break;
    case ODConstant.OPNotLike:
        str = "Not Like";
        break;
    case ODConstant.OPBetween:
        str = "Between";
        break;
    case ODConstant.OPNotBetween:
        str = "Not Between";
        break;
    default:
        str = "Operator unknown";
        break;
    }

    return str;
}

static String getDocTypeString( char type )
{
    String str;

    switch( type )
    {
        case ODConstant.FileTypeAFP:
            str = "AFP";
            break;
        case ODConstant.FileTypeBMP:
            str = "BMP";
            break;
        case ODConstant.FileTypeEMAIL:
            str = "EMAIL";
            break;
        case ODConstant.FileTypeGIF:
            str = "GIF";
            break;
        case ODConstant.FileTypeJFIF:
            str = "JFIF";
            break;
        case ODConstant.FileTypeLINE:
            str = "LINE";
            break;
        case ODConstant.FileTypeMETA:
            str = "META";
            break;
        case ODConstant.FileTypeNONE:
            str = "NONE";
            break;
        case ODConstant.FileTypePCX:
            str = "PCX";
            break;
        case ODConstant.FileTypePDF:
            str = "PDF";
            break;
        case ODConstant.FileTypePNG:
            str = "PNG";
            break;
        case ODConstant.FileTypeTIFF:
            str = "TIFF";
            break;
        case ODConstant.FileTypeUSRDEF:
            str = "USRDEF";
            break;
        default:
            str = "*** Invalid Doc Type ***";
            break;
    }

    return str;
}

static String getLocationString( int loc )
{
    String str;

    switch( loc )
    {
        case ODConstant.DocLocationCache:
            str = "Cache";
            break;
        case ODConstant.DocLocationArchive:
            str = "Archive";
            break;
        case ODConstant.DocLocationExternal:
            str = "External";
            break;
        case ODConstant.DocLocationUnknown:
            str = "Unknown";
            break;
        default:
            str = "*** Invalid Doc Location ***";
            break;
    }
}

```

```
        }
        return str;
    }
}
```

## Cancelling a search

The following example uses the `ODServer.cancel` method to cancel a search in progress.

This example uses `ODServer`, `ODFolder`, and `ODCriteria` methods to logon to a server, open a folder, and set the Date criteria to 1970-2001. A second thread is then initiated to perform a search. When second thread completes, the number of hits is displayed. A second thread is again initiated, to perform a search. The process is put to sleep for .5 seconds and then the search is cancelled. When second thread completes, the number of hits is displayed.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- openFolder
- logoff
- terminate

This example demonstrates these `ODFolder` methods:

- getCriteria
- search
- close

This example demonstrates these `ODCriteria` methods:

- setOperand
- setSearchValues

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

Example of cancelling a search:

```
/*
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

class TestThread extends Thread
{
    ODFolder odFolder;

    TestThread( ODFolder fld )
    {
        odFolder = fld;
```

```

        }

    public void run( )
    {
        Vector hits;

        try
        {
            System.out.println( " Second thread Searching..." );
            hits = odFolder.search( );
            System.out.println( " Search completed - Number of hits: " + hits.size( ) );
        }

        catch ( ODError e )
        {
            System.out.println( "ODError: " + e );
            System.out.println( " id = " + e.getErrordId( ) );
            System.out.println( " msg = " + e.getErrorMsg( ) );
            e.printStackTrace( );
        }

        catch ( Exception e2 )
        {
            System.out.println( "exception: " + e2 );
            e2.printStackTrace( );
        }
    }
}

public class TcCancelSearch
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODCriteria odCrit;
        TestThread search_thread;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcCancelSearch <server> <userid> <password> <folder> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Set the Date criteria to 1970-2001" );
            System.out.println( " Initiate a second thread to perform the search" );
            System.out.println( " When second thread completes, display the number of hits" );
            System.out.println( " Initiate a second thread to perform the search" );
            System.out.println( " Sleep for .5 seconds" );
            System.out.println( " Cancel the search" );
            System.out.println( " When second thread completes, display the number of hits" );
            System.out.println( " " );
            System.out.println( "Ensure that a folder is chosen that includes a criteria named Date." );
            System.out.println( "Ensure that the folder contains many hits and that arsww.ini is" );
            System.out.println( "not overly restricting the number of hits which can be returned." );
            System.out.println( " " );
            System.out.println( "-----" );
            System.out.println( " " );

            //-----
            // Logon to specified server
            //-----
            odServer = new ODServer( );
            odServer.initialize( argv[4], "TcCancelSearch.java" );

            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 5 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                if ( argv.length == 6 )
                    odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

            //-----
            // Open the specified folder and display its name and description
            //-----
            System.out.println( "Opening " + argv[3] + "..." );
            odFolder = odServer.openFolder( argv[3] );
            odCrit = odFolder.getCriteria( "Date" );
        }
    }
}

```

```

odCrit.setOperand( ODConstant.OPBetween );
odCrit.setSearchValues( "01/01/70", "01/01/01" );

//-----
// Start a search on a different thread, sleep briefly, awake and cancel search
//-----
System.out.println( "Main thread initiating search (will not attempt to cancel)..." );
search_thread = new TestThread( odFolder );
search_thread.start( );
search_thread.join( );

System.out.println( "Main thread initiating search (will attempt to cancel)..." );
search_thread = new TestThread( odFolder );
search_thread.start( );
System.out.println( "Main thread sleeping for .5 seconds..." );
( Thread.currentThread( ) ).sleep( 500 );
System.out.println( "Main thread attempting to cancel search..." );
odServer.cancel( );
System.out.println( "Main thread returned from attempt to cancel" );
search_thread.join( );

//-----
// Cleanup
//-----
odFolder.close( );
odServer.logout( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - Ensure that the second search," );
System.out.println( " which was cancelled, yielded fewer hits than the first" );
System.out.println( "" );
}

catch ( ODEException e )
{
    System.out.println( "ODEException: " + e );
    System.out.println( "    id = " + e.getErrorId( ) );
    System.out.println( "    msg = " + e.getErrMsg( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}

```

## **Listing search criteria**

The following example demonstrates how to use `ODCriteria` methods to list the search criteria for a given folder. For each search field, this example lists the name of the search field, the default operator, the operators that are valid for the field, the field type, and any default search values. The default values are listed by the `ODCriteria.getSearchValues` and `ODCriteria.getValues` methods. Fixed search values are listed for any search fields that are defined as `FixedChoice` or `Segment`.

This example demonstrates these `ODCriteria` methods:

- setOperand
  - getValidOperands
  - getType
  - getValues
  - setSearchValues
  - getFixedValues

This example demonstrates these `ODServer` methods:

- initialize
- logon
- openFolder
- logoff
- terminate

This example demonstrates these ODFolder methods:

- getCriteria
- close

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Configuration directory (location of arswww.ini file)
- (optional) Local server directory

Example of accessing search criteria:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcListCriteria
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODCriteria odCrit;
        Enumeration crit_enum;
        Vector value_vec;
        String[] search_values, fixed_values;
        int[] valid_ops;
        int j, opr;
        char field_type;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcListCriteria <server> <userid> <password> <folder> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Display the folder name and description" );
            System.out.println( " Display the number of folder criteria" );
            System.out.println( " For each criteria, display the" );
            System.out.println( "     Name" );
            System.out.println( "     Default operator" );
            System.out.println( "     Valid operators" );
            System.out.println( "     Field Type" );
            System.out.println( "     Default values (by ODCrit.getSearchValues method)" );
            System.out.println( "     Default values (by ODCrit.getValues method)" );
            System.out.println( "     Fixed values (only for FixedChoice and Segment criteria)" );
            System.out.println( " " );
            System.out.println( "Ensure that none of the operators indicates 'Unknown operator','" );
            System.out.println( "that none of the field types indicates 'Unknown type', that the" );
            System.out.println( "default values are the same for each method, and that all" );
            System.out.println( "information is the same as that displayed using the Windows Client." );
            System.out.println( " " );
            System.out.println( "-----" );
        }
    }
}
```

```

System.out.println( "" );

//-----
// Logon to the specified server
//-----
odServer = new ODServer( );
odServer.initialize( argv[4], "TcListCriteria.java" );

System.out.println( "Logging on to " + argv[0] + "..." );
if ( argv.length == 5 )
    odServer.logon( argv[0], argv[1], argv[2] );
else
    if ( argv.length == 6 )
        odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

//-----
// Open the specified folder and display its name and description
//-----
System.out.println( "Opening " + argv[3] + " folder..." );
odFolder = odServer.openFolder( argv[3] );
System.out.println( "Name=" + odFolder.getName( ) + " Desc=" + odFolder.getDescription( ) + "!" );
System.out.println( "There are " + odFolder.getNumCriteria( ) + " criteria:" );

//-----
// For each folder criteria,
//-----
for ( crit_enum = odFolder.getCriteria( ); crit_enum.hasMoreElements( ); )
{
    //-----
    // Display criteria name
    //-----
    System.out.println( "" );
    odCrit = (ODCriteria)crit_enum.nextElement( );
    System.out.println( odCrit.getName( ) );

    //-----
    // Display default operator
    //-----
    opr = odCrit.getOperand( );
    System.out.println( " Default operator: " );
    System.out.println( "    " + getOperatorName( opr ) );

    //-----
    // Display valid operators
    //-----
    valid_oprs = odCrit.getValidOperands( );
    System.out.println( " Valid operators:" );
    for ( j = 0; j < valid_oprs.length; j++ )
        System.out.println( "    " + getOperatorName( valid_oprs[j] ) );

    //-----
    // Display field type
    //-----
    field_type = odCrit.getType( );
    System.out.println( " Type:" );
    System.out.println( "    " + getTypeName( field_type ) );

    //-----
    // Display default value(s) using ODCrit.getValues( )
    //-----
    value_vec = odCrit.getValues( );
    System.out.println( " Default Value(s) (ODCrit.getValues method):" );
    System.out.println( "    " + value_vec.elementAt( 0 ) + "!" );
    System.out.println( "    " + value_vec.elementAt( 1 ) + "!" );

    //-----
    // Display default value(s) using ODCrit.getSearchValues( )
    //-----
    search_values = odCrit.getSearchValues( );
    System.out.println( " Default Values (ODCrit.getSearchValues method):" );
    for ( j = 0; j < search_values.length; j++ )
        System.out.println( "    " + search_values[j] + "!" );

    //-----
    // Display fixed choices
    //-----
    switch ( field_type )
    {
        case ODConstant.InputTypeChoice:
        case ODConstant.InputTypeSegment:
            fixed_values = odCrit.getFixedValues( );
            System.out.println( " Fixed Values (only for field types FixedChoice and Segment):" );
            for ( j = 0; j < fixed_values.length; j++ )
                System.out.println( "    " + fixed_values[j] + "!" );
            break;
    }
}

//-----
// Cleanup
//-----

```

```

//-----
odFolder.close( );
odServer.logoff( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - analyze and compare results to" );
System.out.println( "                                Windows Client if required" );
System.out.println( "" );
}

catch ( ODEception e )
{
    System.out.println( "ODEception: " + e );
    System.out.println( "    id = " + e.getErrorId( ) );
    System.out.println( "    msg = " + e.getErrorMsg( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}

static String getOperatorName( int oper )
{
    String str;

    switch( oper )
    {
        case ODConstant.OPEqual:
            str = "Equal";
            break;
        case ODConstant.OPNotEqual:
            str = "Not Equal";
            break;
        case ODConstant.OPLessThan:
            str = "Less Than";
            break;
        case ODConstant.OPLessThanEqual:
            str = "Less Than or Equal";
            break;
        case ODConstant.OPGreaterThan:
            str = "Greater Than";
            break;
        case ODConstant.OPGreaterThanOrEqual:
            str = "Greather Than or Equal";
            break;
        case ODConstant.OPIn:
            str = "In";
            break;
        case ODConstant.OPNotIn:
            str = "Not In";
            break;
        case ODConstant.OPLike:
            str = "Like";
            break;
        case ODConstant.OPNotLike:
            str = "Not Like";
            break;
        case ODConstant.OPBetween:
            str = "Between";
            break;
        case ODConstant.OPNotBetween:
            str = "Not Between";
            break;
        default:
            str = "*** Unknown operator";
            break;
    }

    return str;
}

static String getTypeName( char type )
{
    String str;

    switch( type )
    {
        case ODConstant.InputTypeNormal:
            str = "Normal";
            break;
        case ODConstant.InputTypeTextSearch:
            str = "TextSearch";
            break;
        case ODConstant.InputTypeNoteTextSearch:
            str = "NoteTextSearch";
            break;
    }

    return str;
}

```

```

        str = "NoteTextSearch";
        break;
    case ODConstant.InputTypeNoteColor:
        str = "NoteColor";
        break;
    case ODConstant.InputTypeChoice:
        str = "FixedChoice";
        break;
    case ODConstant.InputTypeSegment:
        str = "Segment";
        break;
    default:
        str = "*** Unknown type";
        break;
    }

    return str;
}
}

```

---

## Listing folders and folder information

The following example uses `ODServer` methods to print a line showing the number of folders on the specified server that may be searched by the specified userid. The example prints one line for each folder, showing the folder name and description.

This example demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `getNumFolders`
- `getFolderNames`
- `getFolderDescription`
- `logoff`
- `terminate`

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

Example of listing folders and folder information:

```

//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcListFolders
{
    public static void main ( String argv[] )
    {
        ODServer      odServer;
        Enumeration  folders_enum;
        String        folder_name, folder_desc;
        int          num_folders;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 4 )
        {
            System.out.println( "usage: java TcListFolders <server> <userid> <password> <config dir> [<local server dir>]" );
        }
    }
}

```

```

        return;
    }

try
{
//-----
// Set the stage
//-----
System.out.println( "This testcase should:" );
System.out.println( " Display a line showing number of folders on the server available to the userid" );
System.out.println( " Display one line for each folder, showing name and description" );
System.out.println( "" );
System.out.println( "The information should be the same as that displayed using the Windows Client" );
System.out.println( "(with the 'All' button checked if available), but the sequence of the folders" );
System.out.println( "may be different depending on the server specified" );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );

//-----
// Logon to specified server
//-----
odServer = new ODServer( );
odServer.initialize( argv[3], "TcListFolders.java" );

System.out.println( "Logging on to " + argv[0] + "..." );
if ( argv.length == 4 )
    odServer.logon( argv[0], argv[1], argv[2] );
else
    if ( argv.length == 5 )
        odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[4] );

//-----
// Display the number of folders available.
//-----
num_folders = odServer.getNumFolders( );
System.out.println( "" );
System.out.println( "There are " + num_folders + " folders available to " + argv[1] + " on " + argv[0] + ":" );

//-----
// Display the folder names and descriptions
//-----
for ( folders_enum = odServer.getFolderNames( ); folders_enum.hasMoreElements( ); )
{
    folder_name = (String)folders_enum.nextElement( );
    folder_desc = odServer.getFolderDescription( folder_name );
    System.out.println( " " + folder_name + " --- " + folder_desc );
}

//-----
// Cleanup
//-----
odServer.logoff( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - compare results to Windows Client if required" );
System.out.println( "" );
}

catch ( ODError e )
{
    System.out.println( "ODError: " + e );
    System.out.println( " id = " + e.getErrorId( ) );
    System.out.println( " msg = " + e.getErrorMsg( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}
}

```

## Displaying a list of documents

The following example uses `ODFolder` and `ODHit` methods to search a folder using the default search criteria, print the number of documents that matched the query, and lists the documents that matched the query.

This example demonstrates these `ODFolder` methods:

- `getName`
- `getDisplayOrder`
- `search`
- `close`

This example demonstrates these `ODHit` methods:

- `getDisplayValue`

This example also demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `openFolder`
- `logoff`
- `terminate`

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.ini` file)

Example of displaying a list of documents:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcSortedHitlist
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODHit odHit;
        Vector hits;
        String[] display_crit;
        String server, userid, password, folder, value;
        int j, k;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcSortedHitlist <server> <userid> <password> <folder> <config dir>" );
            return;
        }
        try
```

```

{
//-----
// Set the stage
//-----
System.out.println( "This testcase should:" );
System.out.println( " Logon to the specified server" );
System.out.println( " Open the specified folder" );
System.out.println( " Search the folder using the default criteria" );
System.out.println( " Display search message (if any)" );
System.out.println( " Display the number of hits" );
System.out.println( " Display the hitlist" );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );

//-----
// Logon to the server
//-----
server = argv[0];
userid = argv[1];
password = argv[2];
folder = argv[3];
odServer = new ODServer();
odServer.initialize( argv[4], "TcSortedHitlist.java" );
System.out.println( "Logging on to " + server + " as " + userid + "/" + password + "..." );
odServer.logon( server, userid, password );

//-----
// Open and search the folder
//-----
System.out.println( "Opening " + folder + "..." );
odFolder = odServer.openFolder( folder );
System.out.println( "Searching folder with default criteria..." );
hits = odFolder.search( );
System.out.println( " Number of hits: " + hits.size( ) );

//-----
// Display the hits
//-----
if ( hits != null && hits.size( ) > 0 )
{
    display_crit = odFolder.getDisplayOrder( );
    value = " ";
    for( j = 0; j < display_crit.length; j++ )
        value = value + display_crit[j] + " ";
    System.out.println( value );
    for ( j = 0; j < hits.size( ); j++ )
    {
        odHit = (ODHit)hits.elementAt( j );
        value = " ";
        for ( k = 0; k < display_crit.length; k++ )
            value = value + odHit.getDisplayValue( display_crit[k] ) + " ";
        System.out.println( value );
    }
}

//-----
// Cleanup
//-----
odFolder.close( );
odServer.logoff( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - Ensure that the order of the hits" );
System.out.println( " is the same as shown by the Windows Client" );
System.out.println( "" );
}

catch ( ODError e )
{
    System.out.println( "ODError: " + e );
    System.out.println( " id = " + e.getId( ) );
    System.out.println( " msg = " + e.getMessage( ) );
    e.printStackTrace( );
}

```

```
        }

        catch ( Exception e2 )
        {
            System.out.println( "exception: " + e2 );
            e2.printStackTrace( );
        }
    }
}
```

## Retrieving a document

The following example demonstrates three different methods of retrieving a document:

- ODServer
- ODFolder
- ODHit

This example logs on to the specified server, opens the specified folder, searches the folder using the default criteria, displays the number of hits, retrieves the data for the first hit using `ODHit.retrieve`, retrieves the data for the first hit using `ODServer.retrieve`, and retrieves the data for the first hit using `ODFolder.retrieve`. This example displays the length of data retrieved from each method, compares the lengths and data retrieved from each method, and displays the result of the comparisons.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- openFolder
- retrieve
- logoff
- terminate

This example demonstrates these `ODFolder` methods:

- search
- retrieve
- close

This example demonstrates these `ODHit` methods:

- getDocId
- retrieve

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

## Example of retrieving a document:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcRetrieve
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODHit odHit;
        TcCallback callback;
        Vector hits;
        Vector hit_to_retrieve;
        byte[] data_from_hit;
        byte[] data_from_server;
        byte[] data_from_folder;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcRetrieve <server> <userid> <password> <folder> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Search the folder using the default criteria" );
            System.out.println( " Display the number of hits" );
            System.out.println( " Retrieve the data for the first hit using ODHit.retrieve" );
            System.out.println( " Retrieve the data for the first hit using ODServer.retrieve" );
            System.out.println( " Retrieve the data for the first hit using ODFolder.retrieve" );
            System.out.println( " Display length of data retrieved from each method" );
            System.out.println( " Compare the lengths and data retrieved from each method" );
            System.out.println( " Display the result of the comparisons" );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );

            //-----
            // Logon to specified server
            //-----
            odServer = new ODServer( );
            odServer.initialize( argv[4], "TcRetrieve.java" );
            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 5 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

            //-----
            // Open the specified folder and search with the default criteria
            //-----
            System.out.println( "Opening " + argv[3] + " folder..." );
            odFolder = odServer.openFolder( argv[3] );
            System.out.println( "Searching with default criteria..." );
            hits = odFolder.search( );
            System.out.println( "Number of hits: " + hits.size( ) );

            //-----
            // Do some retrieves and comparisons
            //-----
            if ( hits.size( ) > 0 )
            {
                odHit = (ODHit)hits.elementAt( 0 );
                System.out.println( "Retrieving data from first hit using ODHit.retrieve..." );
                data_from_hit = odHit.retrieve( "" );
                System.out.println( "Retrieving data from first hit using ODServer.retrieve..." );
                data_from_server = odServer.retrieve( odHit.getDocId( ), argv[3], "" );
                hit_to_retrieve = new Vector( );
                hit_to_retrieve.addElement( odHit );
                System.out.println( "Retrieving data from first hit using ODFolder.retrieve (uses callback method)..." );
                callback = new TcCallback();
                odFolder.retrieve( hit_to_retrieve, callback );
                data_from_folder = callback.getData( );
            }
        }
    }
}
```

```

System.out.println( "Length of data from:" );
System.out.println( "    ODHit.retrieve=" + data_from_hit.length );
System.out.println( "    ODSERVER.retrieve=" + data_from_server.length );
System.out.println( "    ODFolder.retrieve=" + data_from_folder.length );
if ( data_from_hit.length == data_from_server.length )
{
    for ( j = 0; j < data_from_hit.length; j++ )
    {
        if ( data_from_hit[j] != data_from_server[j] )
            break;
    }
    if ( j == data_from_hit.length )
    {
        System.out.println( "ODHit vs. ODSERVER: Length and content of data match" );
        if ( data_from_hit.length == data_from_folder.length )
        {
            for ( j = 0; j < data_from_folder.length; j++ )
            {
                if ( data_from_hit[j] != data_from_folder[j] )
                    break;
            }
            if ( j == data_from_folder.length )
                System.out.println( "ODHit vs. ODFolder: Length and content of data matches" );
            else
            {
                System.out.println( "*** ODHIT vs. ODFolder: Data mismatch at offset " + j );
                System.out.println( "    ODHIT data is " + data_from_hit[j] );
                System.out.println( "    ODFolder data is " + data_from_folder[j] );
            }
        }
        else
            System.out.println( "*** ODHIT vs. ODFolder: Length mismatch" );
    }
    else
        System.out.println( "*** ODHIT vs. ODSERVER: Data mismatch at offset " + j );
        System.out.println( "    ODHIT data is " + data_from_hit[j] );
        System.out.println( "    ODSERVER data is " + data_from_server[j] );
    }
}
else
    System.out.println( "*** ODHIT vs. ODSERVER: Length mismatch" );
}
else
    System.out.println( "There is no document to retrieve" );

//-----
// Cleanup
//-----
odFolder.close();
odServer.logoff();
odServer.terminate();
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - analyze the result of the comparisons" );
System.out.println( "" );
System.out.println( "If the arswww.ini file specifies 'native' for the data type, all" );
System.out.println( "lengths and data should match; otherwise, differences are expected." );
System.out.println( "" );
}

catch ( ODEException e )
{
    System.out.println( "ODEException: " + e );
    System.out.println( "    id = " + e.getErrorHandlerId() );
    System.out.println( "    msg = " + e.getErrorMsg() );
    e.printStackTrace();
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace();
}
}
}

```

The following example uses `ODCallback` methods for bulk retrieval of document data.

```

//****************************************************************************
import java.util.*;
import java.io.*;

```

```

import com.ibm.edms.od.*;
public class TcCallback extends ODCallback
{
    byte[] data_from_folder;
    boolean init = true;

    TcCallback( )
    {
    }

    public void HitHandleCallback( int hit, int off, int len )
    {
    }

    public boolean HitCallback( String docid, char type, String[] values )
            throws Exception
    {
        return true;
    }

    public boolean DataCallback( byte[] data )
    {
        byte[] temp;
        int j, k;

        //-----
        // If first data block received, initialize container; otherwise,
        // append new data to that previously received.
        //-----
        if ( init )
        {
            data_from_folder = data;
            init = false;
        }
        else
        {
            temp = new byte[ data_from_folder.length + data.length ];
            for ( j = 0; j < data_from_folder.length; j++ )
                temp[j] = data_from_folder[j];
            k = data_from_folder.length;
            for ( j = 0; j < data.length; j++ )
                temp[k++] = data[j];
            data_from_folder = temp;
        }
        return true;
    }

    public byte[] getData( )
    {
        return data_from_folder;
    }
}

```

---

## Printing a document

The following example uses `ODServer` and `ODFolder` methods to list the printers that are available on the server and to print a document to the specified server printer. This example also uses `ODServer` methods to prepare for logon, open the specified folder, and log off.

This example demonstrates these `ODServer` methods:

- initialize
- logon

- openFolder
- getServerPrinters
- logoff
- terminate

This example demonstrates these ODFolder methods:

- search
- printDocs
- close

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Printer name
- Configuration directory (location of arswww.ini file)
- (optional) Local server directory

Example of printing a document:

```
/*
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcPrintHit
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODHit odHit;
        Vector hits, hit_to_print;
        String [] printers;
        String printer_name;
        boolean match;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 6 )
        {
            System.out.println( "usage: java TcPrintHit <server> <userid> <password> <folder> <printer> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Display the list of printers available on the server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Search the folder using the default criteria" );
            System.out.println( " Display the number of hits" );
            System.out.println( " Print the first hit to the specified server printer" );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );

            //-----
            // Logon to specified server
            //-----
            odServer = new ODServer( );
            odServer.initialize( argv[5], "TcPrintHit.java" );
            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 6 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[6] );
        }
    }
}
```

```

-----
// If any server printers are available on the server
-----
System.out.println( "Retrieving list of server printers..." );
printer_name = argv[4];
printers = odServer.getServerPrinters( );
if ( printers.length > 0 )
{
    -----
    // List the available server printers
    -----
    System.out.println( "There are " + printers.length + " printers available on the server:" );
    match = false;
    for( j = 0; j < printers.length; j++ )
    {
        System.out.println( " " + printers[j] );
        if ( printers[j].equals( printer_name ) )
            match = true;
    }

    if ( match )
    {
        -----
        // Open the specified folder and search with the default criteria
        -----
        System.out.println( "Opening " + argv[3] + " folder..." );
        odFolder = odServer.openFolder( argv[3] );
        System.out.println( "Searching with default criteria..." );
        hits = odFolder.search( );
        System.out.println( " Number of hits: " + hits.size( ) );

        -----
        // Print the first hit to the specified server printer
        -----
        if ( hits.size( ) > 0 )
        {
            hit_to_print = new Vector( );
            odHit = (ODHit)hits.elementAt( 0 );
            hit_to_print.addElement( odHit );
            System.out.println( "Printing first hit to " + printer_name + "..." );
            odFolder.printDocs( hit_to_print, printer_name );
        }
        else
            System.out.println( "There is no document to print" );

        odFolder.close( );
    }
    else
        System.out.println( "The specified printer (" + printer_name + ") is not available on this server" );
}
else
    System.out.println( "No printers are available on this server" );

-----
// Cleanup
-----
odServer.logoff( );
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );
System.out.println( "Testcase completed - Analyze the results" );
System.out.println( "" );
}

catch ( ODEException e )
{
    System.out.println( "ODEException: " + e );
    System.out.println( " id = " + e.getErrorId( ) );
    System.out.println( " msg = " + e.getErrorMsg( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}
}

```

## **Listing information about notes**

The following example uses `ODNote` methods to list detailed information about a note. This example logs on to the specified server, opens the specified folder, searches the folder using the default criteria, displays the number of hits, displays the number of notes associated with the first document, and displays detailed information for each note that is attached to the document. The information includes the position of the note on the page of the document, the background color, the date and time that the note was attached to the document, the userid that created the note and other attributes.

This example demonstrates these `ODNote` methods:

- `getColor`
- `getDateTime`
- `getGroupName`
- `getOffsetX`
- `getOffsetY`
- `getPageNum`
- `getText`
- `getUserid`
- `isOkToCopy`
- `isPublic`

This example also demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `openFolder`
- `logoff`
- `terminate`

This example also demonstrates these `ODFolder` methods:

- `search`
- `close`

This example also demonstrates these `ODHit` methods:

- `getNotes`

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

Example of listing information about notes:

```

//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcListNotes
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODHit odHit;
        ODNote odNote;
        Vector hits, notes;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcListNotes <server> <userid> <password> <folder> <config dir> [<local server dir>" );
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Search the folder using the default criteria" );
            System.out.println( " Display the number of hits" );
            System.out.println( " Display the number of notes associated with the first hit" );
            System.out.println( " Display info for each note" );
            System.out.println( " " );
            System.out.println( "-----" );
            System.out.println( " " );

            //-----
            // Logon to specified server
            //-----
            odServer = new ODServer();
            odServer.initialize( argv[4], "TcListNotes.java" );
            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 5 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[6] );

            //-----
            // Open the specified folder and search with the default criteria
            //-----
            System.out.println( "Opening " + argv[3] + " folder..." );
            odFolder = odServer.openFolder( argv[3] );
            System.out.println( "Searching with default criteria..." );
            hits = odFolder.search( );
            System.out.println( " Number of hits: " + hits.size( ) );

            //-----
            // List info for each note for the first hit
            //-----
            if ( hits.size( ) > 0 )
            {
                odHit = (ODHit)hits.elementAt( 0 );
                notes = odHit.getNotes( );
                System.out.println(" There are " + notes.size( ) + " notes for the first hit" );
                for ( j = 0; j < notes.size( ); j++ )
                {
                    odNote = (ODNote)notes.elementAt( j );
                    System.out.println(" " + (j+1) + ". Text=\"" + odNote.getText( ) + "\"" );
                    System.out.println(" UserId=" + odNote.getUserId( ) );
                    System.out.println(" Page=" + odNote.getPageNum( ) );
                    System.out.println(" Color=" + odNote.getColor( ) );
                    System.out.println(" Date=" + odNote.getDateTim( ) );
                    System.out.println(" Group=" + odNote.getGroupName( ) );
                    System.out.println(" Offset=( " + odNote.getOffsetX( ) + ", " + odNote.getOffsetY( ) + ")" );
                    System.out.println(" OkToCopy=" + odNote.isOkToCopy( ) );
                    System.out.println(" Public=" + odNote.isPublic( ) );
                }
            }
            else
                System.out.println( "There is no document - cannot list notes" );
        //-----
        // Cleanup

```

```

//-----
odFolder.close( );
odServer.logoff( );
odServer.terminate( );
System.out.println( " " );
System.out.println( "-----" );
System.out.println( " " );
System.out.println( "Testcase completed - Ensure that the information" );
System.out.println( " is the same as shown by the Windows Client" );
System.out.println( " " );
}

catch ( ODException e )
{
    System.out.println( "ODException: " + e );
    System.out.println( " id = " + e.getErrorCode( ) );
    System.out.println( " msg = " + e.getErrorMessage( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}
}

```

---

## Adding a note

An object of the class `ODHit` represents an OnDemand document. The following example uses `ODHit` methods to display the number of notes associated with the document and add a new note with the these attributes:

- The specified note text
- `OkToCopy=false`
- `Public=false` (that is, a private note)
- An empty group name

This example demonstrates these `ODHit` methods:

- `getNotes`
- `addNote`

This example also uses `ODServer` methods to prepare for logon, open the specified folder, and log off and uses the `ODFolder` methods to search the folder, get the number of hits that matched the query, and close the folder. This example demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `openFolder`
- `logoff`
- `terminate`

This example demonstrates these `ODFolder` methods:

- `search`
- `getHits`
- `close`

This example uses these run-time parameters:

- Server name
- User Id

- Password
- Folder name
- Text of note
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

### Example of adding an annotation:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcAddNote
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODHit odHit;
        ODNote odNote;
        Vector hits, notes;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 6 )
            System.out.println( "usage: java TcAddNote <server> <userid> <password> <folder> <note text> <config dir> [<local server dir>]" );
        {
            return;
        }

        try
        {
            //-----
            // Set the stage
            //-----
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Search the folder using the default criteria" );
            System.out.println( " Display the number of hits" );
            System.out.println( " Display the number of notes associated with the first hit" );
            System.out.println( " Add a new note with the these attributes" );
            System.out.println( " The specified note text" );
            System.out.println( " OKToCopy=false" );
            System.out.println( " Public=false (i.e., a private note)" );
            System.out.println( " An empty group name" );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );

            //-----
            // Logon to specified server
            //-----
            odServer = new ODServer( );
            odServer.initialize( argv[5], "TcAddNote.java" );
            System.out.println( "Logging on to " + argv[0] + "..." );
            if ( argv.length == 6 )
                odServer.logon( argv[0], argv[1], argv[2] );
            else
                odServer.logon( argv[0], argv[1], argv[2], ODConstant.CONNECT_TYPE_LOCAL, 0, argv[6] );

            //-----
            // Open the specified folder and search with the default criteria
            //-----
            System.out.println( "Opening " + argv[3] + " folder..." );
            odFolder = odServer.openFolder( argv[3] );
            System.out.println( "Searching with default criteria..." );
            odFolder.search( );
            hits = odFolder.getHits( );
            System.out.println( " Number of hits: " + hits.size( ) );

            //-----
            // Add a new note
            //-----
            if ( hits.size( ) > 0 )
            {
                odHit = (ODHit)hits.elementAt( 0 );
                notes = odHit.getNotes( );
                System.out.println(" There are " + notes.size( ) + " notes for the first hit" );

                odNote = new ODNote( );
                odNote.setText( argv[4] );
                odNote.setGroupName( "" );
                odNote.setOkToCopy( false );
                odNote.setPublic( false );
            }
        }
    }
}
```

```

        System.out.println(" Adding a new note with:" );
        System.out.println("   Text=" + odNote.getText( ) + "''");
        System.out.println("   OkToCopy=" + odNote.isOkToCopy( ) );
        System.out.println("   Public=" + odNote.isPublic( ) );
        System.out.println("   Group=" + odNote.getGroupName( ) );

        odHit.addNote( odNote );
    }
    else
        System.out.println( "No document - cannot list notes" );

    //-----
    // Cleanup
    //-----
    odFolder.close( );
    odServer.logoff( );
    odServer.terminate( );
    System.out.println( "" );
    System.out.println( "-----" );
    System.out.println( "" );
    System.out.println( "Testcase completed - Ensure that the new note was correctly" );
    System.out.println( " added by displaying it with the Windows Client" );
    System.out.println( "" );
}

catch ( ODException e )
{
    System.out.println( "ODException: " + e );
    System.out.println( "   id = " + e.getId( ) );
    System.out.println( "   msg = " + e.getMessage( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}
}

```

## Updating a document

The following example demonstrates how to update a document.

This example uses `ODServer`, `ODFolder`, and `ODCriteria` methods to connect to a server using the specified userid and password, open the specified folder, set the search values for two search fields, set the Date search field to null, and search the folder. For the document that matches the query, `ODHit` methods are then used to update the value of the second search field with the new (specified) search value.

This example demonstrates these `ODServer` methods:

- `initialize`
- `logon`
- `openFolder`
- `logoff`
- `terminate`

This example demonstrates these `ODFolder` methods:

- `getName`
- `getDisplayOrder`
- `getCriteria`
- `search`
- `closeInitialize`

This example demonstrates these OD`Criteria` methods:

- `setOperand`
- `setSearchValue`

This example demonstrates these OD`Hit` methods:

- `getDisplayValue`
- `update`

This example uses these run-time parameters:

- Server name
- User Id
- Password
- Folder name
- Criteria name 1
- Search value 1
- Criteria name 2
- Search value 2
- New search value to replace search value 2
- Configuration directory (location of `arswww.ini` file)

Example of updating a document:

```
//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcUpdate
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        ODFolder odFolder;
        ODCriteria odCrit;
        ODHit odHit;
        Hashtable hash;
        Vector hits;
        String[] display_crit;
        String line, crit1, crit2, value1, value2, new_value;
        int j;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 10 )
        {
            System.out.println( "usage: java TcUpdate <server> <userid> <password> <folder> <criterial> <value1>" +
                               "<critera2> <value2> <new value2>" );
            return;
        }

        try
        {
            System.out.println( "This testcase should:" );
            System.out.println( " Logon to the specified server" );
            System.out.println( " Open the specified folder" );
            System.out.println( " Set the search values" );
            System.out.println( " Search the folder" );
            System.out.println( " For the first hit, change the value of 2nd specified criteria" );
            System.out.println( " to the new value" );
            System.out.println( "" );
            System.out.println( "Using the Windows Client, ensure that the value has been changed." );
            System.out.println( "" );
            System.out.println( "-----" );
            System.out.println( "" );
        }
    }
}
```

```

//-----
// Logon to specified server
//-----
odServer = new ODServer( );
odServer.initialize( argv[9], "TcUpdate.java" );
System.out.println( "Logging on to " + argv[0] + "..." );
odServer.logon( argv[0], argv[1], argv[2] );

//-----
// Open the specified folder and set the requested criteria
//-----
crit1 = argv[4];
crit2 = argv[6];
value1 = argv[5];
value2 = argv[7];
new_value = argv[8];
System.out.println( "Opening " + argv[3] + " folder..." );
odFolder = odServer.openFolder( argv[3] );
odCrit = odFolder.getCriteria( crit1 );
odCrit.setOperand( ODConstant.OPEqual );
odCrit.setSearchValue( value1 );
odCrit = odFolder.getCriteria( crit2 );
odCrit.setOperand( ODConstant.OPEqual );
odCrit.setSearchValue( value2 );

//-----
// Search the folder
//-----
System.out.println( " Searching for " + crit1 + " = " + value1 + " and " + crit2 + " = " + value2 + "..." );
hits = odFolder.search( );

//-----
// If there was at least one hit
//-----
if ( hits != null & hits.size( ) > 0 )
{
    //-----
    // Display the values for the first hit
    //-----
    System.out.println( " For first hit:" );
    line = " ";
    display_crit = odFolder.getDisplayOrder( );
    for( j = 0; j < display_crit.length; j++ )
        line = line + display_crit[j] + " ";
    System.out.println( line );
    line = " ";
    odHit = (ODHit)hits.elementAt( 0 );
    for ( j = 0; j < display_crit.length; j++ )
        line = line + odHit.getDisplayValue( display_crit[j] ) + " ";
    System.out.println( line );

    //-----
    // Create a hash table of existing critera/value pairs, except for critera 2
    // which will be set to the new value. Update the hit values
    //-----
    System.out.println( " Replacing " + crit2 + " = " + value2 + " with " + crit2 + " = " + new_value );
    hash = new Hashtable( );
    for ( j = 0; j < display_crit.length; j++ )
    {
        if ( display_crit[j].equals( crit2 ) )
            hash.put( display_crit[j], new_value );
        else
            hash.put( display_crit[j], odHit.getDisplayValue( display_crit[j] ) );
    }
    odHit.update( hash );
}
else
    System.out.println( "There were no hits" );

//-----
// Cleanup
//-----
odFolder.close( );
odServer.logout( );
odServer.terminate( );
System.out.println( "-----" );
System.out.println( " " );
System.out.println( "-----" );
System.out.println( " " );
System.out.println( "Testcase completed - Using the Windows Client," );

```

```

        System.out.println( " ensure that the value has been changed." );
        System.out.println( "" );
    }

    catch ( ODError e )
    {
        System.out.println( "ODError: " + e );
        System.out.println( " id = " + e.getErrorHandlerId( ) );
        System.out.println( " msg = " + e.getErrorMessage( ) );
        e.printStackTrace( );
    }

    catch ( Exception e2 )
    {
        System.out.println( "exception: " + e2 );
        e2.printStackTrace( );
    }
}
}

```

## Changing a password

The following example uses the `ODServer` method `changePassword` to change the specified user's password to a new password. This example also uses `ODServer` methods to prepare for logon and log off.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- `changePassword`
- logoff
- terminate

This example uses these run-time parameters:

- Server name
- User Id
- Password
- New password
- Configuration directory (location of `arswww.ini` file)
- (optional) Local server directory

Example of changing a password:

```

//*****
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;

public class TcChangePassword
{
    public static void main ( String argv[] )
    {
        ODServer odServer;
        String server, userid, original_password, new_password;

        //-----
        // If too few parameters, display syntax and get out
        //-----
        if ( argv.length < 5 )
        {
            System.out.println( "usage: java TcChangePassword <server> <userid> <password> <new password> <config dir> [<local server dir>]" );
            return;
        }

        try
        {
            //-----
            // Set the stage

```

```

//-----
System.out.println( "This testcase should:" );
System.out.println( " Logon to the server using the specified password" );
System.out.println( " Change the password to the new password" );
System.out.println( " Logoff" );
System.out.println( " Logon to the server using the new password" );
System.out.println( " Change the password back to the original password" );
System.out.println( " Logoff" );
System.out.println( " If the testcase executes without exception, no further analysis" );
System.out.println( " is required." );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "" );

//-----
// Create the specified server
//-----
server = argv[0];
userid = argv[1];
original_password = argv[2];
new_password = argv[3];
odServer = new ODServer( );
odServer.initialize( argv[4], "TcChangePassword.java" );

//-----
// Logon to the server using the original password
//-----
System.out.println( "Logging on to " + server + " using original password..." );
if ( argv.length == 5 )
    odServer.logon( server, userid, original_password );
else
    if ( argv.length == 6 )
        odServer.logon( server, userid, original_password, ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

//-----
// Change to the new password and logoff
//-----
System.out.println( "Changing to new password..." );
odServer.changePassword( new_password );
System.out.println( "Logging off..." );
odServer.logoff( );

//-----
// Logon to the server using the new password
//-----
System.out.println( "Logging on to " + server + " using new password..." );
if ( argv.length == 5 )
    odServer.logon( server, userid, new_password );
else
    if ( argv.length == 6 )
        odServer.logon( server, userid, new_password, ODConstant.CONNECT_TYPE_LOCAL, 0, argv[5] );

//-----
// Change back to the original password and logoff
//-----
System.out.println( "Changing back to original password..." );
odServer.changePassword( original_password );
System.out.println( "Logging off..." );
odServer.logoff( );

//-----
// Cleanup
//-----
odServer.terminate( );
System.out.println( "" );
System.out.println( "-----" );
System.out.println( "Testcase completed successfully" );
System.out.println( "" );
}

catch ( ODError e )
{
    System.out.println( "ODError: " + e );
    System.out.println( " id = " + e.getErrorId( ) );
    System.out.println( " msg = " + e.getErrMsg( ) );
    e.printStackTrace( );
}

catch ( Exception e2 )
{
    System.out.println( "exception: " + e2 );
    e2.printStackTrace( );
}
}
}

```



## Appendix E. Java line data viewer

Beginning with Version 7.1.0.11, IBM now provides an enhanced Java line data viewer. Functional improvements include enhanced printing functions, such as printing the entire width of the page. The graphical user interface is based on the Swing library.

IBM now provides two versions of the Java line data viewer in the applets directory:

**ODLineDataViewer.jar** is the old Java line data viewer, which requires Version 1.1.8 or later of the Java plugin.

**ODLineDataViewer2.jar** is the new Java line data viewer, which requires Version 1.4.1 or later of the Java plugin.

Customers can use the new Java line data viewer or the old Java line data viewer. The choice is specified by setting parameters in the [DEFAULT BROWSER] section of the ARSWWW.INI file. In addition, the new Java line data viewer requires Version 1.4.1 or later of the Java plugin for the browser. Additional parameters in the ARSWWW.INI file determine the version number and the location of the Java plugin installation file for users that do not have the required version of the Java plugin installed on their workstations.

Table 21 describes new parameters in the ARSWWW.INI file that support the Java line data viewer.

*Table 21. Parameters in ARSWWW.INI File for Java Line Data Viewer*

Parameter	Value	Comments
ODApplet.version	1	Specifies to invoke the old Java line data viewer. If specified, ignore the remaining parameters. <b>Note:</b> This is the default value. Also, if this parameter is omitted, ODWEK will use the old Java line data viewer.
	2	Specifies to invoke the new Java line data viewer (enhanced version). If specified, use the following three parameters.

*Table 21. Parameters in ARSWWW.INI File for Java Line Data Viewer (continued)*

Parameter	Value	Comments
ODApplet.jre.path.IE	<p>http://java.sun.com/getjava/installer.html</p>	<p>For Internet Explorer. Specifies to automatically download and install the latest version of the Java plugin from the java.sun.com Web site. See <a href="http://java.sun.com/getjava/install-windows.html">http://java.sun.com/getjava/install-windows.html</a> for a preview of what happens when users automatically download and install the Java plugin. <b>Note:</b> The user may need to restart the browser after installing the plugin.</p>
	<p>&lt;location&gt;</p>	<p>Specifies the location of the Java plugin installation file within a company's intranet. The location must include a valid browser protocol, such as http, file, or ftp. For example:  <code>file:///shareName/java/plugins/plugin.exe</code></p> <p><b>Note:</b> An administrator must download the Java plugin installation file and store it in the specified location.</p> <p>By specifying the location of the installation file, the browser will automatically install the Java plugin on the workstation. The user may need to restart the browser after the installation completes.</p>

*Table 21. Parameters in ARSWWW.INI File for Java Line Data Viewer (continued)*

Parameter	Value	Comments
ODApplet.jre.path.NN	<p><code>http://java.sun.com/j2se/1.4.1/download.html</code></p>	<p>For Netscape. Specifies to open the JRE/J2SE Download page for choosing the Java plugin to install. The user would follow the link to download the Java plugin installation file for their platform. After downloading the Java plugin installation file, the user must install the plugin on the workstation. The user may need to restart the browser after installing the plugin.</p>
	<p><code>&lt;location&gt;</code></p>	<p>Specifies the location of the plugin file(s) within a company's intranet. The location must include a valid browser protocol, such as http, file, or ftp. For example:</p> <p><code>http://webServer/tmp/ondemand/java/plugins</code></p> <p><b>Note:</b> An administrator must download the plugin file(s) and store them in the specified location. You cannot specify a path to a specific file because it is not known on which operating system that Netscape is running. Also, the specified format allows the administrator to download the plugin for other platforms, if desired.</p> <p>The user must install the Java plugin on the workstation. The user may need to restart the browser after installing the plugin.</p>

*Table 21. Parameters in ARSWWW.INI File for Java Line Data Viewer (continued)*

Parameter	Value	Comments
ODApplet.jre.version	<version>	Specifies the version of the Java plugin to use. Must specify version 1.4 or later. Specify a major version number (for example, 1.4) to support any release of the plugin at that level (for example, 1.4.0, 1.4.0_03, 1.4.1_01). Specify a specific version number (for example 1.4.1_01) to support only that version of the Java plugin. Obtain valid version numbers from the java.sun.com Web site. For example: 1.4 or: 1.4.1_01

The following example shows how to configure the ARSWWW.INI file to support the old Java line data viewer.

```
[DEFAULT BROWSER]
ODApplet.version=1
```

**Notes:**

1. If you omit the ODApplet.version parameter from the ARSWWW.INI file, ODWEK will use the old Java line data viewer.
2. The ODApplet parameters have a global scope and may only be specified in the DEFAULT BROWSER section. (If these parameters are specified in some other browser section, they will be ignored.)

The following shows an example of how to configure the ARSWWW.INI file to support the new Java line data viewer (enhanced version) and Version 1.4 or later of the Java plugin. For Internet Explorer, users can automatically download and install the latest version of the Java plugin from the java.sun.com Web site. For Netscape, an administrator has stored copies of the Java plugin installation files for the various platforms in the specified location on a local Web server so that users do not have to go to the java.sun.com JRE/J2SE Download page. **Note:** Only users that do not have Version 1.4 or later of the Java plugin installed on their workstations will be prompted to download / install the plugin.

```
| [DEFAULT BROWSER]  
| ODApplet.version=2  
| ODApplet.jre.path.IE=http://java.sun.com/getjava/installer.html  
| ODApplet.jre.path.NN=http://localWebServer/java/plugins  
| ODApplet.jre.version=1.4
```



## Appendix F. Xenos transforms

You can use ODWEK and the Xenos transforms to retrieve AFP and Metacode documents from an OnDemand server, transform the documents, and send the output files to the Web browser. For example, you could use the Xenos transform with the ARSLOAD program to process and load Metacode print files. Then, you could use ODWEK to retrieve a Metacode document from the system, call the Xenos transform to convert the Metacode document and send the converted output to the browser.

When retrieving documents from the system by using ODWEK, you can use the Xenos transforms to:

- Convert AFP documents to HTML, PDF or XML files.
- Convert Metacode documents to AFP, HTML, PDF, or XML files.

The files that the Xenos transforms creates can then be sent to a Web browser for viewing and printing.

**Important:** Before you attempt to use the Xenos transforms on your system, you must obtain the transform programs, license, and documentation. See your IBM representative for more information. Also see your IBM representative for information on education and other types of help and support for installing and configuring the transform programs and processing input files with the transform programs.

This section describes how to configure ODWEK to run the Xenos transforms. This section contains the following topics:

- “Converting data streams” on page 168
- “Viewing and printing with the Web Enablement Kit” on page 170
- “Configuring the ARSWWW.INI file” on page 173
- “Configuring the ARSXENOS.INI file” on page 176
- “Example of a parameter file” on page 178
- “Example of a script file” on page 181
- “Changes to the Retrieve Document API” on page 183

## Converting data streams

OnDemand customers can use the Xenos transforms with ODWEK for the following types of data conversions:

- Convert AFP documents to HTML
- Convert AFP data to PDF
- Convert AFP documents to XML
- Convert Metacode data to AFP
- Convert Metacode documents to HTML
- Convert Metacode data to PDF
- Convert Metacode documents to XML

The following sections describe each type of data. Please see the Xenos documentation to better understand the data formats and for more information about the transforms.

### AFP documents to HTML

There are two AFP to HTML transforms:

- The AFP to HTML/CSS transform converts AFP documents into HTML files with linked Cascading Style Sheets (CSS) for display with a Web browser. The output uses fonts from the user's system to display the pages, so it may not match the original as closely as other formats. Graphics are displayed using linked PNG image files. A separate HTML file is generated for each page in the original document. An optional HTML frameset file can also be generated to permit easier navigation between pages. The frameset is an HTML page that displays the individual HTML pages in a large frame at the top, and has links to the other pages in a small frame at the bottom.
- The AFP to HTML/Template Merger allows data from the input document to be inserted in a predefined template file. This allows the pages to be viewed to have a different format and layout than the printed page for redesigning the page to look and fit better on a standard computer screen. Multiple templates can be used in a single application, as needed, to suit the variety of information that is found from page to page in the input document.

### AFP data to PDF

When retrieving AFP documents from the system by using ODWEK, the transform processes fully-composed MO:DCA-P files. The PDF output can be viewed at the Web browser by using an Adobe PDF viewer. To run the AFP to PDF transform when retrieving AFP documents from the system, set the AFPVIEWING parameter to XENOS in the ARSWWW.INI file and specify PDF as the OUTPUTTYPE in the ARSXENOS.INI file.

## **AFP documents to XML**

The AFP data to XML transform uses the AFP to HTML/Template Merger to allow data from the input document to be inserted in a predefined template file. This allows the pages to be viewed to have a different format and layout than the printed page, such as separating summary information from transaction details. The template is written in XML, for applications such as an online payment system. Multiple templates can be used in a single application, as needed, to suit the variety of information that is found from page to page in the input document.

## **Metacode to AFP**

When retrieving Metacode documents from the system by using ODWEK, the transform processes pure Metacode documents. The AFP output can be viewed at the Web browser by using the AFP Web Viewer. To run the Metacode to AFP transform when retrieving Metacode documents from the system by using ODWEK, set the METAVIEWING parameter to XENOS in the ARSWWW.INI file and specify AFP as the OUTPUTTYPE in the ARSXENOS.INI file.

## **Metacode documents to HTML**

There are two Metacode to HTML transforms:

- The Metacode to HTML/CSS transform converts Metacode documents into HTML files with linked Cascading Style Sheets (CSS) for display with a Web browser. The output uses fonts from the user's system to display the pages, so it may not match the original as closely as other formats. Graphics are displayed using linked PNG image files. A separate HTML file is generated for each page in the original document. An optional HTML frameset file can also be generated to permit easier navigation between pages. The frameset is an HTML page that displays the individual HTML pages in a large frame at the top, and has links to the other pages in a small frame at the bottom.
- The Metacode to HTML/Template Merger transform allows data from the input document to be inserted in a predefined template file. This allows the pages to be viewed to have a different format and layout than the printed page for redesigning the page to look and fit better on a standard computer screen. Multiple templates can be used in a single application, as needed, to suit the variety of information that is found from page to page in the input document.

## **Metacode to PDF**

When retrieving Metacode documents from the system by using ODWEK, the transform processes pure Metacode documents. The PDF output can be viewed at the Web browser by using an Adobe PDF viewer. To run the Metacode to PDF transform when retrieving Metacode documents from the

system by using ODWEK, set the METAVIEWING parameter to XENOS in the ARSWWW.INI file and specify PDF as the OUTPUTTYPE in the ARSXENOS.INI file.

## Metacode documents to XML

The Metacode to XML transform uses the Metacode to HTML/Template Merger to allow data from the input document to be inserted in a predefined template file. This allows the pages to be viewed to have a different format and layout than the printed page, such as separating summary information from transaction details. The template is written in XML, for applications such as an online payment system. Multiple templates can be used in a single application, as needed, to suit the variety of information that is found from page to page in the input document.

---

## Viewing and printing with the Web Enablement Kit

Figure 8 on page 172 shows the steps that you take to view and print documents locally with ODWEK.

1. The process begins with the Index Data (1a), Documents (1b), and Resources (1c) that were loaded into the database and on to storage volumes with the ARSLOAD program (AFP, Metacode/DJDE, or PDF documents).
2. In the ARSWWW.INI file, you specify information about the documents that ODWEK retrieves from the OnDemand server. For AFP and Metacode/DJDE documents, you can specify that a different type of output should be sent to the Web browser. For example, you can specify that Metacode/DJDE documents that are retrieved from the system should be transformed into PDF files that are sent to the Web browser. See “Configuring the ARSWWW.INI file” on page 173 for information about how to configure the ARSWWW.INI file to support the Xenos transform.
3. In the ARSXENOS.INI file, you specify the output type and the name of the parameter file and the script file that are used by the Xenos transform to process the input document and create the output that is sent to the Web browser. See “Configuring the ARSXENOS.INI file” on page 176 for an example of the ARSXENOS.INI file.
4. When ODWEK retrieves a document from the system, it checks the ARSWWW.INI file to determine if document conversion is required.
5. If document conversion is required, then ODWEK sends the document to the Xenos transform. Otherwise, ODWEK sends the document to the Web browser.
6. You create the Parameter File, which provides information about the type of conversion that the Xenos transform performs. See “Example of a parameter file” on page 178 for an example of a parameter file.

- | 7. You create the Script file, which the Xenos transform uses to create the
- | output file. See “Example of a script file” on page 181 for an example of a
- | script file.
- | 8. The Xenos transform converts the document from the format in which it
- | was stored on the server to a format that can be viewed at the Web
- | browser.
- | 9. The user views the document from the Web browser. The user can also
- | print documents on local printers from the Web browser.

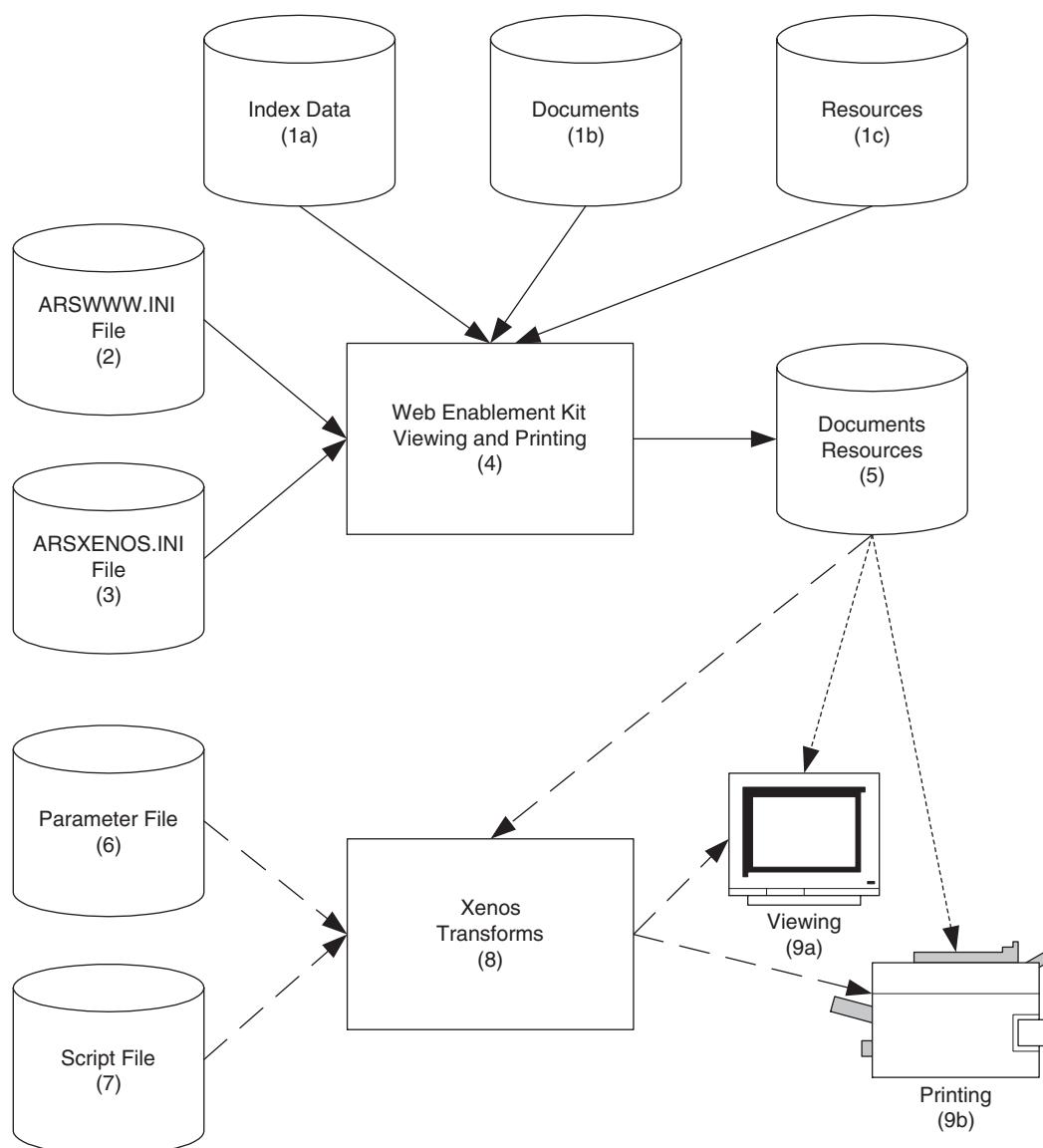


Figure 8. Viewing and Printing with the Web Enablement Kit

## Configuring the ARSWWW.INI file

You must make the following changes to the ARSWWW.INI file to enable the Xenos transform:

- Add the XENOS section
- Specify an AFP transform option
- Specify a Metacode/DJDE transform option

### [XENOS]

The XENOS section contains the parameters that are used by the Xenos transform. You can run the Xenos transform from ODWEK to convert AFP documents into HTML, PDF, or XML output or Metacode documents into AFP, HTML, PDF, or XML output that can be viewed from a Web browser.

#### Notes:

1. To convert documents, an administrator must obtain the Xenos transforms and install and configure them on the server. See your IBM representative for more information about the Xenos transforms. An administrator must also provide configuration options for the Xenos transforms. See “Configuring the ARSXENOS.INI file” on page 176 for more information about the configuration file.
2. To convert AFP documents with the Xenos transforms, you must specify the AFPVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections). See “AFPVIEWING” on page 59 for details. (If you plan to use the Retrieve Document API, then you should specify the \_afp=XENOS parameter. See “Changes to the Retrieve Document API” on page 183 for details.)
3. To convert Metacode documents with the Xenos transforms, you must specify the METAVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections). See “METAVIEWING” on page 175 for details. (If you plan to use the Retrieve Document API, then you should specify the \_meta=XENOS parameter. See “Changes to the Retrieve Document API” on page 183 for details.)
4. If a document is stored in OnDemand as a large object, then the value of the ALLOBJECTS parameter determines how ODWEK handles the document. See “Configuring the ARSXENOS.INI file” on page 176 for details.

This section has a global scope, and you specify it only once in the ARSWWW.INI file.

This section is optional.

This section can contain the following parameters:

## **CONFIGFILE**

Specifies the configuration file that contains the options used by the Xenos transform to convert AFP and Metacode/DJDE documents. “Configuring the ARSXENOS.INI file” on page 176 provides information about the configuration file that is provided with OnDemand.

This parameter has a global scope, and you specify it only once in the XENOS section.

This parameter is optional.

Example:

```
[XENOS]
CONFIGFILE=/usr/lpp/ars/www/arsxenos.ini
```

## **INSTALLDIR**

Specifies the directory that contains the Xenos transform programs and configuration files. Specify the full path name of the directory on the Web server.

**Note:** Verify the permissions of the directory that you specify. The processes that run ODWEK programs must read the install directory.

This parameter has a global scope, and you specify it only once in the XENOS section.

This parameter is optional.

Example:

```
[XENOS]
INSTALLDIR=/usr/lpp/ars/www
```

## **Specifying the AFP transform option**

If you plan to process AFP documents with the Xenos transform, then you need to set the value of the AFPVIEWING parameter in the DEFAULT BROWSER (or other browser sections) of the ARSWWW.INI file.

## **AFPVIEWING**

When ODWEK retrieves an AFP document from the OnDemand server, the value of this parameter determines what action, if any, that ODWEK takes before sending the document to the Web browser. To convert AFP documents to HTML, PDF, or XML output with the Xenos transform, specify AFPVIEWING=XENOS so that ODWEK will call the Xenos transform to convert the AFP document before sending it to the Web browser. The type of output that is generated is determined by the value of the OUTPUTTYPE parameter in the ARSXENOS.INI file (see “Configuring the ARSXENOS.INI file” on page 176).

This parameter has a global scope, and you specify it only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the \_afp parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
AFPVIEWING=XENOS
```

## Specifying the Metacode transform option

If you plan to process Metacode/DJDE documents with the Xenos transform, then you need to set the value of the METAVIEWING parameter in the DEFAULT BROWSER (or other browser sections) of the ARSWWW.INI file.

### METAVIEWING

When the WEK retrieves a Metacode/DJDE document from the OnDemand server, the value of this parameter determines what action, if any, that ODWEK takes before sending the document to the Web browser. For example, to convert Metacode/DJDE documents to AFP, HTML, PDF, or XML output with the Xenos transform, specify METAVIEWING=XENOS so that ODWEK will call the Xenos transform to convert the Metacode/DJDE document before sending it to the Web browser. The type of output that is generated is determined by the value of the OUTPUTTYPE parameter in the ARSXENOS.INI file (see “Configuring the ARSXENOS.INI file” on page 176).

You can set the parameter to one of the following values:

**NATIVE**      ODWEK extracts and will uncompress Metacode/DJDE documents and their resources from OnDemand.

**XENOS**      ODWEK calls the Xenos transform to convert Metacode/DJDE documents to AFP, HTML, PDF, or XML output.

This parameter has a global scope, and you specify it only once in the DEFAULT BROWSER section. When using the Retrieve Document function, you can override the specified action with the \_meta parameter.

This parameter is optional.

Example:

```
[DEFAULT BROWSER]  
METAVIEWING=XENOS
```

## Configuring the ARSXENOS.INI file

The Xenos transforms can convert AFP documents into HTML, PDF, or XML output or Metacode documents into AFP, HTML, PDF, or XML output that can be viewed from a Web browser. The Xenos transforms are licensed software. An administrator must install and configure the Xenos transforms on the Web server. See your IBM representative for more information about the Xenos transforms. An administrator must also specify configuration options for the documents that you plan to process with the Xenos transforms. This section describes how to specify the configuration options.

**Note:** In this document, the name ARSXENOS.INI refers to the configuration file. To specify the file that contains the configuration options, see “CONFIGFILE” on page 174.

The ARSXENOS.INI file provides configuration options for the Xenos transform. You typically configure the ARSXENOS.INI file with options for specific OnDemand applications. However, you can also provide a set of default options. The Xenos transform uses the default options when it converts documents from applications that are not identified in the ARSXENOS.INI file.

The following topics provide additional information about the ARSXENOS.INI file:

- Specifying the ARSXENOS.INI file
- Viewing converted documents

**Note:** To convert AFP documents with the Xenos transform, you must also specify the AFPVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See “AFPVIEWING” on page 59 for details. (If you plan to use the Retrieve Document API, then you should specify the \_afp=XENOS parameter. See “Changes to the Retrieve Document API” on page 183 for details.) To convert Metacode documents with the Xenos transform, you must also specify the METAVIEWING=XENOS parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See “METAVIEWING” on page 175 for details. (If you plan to use the Retrieve Document API, then you should specify the \_meta=XENOS parameter. See “Changes to the Retrieve Document API” on page 183 for details.)

### Specifying the ARSXENOS.INI file

The following is an example of an ARSXENOS.INI file:

```
[CREDIT-CREDIT]
ParmFile=/usr/lpp/ars/www/afp2pdf/sample.par
ScriptFile=/usr/lpp/ars/www/noindex.dms
```

```
| LicenseFile=/usr/lpp/ars/www/dmlic.txt  
| OutputType=pdf  
  
[default]  
ParmFile=/usr/lpp/ars/www/afp2pdf/sample.par  
ScriptFile=/usr/lpp/ars/www/noindex.dms  
LicenseFile=/usr/lpp/ars/www/dmlic.txt  
OutputType=pdf  
AllObjects=0  
WarningLevel=4
```

The structure of the file is similar to a Windows INI file, and contains one stanza for each OnDemand application and one default stanza. The title line of the stanza identifies the application group and application. For example, the title line:

```
[CREDIT-CREDIT]
```

Identifies the CREDIT application group and the CREDIT application. Use the – (dash) character to separate the names in the title line. The names must match the application group and application names in OnDemand. If the application group contains more than one application, then create one stanza for each application.

The parameters that you specify in the [default] stanza are used by the Xenos transform to process documents from applications that are not identified in the ARSXENOS.INI file. The default parameters are also used if an application stanza does not include one of the parameters.

The ParmFile parameter identifies the full path name of the file that contains the parameters that are used by the Xenos transform to convert the document. See the Xenos documentation for details about the parameters that you can specify in the parameter file. “Example of a parameter file” on page 178 shows an example of a parameter file.

The ScriptFile parameter identifies the full path name of the file that contains the script statements that are used by the Xenos transform to create the output file. See the Xenos documentation for details about the script file. “Example of a script file” on page 181 shows an example of a script file.

The LicenseFile parameter identifies the full path name of a valid license file that you obtained from Xenos.

The OutputType parameter determines the type of conversion that the Xenos transform performs. If the input document is AFP, you can set this parameter to HTML, PDF, or XML. If the input document is Metacode, you can set this parameter to AFP, HTML, PDF, or XML.

The AllObjects parameter determines how ODWEK will process documents that are stored as large objects in OnDemand. If you specify 0 (zero), then ODWEK will retrieve only the first segment of a document. If you specify 1 (one), then ODWEK will retrieve all of the segments and convert them before sending the document to the viewer. **Note:** If you enable large object support for very large documents, then your users may experience a significant delay before they can view the document.

The WarningLevel parameter determines how ODWEK will handle return codes from the Xenos transform. The Xenos transform sets a return code after each document is converted. Use this parameter to specify the maximum return code that ODWEK will consider good and send the converted document to the viewer. For example, if you specify 4 (four), then the return code that is set by the Xenos transform must be four or less; otherwise, ODWEK will not send the converted document to the viewer. The default value is zero. If you do not specify this parameter, then the Xenos transform must set a return code of zero. Otherwise, ODWEK will not send the converted document to the viewer. See the Xenos documentation for details about return codes.

## **Viewing converted documents**

To view AFP documents requires the AFP Web Viewer that is provided with ODWEK.

To view HTML or XML documents requires Internet Explorer Version 4.01 or later or Netscape Version 4.06 or later.

To view PDF documents requires an Adobe PDF viewer.

---

## **Example of a parameter file**

Figure 9 on page 180 shows an example of a parameter file that is used by the Xenos transform to convert AFP documents that are stored in OnDemand to PDF documents that ODWEK sends to the Web browser.

The script variables Parser and Generator determine the type of conversion taking place. The script variable Parser represents the input file format and the script variable Generator represents the output file format. Table 22 lists the possible values for the Parser and Generator variables when retrieving documents from the system and transforming them before sending the documents to the Web browser.

*Table 22. Parser and Generator Variables*

Input	Output	Parser	Generator
AFP	HTML	scriptvar='('Parser', 'AFP')	scriptvar='('Generator', 'HTML')

*Table 22. Parser and Generator Variables (continued)*

Input	Output	Parser	Generator
AFP	PDF	scriptvar=('Parser', 'AFP')	scriptvar=('Generator', 'PDF')
Metacode	AFP	scriptvar=('Parser', 'META')	scriptvar=('Generator', 'AFP')
Metacode	HTML	scriptvar=('Parser', 'META')	scriptvar=('Generator', 'HTML')
Metacode	PDF	scriptvar=('Parser', 'META')	scriptvar=('Generator', 'PDF')

```

/* Sample processing parameters for the Xenos transform */

JS:

/* DM Script Library - XG supplied functions */
fddmslib = '/usr/lpp/ars/www/dmsl.lib'

scriptvar = ('Parser', 'AFP')
scriptvar = ('Generator', 'PDF')

AFPDL-AFPP:
/* AFP Parser Options */
formdef = f1a10111
pagedef = p1a06462
CC      = on
trc     = off
startpage = 0
stoppage = 0
native   = no
position  = word

/* File Defs */
FDpagesegs = '/usr/lpp/ars/www/resources/afp/%s.psg'
FDafpffonts = '/usr/lpp/ars/www/resources/afp/%s.fnt'
FDpagedefs = '/usr/lpp/ars/www/resources/afp/%s.pde'
FDformdefs = '/usr/lpp/ars/www/resources/afp/%s.fde'
FDoverlays = '/usr/lpp/ars/www/resources/afp/%s.ovr'

FDfontcor = '/usr/lpp/ars/www/newfont.tab'

PDFGEN-PDFOUT:

/* PDF Out Generator Options */
offset      = (0,0)
scaleby    = 100
border     = NONE
compress  = (NONE,NONE,NONE)
orient     = AUTO
pdfauthor = 'Xenos Group'
pdfopenact = '/FitH 800'
bmorder   = (AsIs,AsIs,AsIs)

```

*Figure 9. Sample Parameters for the Xenos Transform*

## Example of a script file

Figure 10 on page 182 shows an example of a script file that is used by the Xenos transform to generate the output file that ODWEK sends to the Web browser.

### Notes:

1. The script file does not create an index file because an index file is not needed for this step.
2. If the input file is Metacode, it must be read using the Xenos {lf2xm} I/O directive. For example:

```
rc = dm_SetParm(par_h, 'fdinput','{lf2xm}'inputfile)
```

```

| TRUE = 1
| FALSE = 0
|
| CALL dm_Initialize
|
| par_h = dm_StartParser(Parser)
| gen_h = dm_StartGenerator(Generator)
|
| rc = dm_SetParm(par_h, 'fdinput', inputfile);
| rc = dm_SetParm(gen_h, 'fdoutput', outputfile);
|
| /* initialize */
| file_open = FALSE
|
| dlpage = dm_GetDLPage(par_h)
|
| DO WHILE(dlpage <> 'EOF')
|   if file_open = FALSE then do
|     if( generator = 'AFP' ) then do
|       rc = dm_AFPGenOpen(gen_h)
|     end
|     else if( generator = 'PDF' ) then do
|       rc = dm_PDFGenOpen(gen_h)
|     end
|     file_open = TRUE
|   end
|
|   if( generator = 'AFP' ) then do
|     rc = dm_AFPGenWrite(gen_h, dlpage)
|   end
|   else if( generator = 'PDF' ) then do
|     rc = dm_PDFGenWrite(gen_h, dlpage)
|   end
|
|   dlpage = dm_GetDLPage(par_h)
| END
|
| if file_open = TRUE then do
|   if( generator = 'AFP' ) then do
|     rc = dm_AFPGenClose(gen_h)
|   end
|   else if( generator = 'PDF' ) then do
|     rc = dm_PDFGenClose(gen_h)
|   end
| END
|
| RETURN

```

*Figure 10. Sample JS Program Script File*

## Changes to the Retrieve Document API

This section describes the changes that were made to the Retrieve Document API to support the Xenos transform.

1. The `_afp` parameter now accepts a value of **XENOS**. When you specify `_afp=XENOS`, ODWEK calls the Xenos transform to convert the AFP document that was retrieved from the system into HTML, PDF, or XML output that will be sent to the Web browser. The type of output that is generated is determined by the value of the `OUTPUTTYPE` parameter in the ARSXENOS.INI file (see “Configuring the ARSXENOS.INI file” on page 176).
2. The `_meta` parameter was added. You can use the `_meta` parameter to specify the transform that should take place when ODWEK retrieves a Metacode/DJDE document from the system. You can specify one of the following values:

**NATIVE**      ODWEK extracts and will uncompress the Metacode/DJDE document and its resources from OnDemand.

**XENOS**      ODWEK calls the Xenos transform to convert the Metacode/DJDE document into AFP, HTML, PDF, or XML output. The type of output that is generated is determined by the value of the `OUTPUTTYPE` parameter in the ARSXENOS.INI file (see “Configuring the ARSXENOS.INI file” on page 176).



---

## Appendix G. AFP to HTML transform

**Important:** The AFP2WEB and AFP2PDF transforms must be installed to separate directories so that the fonts are not intermixed for both transforms. The two transforms must be mutually exclusive or they will not produce correct output.

The AFP to HTML transform process converts AFP documents and resources into HTML documents. The AFP to HTML transform process requires the AFP2WEB Transform service offering from IBM Printing Systems Division. An administrator must install and configure the AFP2WEB Transform on the Web server. See your IBM representative for more information about the AFP2WEB Transform service offering. Someone in your organization must also specify configuration options for the AFP documents and resources that you plan to process with the AFP2WEB Transform. This section describes how to specify the configuration options.

**Note:** In this document, the name AFP2HTML.INI refers to the configuration file. To specify the file that contains the configuration options, see "CONFIGFILE" on page 47.

The AFP2HTML.INI file provides configuration options for the AFP2WEB Transform. You typically configure the AFP2HTML.INI file with options for specific AFP applications. However, you can also provide a set of default options. The AFP2WEB Transform uses the default options when converting documents and resources for AFP applications that are not identified in the AFP2HTML.INI file. To learn more details about the options and the conversion process, see the AFP2WEB Transform documentation.

The following topics provide additional information about the AFP2HTML.INI file:

- Format of the AFP2HTML.INI file
- Options for the AFP2WEB Transform
- Viewing converted documents

**Note:** To convert documents with the AFP2HTML applet, you must also specify the AFPVIEWING=HTML parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See "AFPVIEWING" on page 59 for details. (If you plan to use the Retrieve Document API, then you should specify the \_afp=HTML parameter. See

“Retrieve Document” on page 103 for details.) You must also specify the directory that contains the AFP2WEB Transform programs (see “CONFIGFILE” on page 47).

---

## Format of the AFP2HTML.INI file

The following is an example of an AFP2HTML.INI file:

```
[CREDIT-CREDIT]
UseApplet=FALSE
ScaleFactor=1.0
CreateGIF=TRUE
SuppressFonts=FALSE
FontMapFile=creditFontMap.cfg
ImageMapFile=creditImageMap.cfg

[default]
ScaleFactor=1.0
CreateGIF=TRUE
SuppressFonts=FALSE
FontMapFile=fontmap.cfg
ImageMapFile=imagemap.cfg
```

The structure of the file is similar to a Windows INI file, and contains one stanza for each AFP application and one default stanza. The title line of the stanza identifies the application group and application. For example, the title line:

```
[CREDIT-CREDIT]
```

Identifies the CREDIT application group and the CREDIT application. Use the – (dash) character to separate the names in the title line. The names must match the application group and application names defined to the OnDemand server. If the application group contains more than one application, then create one stanza for each application.

The options in the [default] stanza are used by the AFP2WEB Transform to process documents for AFP applications that are not identified in the AFP2HTML.INI file. The defaults are also used if an AFP application stanza does not include one of the options.

The UseApplet option is a directive to ODWEK. It determines whether the AFP2HTML applet will be used to view the output from the AFP2WEB Transform. The default value is TRUE. If you specify FALSE (the AFP2HTML applet is not used to view the output), the output is formatted and displayed by the Web browser.

The remaining five options are directives to the AFP2WEB Transform. “Options for the AFP2WEB Transform” on page 187 briefly describes how they are used by the AFP2WEB Transform.

## Options for the AFP2WEB Transform

Table 23 lists the options that you can specify in the AFP2HTML.INI file to convert documents with the AFP2WEB Transform.

*Table 23. Options for the AFP2WEB Transform*

Option in AFP2HTML.INI file	Description
AllObjects	Determines how ODWEK processes documents that are stored as large objects in OnDemand. The default value is 0 (zero), and means that ODWEK will retrieve only the first segment of a document. If you specify 1 (one), then ODWEK will retrieve all of the segments and convert them before sending the document to the client. <b>Note:</b> If you enable large object support for very large documents, then your users may experience a significant delay before they can view the document.
ScaleFactor	Scales the output with the given scale factor. The default value is 1.0. For example, specifying a value of ScaleFactor=2.0 scales the output to be twice as large as the default size; specifying a value of ScaleFactor=0.5 scales the output to one half of the default size. The default size is derived from the Zoom setting on the Logical Views page in the OnDemand application.
SuppressFonts	Determines whether the AFP text strings are transformed. If you specify SuppressFonts=TRUE, any text that uses a font listed in the Font Map file is not transformed. The default value is FALSE, which means that all of the AFP text strings are transformed. The Font Map file is identified with the FontMapFile option.
FontMapFile	Identifies the full path name of the Font Map file. The Font Map file contains a list of fonts that require special processing. The default Font Map file is named <code>imagfont.cfg</code> and resides in the directory that contains the AFP2WEB Transform programs. See the AFP2WEB Transform documentation for details about the Font Map file.

*Table 23. Options for the AFP2WEB Transform (continued)*

Option in AFP2HTML.INI file	Description
ImageMapFile	Identifies the image mapping file. The image mapping file can be used to remove images from the output, improve the look of shaded images, and substitute existing images for images created by the AFP2WEB Transform. Mapping images that are common across your AFP documents (for example, a company logo) reduces the time required to transform documents. If specified, the image mapping file must exist in the directory that contains the AFP2WEB Transform programs. See the AFP2WEB Transform documentation for details about the image mapping file.

**Note:** ODWEK sends the following options to the AFP2WEB Transform when converting documents. These options are not specified in the AFP2HTML.INI file.

- Orientation. Determines the rotation value to use when viewing the document. The default value is derived from the Orientation setting on the View Information page in the OnDemand application.
- Image Color. Determines the color to use when viewing images and graphics. The default value is derived from the Image Color setting on the Logical Views page in the OnDemand application.

### **Viewing converted documents**

The UseApplet option in the AFP2HTML.INI file is a directive to ODWEK that determines whether the AFP2HTML applet will be used to view the converted output. The default value is TRUE. If you specify FALSE (the AFP2HTML applet is not used to view the output), the output is formatted and displayed by the Web browser.

In general, IBM recommends that you always use the AFP2HTML applet to view converted documents. If a document was stored in OnDemand as a large object, then the AFP2HTML applet adds controls to help users easily move to any page in the document.

---

## Appendix H. AFP to PDF transform

**Important:** The AFP2PDF and AFP2WEB transforms must be installed to separate directories so that the fonts are not intermixed for both transforms. The two transforms must be mutually exclusive or they will not produce correct output.

The AFP2PDF Transform converts AFP documents and resources into PDF documents. The AFP2PDF Transform is a services offering from IBM Printing Systems Division. An administrator must install and configure the AFP2PDF Transform on the Web server. See your IBM representative for more information about the AFP2PDF Transform services offering. Someone in your organization must also specify configuration options for the AFP documents and resources that you plan to process with the AFP2PDF Transform. This section describes how to specify the configuration options.

**Note:** In this document, the name AFP2PDF.INI refers to the configuration file. To specify the file that contains the configuration options, see “CONFIGFILE” on page 49.

The AFP2PDF.INI file provides configuration options for the AFP2PDF Transform. You typically configure the AFP2PDF.INI file with options for specific AFP applications. However, you can also provide a set of default options. The AFP2PDF Transform uses the default options when converting documents and resources for AFP applications that are not identified in the AFP2PDF.INI file. To learn more details about the options and the conversion process, see the AFP2PDF Transform documentation.

The following topics provide additional information about the AFP2PDF.INI file:

- Specifying the AFP2PDF.INI file
- Viewing converted documents

**Note:** To convert documents, you must also specify the AFPVIEWING=PDF parameter in the DEFAULT BROWSER section (or other browser sections) of the ARSWWW.INI file. See “AFPVIEWING” on page 59 for details. (If you plan to use the Retrieve Document API, then you should specify the \_afp=PDF parameter. See “Retrieve Document” on page 103 for details.)

---

### Specifying the AFP2PDF.INI file

The following is an example of an AFP2PDF.INI file:

```
[CREDIT-CREDIT]
OptionsFile=
ImageMapFile=creditImageMap.cfg

[default]
OptionsFile=
ImageMapFile=imagemap.cfg
AllObjects=0
```

The structure of the file is similar to a Windows INI file, and contains one stanza for each AFP application and one default stanza. The title line of the stanza identifies the application group and application. For example, the title line:

```
[CREDIT-CREDIT]
```

Identifies the CREDIT application group and the CREDIT application. Use the – (dash) character to separate the names in the title line. The names must match the application group and application names defined to the OnDemand server. If the application group contains more than one application, then create one stanza for each application.

The parameters that you specify in the [default] stanza are used by the AFP2PDF Transform to process documents for AFP applications that are not identified in the AFP2PDF.INI file. The default parameters are also used if an AFP application stanza does not include one of the parameters specified.

The OptionsFile parameter identifies the full path name of the file that contains the transform options used by the AFP2PDF Transform. The transform options are used for AFP documents that require special processing. See the AFP2PDF Transform documentation for details about the transform options file.

The ImageMapFile parameter identifies the image mapping file. The image mapping file can be used to remove images from the output, improve the look of shaded images, and substitute existing images for images created by the AFP2PDF Transform. Mapping images that are common in most of your AFP documents (such as a company logo) reduces the time required to transform documents. If specified, the image mapping file must exist in the directory that contains the programs for the AFP2PDF Transform. To specify the directory that contains the programs for the AFP2PDF Transform, see “INSTALLDIR” on page 49. See the AFP2PDF Transform documentation for details about the image mapping file.

The AllObjects parameter determines how ODWEK processes documents that are stored as large objects in OnDemand. The default value is 0 (zero), and means that ODWEK will retrieve only the first segment of a document. If you specify 1 (one), then ODWEK will retrieve all of the segments and convert

| them before sending the document to the viewer. **Note:** If you enable large  
| object support for very large documents, then your users may experience a  
| significant delay before they can view the document.

---

## Viewing converted documents

To view converted documents with the Adobe Acrobat viewer, you must obtain the plug-in for the browsers used by your organization.



---

## Appendix I. Distributing user-defined files

You can distribute user-defined files with the IBM OnDemand AFP Web Viewer software that is supplied by IBM. For example, suppose that someone in your organization creates AFP font files for documents that are stored in OnDemand. You can distribute the font files with the AFP Web Viewer software. That way, when a user views an AFP document, the document will be displayed with the correct fonts.

To distribute user-defined files with the AFP Web Viewer, you must package the files into an installation file and store the installation file in a shared location. When a user runs the installation file, the Setup program automatically installs the AFP Web Viewer and the user-defined files on the user's workstation.

You can distribute the following types of user-defined files with the AFP Web Viewer:

- AFP font files. These files are copied to the FONT subdirectory of the AFP Web Viewer destination directory on the workstation.
- Adobe Type 1 font files. These files are copied to a directory specified by the user and installed in ATM by the Setup program.

**Note:** If ATM is not installed on the user's workstation, the Setup program copies the font files to a directory specified by the user. However, the Setup program does not install the fonts. If the user installs ATM on the workstation after running the installation file, then the user can install the fonts from the ATM applet in Control Panel.

- TrueType font files. These files are copied to the Windows FONTS directory and installed in Windows by the Setup program.
- Miscellaneous user-defined files. These files are copied to the AFP Web Viewer destination directory on the user's workstation.

**Note:** The Setup program copies user-defined files to the workstation after the AFP Web Viewer files that are supplied by IBM. If you name a user-defined file the same as one of the files supplied by IBM, then the user-defined file will replace the file supplied by IBM. You can take advantage of this feature, for example, to distribute an updated FLDPORT2.INI file or to distribute IBM AFP font files that your organization has modified.

The following topics contain more information about configuring and distributing the AFP Web Viewer:

- Install the AFP Web Viewer files supplied by IBM
- Add subdirectories to hold user-defined files
- Store user-defined files in subdirectories
- Configure font files
- Build the AFP Web Viewer installation file
- Install the AFP Web Viewer on a user's workstation

---

## Installing the AFP Web Viewer files

Most customers use one of two ways to distribute the viewer files from a server, depending on whether they plan to distribute user-defined files with the AFP Web Viewer:

- Standard Install. Use to distribute the AFP Web Viewer files supplied by IBM and to prepare for distributing user-defined files with the AFP Web Viewer. When an administrator installs the ODWEK software on the Web server, the installation files for the viewers are stored in a directory on the server. There should be an installation file (EXE) for each viewer and a ZIP archive file for the AFP Web Viewer. The administrator typically moves the installation files to the public directory on the server and creates a web page with the links to the files. A user installs a viewer by loading the web page into their browser and activating the link to the appropriate installation file.
- Custom Install for the AFP Web Viewer. Use to distribute user-defined files with the AFP Web Viewer.
  1. Set up the server for a Standard Install.
  2. Before any users actually install the viewer, obtain a copy of the AFP Web Viewer ZIP archive file.
  3. Extract the files from the ZIP archive file to an empty work directory.
  4. Add subdirectories to the work directory and store user-defined files in the directories. See “Adding subdirectories” on page 195 and “Storing user-defined files” on page 196 for details.
  5. If distributing user-defined Adobe Type 1 font files, then create a font configuration file. See “Configuring font files” on page 196 for details.
  6. After all of the directories and files have been configured, create a self-extracting EXE file for distribution. See “Building the AFP Web Viewer installation file” on page 197 for details.
  7. Replace the EXE file provided by IBM for a Standard Install with the self-extracting EXE file that you created.
  8. After an administrator completes steps 1 through 7, users can install the AFP Web Viewer and the user-defined files by loading the web page into their browsers and activating the link to the updated installation file.

---

## Adding subdirectories

The user-defined files that you plan to distribute must be stored in the CUSTOM subdirectory tree under the main viewer installation directory. For example, you could name the main viewer installation directory \\ONDEMAND\\AFP32.

To configure the main viewer installation directory to hold user-defined files:

1. Create a CUSTOM directory under the main viewer installation directory. For example:

\\ondemand\\afp32\\custom

**Note:** The CUSTOM directory can hold other<sup>1</sup> user-defined files that you want to distribute to your users. The Setup program copies files from this directory to the AFP Web Viewer destination directory on the workstation.

2. Add one or more of the following subdirectories to the CUSTOM directory. The subdirectories you add depend on the type of user-defined files that you want to distribute to your users.

- Create a FONT subdirectory under the CUSTOM directory to hold AFP font files (file types FNT and MAP). For example:

\\ondemand\\afp32\\custom\\font

The Setup program copies these files to the AFP Web Viewer FONT directory on the workstation.

- Create a TYPEONE subdirectory under the CUSTOM directory to hold Adobe Type 1 font files (file types PFB and PFM) and the font configuration file. For example:

\\ondemand\\afp32\\custom\\typeone

The Setup program copies these files to a directory specified by the user and installs the fonts in ATM.

- Create a TRUEETYPE subdirectory under the CUSTOM directory to hold Windows TrueType font files (file type TTF). For example:

\\ondemand\\afp32\\custom\\truetype

The Setup program copies files from this directory to the Windows FONT directory and installs the fonts in Windows.

---

1. Other than AFP font files, Adobe Type 1 font files, and Windows TrueType font files.

---

## Storing user-defined files

After extracting the IBM-supplied installation files to the work directory and creating the CUSTOM directories, you can store the user-defined files in the individual subdirectories. For example, copy Adobe Type 1 font files (file types PFB and PFM) that you want to distribute to your users to the \\ONDEMAND\\AFP32\\CUSTOM\\TYPEONE directory.

---

## Configuring font files

If you plan to distribute user-defined Adobe Type 1 font files to your users, then you must complete the following steps:

1. Store the user-defined Type 1 font files (file types PFB and PFM) in the TYPEONE subdirectory of the CUSTOM directory. See “Adding subdirectories” on page 195 for more information.
2. Create a Type 1 font configuration file. The following information describes how to create the Type 1 font configuration file.

The Type 1 font configuration file must be named ATM\_INI.CFG and must be stored in the TYPEONE subdirectory of the CUSTOM directory. See “Adding subdirectories” on page 195 for more information about the distribution directories.

Each record (line) in the Type 1 font configuration file identifies one and only one user-defined Adobe Type 1 font that you want to distribute to your users. The format of a record is:

```
fontname=filename.PFM, filename.PFB
```

Where fontname is the name of the Type 1 font as it appears in the ATM Control Panel fonts list, filename.PFM is the name of the PFM file for the font, and filename.PFB is the name of the PFB file for the font. The following example shows a Type 1 font configuration file with two records:

```
Courier,BOLD=coub.pfm,coub.pfb  
SonoranSansSerif_36,BOLDITALIC=c0a175z0.pfm,c0a175z0.pfb
```

The first record in the file identifies the font named Courier,BOLD and its PFM font file coub.pfm and PFB font file coub.pfb. The second record in the file identifies the font named SonoranSansSerif\_36,BOLDITALIC and its PFM font file c0a175z0.pfm and PFB font file c0a175z0.pfb.

When a user runs a AFP Web Viewer installation file that contains user-defined Adobe Type 1 font files, the Setup program processes font files in the following way:

1. Copies all of the user-defined Adobe Type 1 font files (file types PFB and PFM) found in the TYPEONE directory to the destination directory. The user specifies the destination directory.
2. Verifies that two font files were copied for each font identified in the Type 1 font configuration file (ATM\_INI.CFG). The name of the files copied to the workstation must match the names specified in the font configuration file.

**Note:** If the names of the font files specified in the font configuration file do not match the names of the files copied to the workstation, the Setup program displays a warning message and does not install the font.

3. Adds path information for the PFB and PFM files, using the destination directory specified by the user.
4. Installs the fonts in ATM.

**Note:** If ATM is not installed on the user's workstation, then the Setup program copies the font files to a directory specified by the user. However, the Setup program does not install the fonts. If the user installs ATM on the workstation after running the installation file, then the user can install the fonts with the ATM applet in Control Panel.

---

## Building the AFP Web Viewer installation file

After you have finished creating directories and storing files in the CUSTOM directory tree, you must create an installation file that contains your user-defined files and the AFP Web Viewer files supplied by IBM. The installation file is usually named Setup.exe.

Several companies make software for packaging files and applications into a single, self-extracting AFP Web Viewer executable file for distribution. For example, the InstallShield Software Corporation offers a product called PackageForTheWeb.

**Note:** Software provided by other companies is not supported by IBM.

After you have obtained the packaging software, run it and follow the instructions provided to create a AFP Web Viewer installation file that contains your user-defined files and the AFP Web Viewer files supplied by IBM.

---

## **Installing the AFP Web Viewer on a user's workstation**

After you set up the CUSTOM directory tree, build the AFP Web Viewer installation file, and replace the AFP Web Viewer installation file on the server, users can begin installing the AFP Web Viewer and the user-defined files. The next time a user activates the link to the AFP Web Viewer installation file from the server, the Setup program installs the AFP Web Viewer on the user's workstation and copies all of the user-defined files that you packaged with the AFP Web Viewer installation file to the user's workstation.

---

## Appendix J. Mapping AFP fonts

AFP fonts that a document was created with need to be mapped to fonts that can be displayed using the AFP Web Viewer. ODWEK provides font definition files that map the IBM Core Interchange (Latin only) and compatibility fonts to TrueType fonts. The font definition files and font map files are stored in the FONT subdirectory in which the AFP Web Viewer files reside.

If your documents use fonts that are not defined to the AFP Web Viewer, if you or others in your organization have modified the IBM Core fonts, or if you or others in your organization have created AFP fonts, then you must define the fonts in the font definition files so that the AFP Web Viewer can correctly display the documents. Refer to the *AFP Workbench Technical Reference* for details about how to map AFP fonts, font definition files, and other technical information related to AFP and TrueType fonts.



---

## Appendix K. No HTML output

ODWEK uses the `_nohtml` directive to determine the type of output generated by a function (such as Logon). By default, ODWEK generates HTML output. If you specify `_nohtml=1`, then ODWEK generates delimited ASCII output. This chapter describes the delimited ASCII output generated by ODWEK.

---

### Delimited ASCII output

The delimited ASCII output generated by ODWEK is a set of output records that contain character string values, keywords, and function, record, and string delimiters and separators:

- Character string values are output data of a function, other than keywords, delimiters, and separators. For example, the next function to be called, the name of the folder, the folder field names, search operators, and field values are character string values.
- Keywords consist of a specific character string. For example, ACTION, DOC, FOLDER, NUMROWS, and ROW are keywords.
- The function delimiters consist of the specific character strings [BEGIN] and [END].
- The record delimiter is the new line character, `\n`. All records are delimited by the new line character.
- By default, string delimiters and separators are the caret character (`^`) and the left bracket (`[`) and right bracket (`]`) characters. For example:  
`[folderName^folderDesc]`

If a keyword record contains more than one character string value, then the values are separated by the caret character. Each keyword's set of character string values is delimited by the left bracket and right bracket characters.

Some character string values may be stored in a list, separated by the caret character and enclosed in left bracket and right bracket characters. For example, the list of valid search operators for a field may appear as follows:

`[1^2^4^8^16^32]`

You can override the default characters for the string delimiters and separators. See “[NO HTML]” on page 56 for details.

- A single null character string value is indicated by the absence of a value inside two double quote characters (“”). A null list is indicated by the absence of a value inside left bracket and right bracket characters ([ ]).

---

## Logon

The following shows an example of the delimited ASCII output generated by the Logon function:

```
[BEGIN]\nACTION=searchCriteriaUrl\nFOLDER=[folderName\folderDesc]\nFOLDER=[folderName\folderDesc]\n\n:\n[END]\n
```

### Notes

1. The string `searchCriteriaUrl` identifies the name of the next function to be executed and its parameters.
2. The string `folderName` identifies a folder name. The name is not quoted.
3. The string `folderDesc` is the description of the folder. The description is not quoted.

---

## Search Criteria

The following shows an example of delimited ASCII data generated by the Search Criteria function:

```
[BEGIN]\nACTION=hitListUrl\nDISPLAY_ORDER=[field1\field2\...fieldN]\nNUMROWS=numberOfRows\nROW=[criteriaName\[[validOp]\ndefOp]\[inpType\inpAssocData]]\n\n:\n[END]\n
```

### Notes

1. The string `hitListUrl` identifies the name of the next function to be executed and its parameters.
2. The `DISPLAY_ORDER` keyword specifies the order in which the folder fields should be displayed.
3. The string `numberOfRows` identifies the number of `ROW` keyword records that follow. The function generates one `ROW` keyword record for each search field.
4. The string `criteriaName` represents the search criteria for a search field. The search criteria is not quoted.

5. The string `validOp` is the list of integer values that represent the valid search operators for the search field:
  - 1** Equal
  - 2** Not Equal
  - 4** Less Than
  - 8** Less Than or Equal
  - 16** Greater Than
  - 32** Greater Than or Equal
  - 64** In
  - 128** Not In
  - 256** Like
  - 512** Not Like
  - 1024** Between
  - 2048** Not Between
6. The string `defOp` is an integer value representing the default search operator.
7. The string `inpType` represents the type of search field:
  - A** Annotation Text Search
  - C** Choice
  - N** Normal
  - S** Segment
  - T** Text Search
  - Z** Annotation Color Search
8. The string `inpAssocData` is a list associated with the `defOp` and `inpType`:

<b>defOp</b>	<b>inpType</b>	<b>inpAssocData</b>
Between, Not Between	N	Null: [ ] or a list: [defaultField1 ∧ ... ∧ defaultFieldN] For example: ["01/31/96" ∧ "01/31/97"] ["01/31/96" ∧ ""] ["" ∧ "01/31/97"]
Other valid operators	A, N, T, Z	Null: [ ] or a single string value that represents the default field value
Other valid operators	C, S	[ [listOfChoices] ∧ defaultChoice] For example: [[ "JFIF" ∧ "TIFF" ∧ "PCX" ] ∧ "TIFF"] [[ "JFIF" ∧ "TIFF" ∧ "PCX" ] ∧ ""]

## Document Hit List

The following shows an example of delimited ASCII output generated by the Document Hit List function:

```
[BEGIN]\nACTION=hitListURL\nMSG=Only 20 documents can be listed for this folder.\nDOC=[criteria1\criteria2\criteriaN\docid\fileType\docLocation]\n\n:\n[END]\n
```

## Notes

1. The string `hitListURL` identifies the name of the next function to be executed and the parameters for the function.
2. The `MSG` keyword shows an example of an error message in the delimited ASCII output. By default, ODWEK sends error messages to the client. However, when a function contains the `_nohtml=1` directive, ODWEK generates the message text in the delimited ASCII output instead.
3. The strings `criteria1`, `criteria2`, and `criteriaN` represent search criteria values. The values are listed in the order in which they appear in the document list. The values are not quoted.
4. The string `docid` is the document identifier for the document.
5. The string `fileType` identifies the data type of the document:
  - A** AFP
  - B** BMP
  - E** Email
  - F** JFIF
  - G** GIF
  - L** Line
  - N** None
  - O** OD Defined
  - P** PDF
  - T** TIFF
  - U** User Defined
  - X** PCX
6. The string `docLocation` identifies the storage location of the document:
  - 0** Unknown
  - 1** OnDemand cache storage
  - 2** Archive storage
  - 3** External cache storage

---

## View Annotations

The following shows an example of delimited ASCII output generated by the View Annotations function:

```
[BEGIN]\nNOTE 4: 15:42:44 PM Mountain Standard Time Thursday November 19, 1998...\nPublic - Cannot be copied to another server\nTest note from the OnDemand Internet Client.\n[END]\n
```

---

## Error Message

The following shows an example of delimited ASCII output generated when errors occur:

```
[ERROR]\nID=nnnn\nMSG=errorMessageText\n
```

### Notes

1. The string nnnn is the error message number.
2. The string errorMessageText is the error message text.



---

## Appendix L. Problem determination tools

You can use the tools listed in Table 24 to gather information about the system and documents. You can use the information to help solve problems you are having configuring ODWEK and help other people in your organization who are having problems using the applets and viewers.

*Table 24. Problem Determination Tools*

Tool	Purpose	How to Enable
HTML Output	Save a copy of the HTML that ODWEK is returning to the browser.	Choose Save As from the browser's File menu
Server Log Files	Save access information, errors, and server information.	<p>Do the following:</p> <ol style="list-style-type: none"><li>1. In the DEBUG section of the ARSWWW.INI file, set the LOG parameter to 1 (one). The log file that is generated by ODWEK is named ARSWWW.LOG and is written to the directory specified by the LOGDIR parameter. (The default directory is /tmp.)</li><li>2. Configure logging for your HTTP server. (Each HTTP server may have a different way to configure logging and may have different logs and options you can enable to collect more or less detailed information.)</li></ol> <p><b>Note:</b> Because a significant amount of information can be written to a log file, IBM recommends that you enable logging only when needed, such as when recreating a problem. If you need to enable logging for extended periods of time, make sure that the log file paths point to storage devices with plenty of free space. Remember to periodically delete old log files from the server.</p>

*Table 24. Problem Determination Tools (continued)*

Tool	Purpose	How to Enable
Java Console	Display messages generated by the applets.	<ul style="list-style-type: none"> <li>• Netscape: From the Communicator menu, select Tools and then Java Console.</li> <li>• Internet Explorer:           <ol style="list-style-type: none"> <li>1. From the View menu, select Internet Options.</li> <li>2. On the Advanced page, select Java Console.</li> <li>3. Restart the browser.</li> <li>4. From the View menu, select Java Console.</li> </ol> </li> </ul>
AFP Web Viewer Trace Facility	Capture detailed information about AFP documents being viewed with the AFP Web Viewer.	<p>Make sure the following section exists in the FLDPORT2.INI file on the user's workstation:</p> <pre>[Misc] ViewTraceFile=d:\temp\afppugin.log Trace=TRUE</pre> <p>Verify the path of the log file. Remember to turn off logging when you have gathered the information you need.</p>
OnDemand System Log	Save system messages (such as log on and log off) and application group messages having to do with documents (such as query and retrieve) and annotations.	<p>Do the following:</p> <ol style="list-style-type: none"> <li>1. Enable system and application group logging for the OnDemand server. Update the system parameters for the server using the administrative client.</li> <li>2. Enable the specific application group messages that you want to log. Update the message logging options for the application group using the administrative client.</li> </ol>

---

## Appendix M. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only the IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe on any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**  
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
Dept. MYGA – 001L  
6300 Diagonal Highway  
Boulder, CO 80501-9191  
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM

products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

---

## Trademarks and service marks

Advanced Function Presentation, AFP, AIX, CICS, IBM, iSeries, OS/390, WebSphere, and z/OS are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Adobe, the Adobe logo, Acrobat and the Acrobat logo are trademarks of Adobe Systems Incorporated, which may be registered in certain jurisdictions.

Lotus is a trademark of Lotus Development Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and Windows NT are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.



---

# Index

## Special characters

@SRV@\_DEFAULT section 35  
@SRV@\_server section 36

## A

about the OnDemand Internet Connection 1  
about this publication 1

add annotation  
    API 86  
        function description 7  
        parameters 86  
        sample function call 88

ADDEXTENSION parameter 58  
ADDFIELDSTODOCID parameter 58  
ADDNOTES parameter 59

AFP documents  
    converting 59, 174  
    media type 51  
    MIME content type 51  
    viewing 59, 174

AFP fonts

    mapping 199  
AFP Web Viewer  
    about 1  
    AFP fonts 199  
    customizing the installation 193  
    fonts 199  
    installing 75  
    installing user-defined files 193  
    mapping AFP fonts 199  
    requirements 76  
    user-defined files 193

AFP2HTML configuration file 185

AFP2HTML Java applet  
    about 1  
    APPLETCACHEDIR parameter 38  
    installing 75

    large object support 47, 188  
    requirements 13, 17

AFP2HTML section 46

AFP2PDF configuration file 189

AFP2PDF Java applet  
    directory 49  
    enabling 49

AFP2PDF section 48

AFP2PDF Transform

    about ix  
    configuring 189  
    enabling 48

AFP2PDF Transform (*continued*)

    installing 13, 17

AFP2WEB Transform  
    about ix  
    configuring 185  
    enabling 46  
    installing 13, 17

AFPVIEWING parameter 59, 174

AIX  
    installation 19  
annotations  
    ADDNOTES parameter 59, 66  
    API 86, 112  
    delimited ASCII output 204  
    function description 7, 9  
    Java API 151, 153  
    parameters 86, 112  
    sample function call 88, 113

API  
    add annotation 86  
    annotations 86, 112  
    CGI API reference 85  
    change password 89  
    classes 119  
    diagnostic information 122  
    document hit list 91  
    environment variables 121  
    examples 119  
    exception handling 123  
    Java API programming guide 119  
    Java API reference 117  
    logoff 95  
    logon 97  
    packaging 119  
    print document 99  
    programming guide 119  
    reference 85, 115, 117  
    retrieve document 103  
    sample code 119  
    search criteria 107  
    server print 99  
    system environment 121  
    tracing and diagnostic information 122  
    update document 110  
    view annotations 112  
    Xenos transforms 183

APPLETCACHEDIR parameter 38

APPLETDIR parameter 38

applets 38

applets (*continued*)  
 about 1  
 APPLETCACHEDIR parameter 38  
 directory 49  
 enabling 49  
 installing 75  
 large object support 47, 188  
 requirements 76  
 application groups in a folder  
   Java API 128  
 application name  
   Java API 125  
 application programming interface (API)  
   *See API*  
 ARSWWW.INI file  
   @SRV@\_DEFAULT section 35  
   @SRV@\_server section 36  
   ADDEXTENSION parameter 58  
   ADDFIELDSTODOCID parameter 58  
   ADDNOTES parameter 59  
   AFP2HTML section 46  
   AFP2PDF section 48  
   AFP2PDF Transform 48  
   AFP2WEB Transform 46  
   AFPVIEWING parameter 59, 174  
   APPLETCACHEDIR parameter 38  
   ATTACHMENT IMAGES section 55  
   AUTODOCRETRIEVAL parameter 60  
   BEGIN parameter 56  
   browser options 66  
   browser section 66  
   CACHEDIR parameter 39  
   CACHEDOCS parameter 39  
   CACHEMAXTHRESHOLD parameter 40  
   CACHEMINTHRESHOLD parameter 40  
   CACHESIZE parameter 40  
   CACHEUSERIDS parameter 41  
   CODEPAGE parameter 42  
   CONFIGFILE parameter 47, 49, 174  
   CONFIGURATION section 37  
   configuring 35  
   debug section 67  
   DEFAULT BROWSER section 57, 174, 175  
   DOCSIZE parameter 42  
   EMAILVIEWING parameter 61  
   ENCRYPTCOOKIE parameter 61  
   ENCRYPTURL parameter 62  
   END parameter 57  
   FOLDERDESC parameter 62  
   HOST parameter 36  
   IMAGEDIR parameter 42  
   INSTALLDIR parameter 47, 49, 174  
   LANGUAGE parameter 43  
   LINEVIEWING parameter 62

ARSWWW.INI file (*continued*)  
   LOG parameter 68, 207  
   LOGDIR parameter 68  
   MAXHITS parameter 63  
   METAVIEWING parameter 175  
   MIMETYPES section 50  
   NOHTML section 56  
   NOLINKS parameter 64  
   ODApplet.jre.path.IE parameter 64  
   ODApplet.jre.path.NN parameter 64  
   ODApplet.jre.version parameter 64  
   ODApplet.version parameter 64  
   PORT parameter 35, 37  
   PROTOCOL parameter 36, 37  
   REPORTSERVERTIMEOUT parameter 45  
   SECURITY section 45  
   SEPARATOR parameter 57  
   SERVERACCESS parameter 46  
   SERVERPRINT parameter 64  
   SERVERPRINTERS parameter 65  
   SHOWDOCLOCATION parameter 65  
   specifying 35  
   TEMPDIR parameter 44  
   TEMPLATEDIR parameter 45  
   USEEXECUTABLE parameter 48, 49  
   VIEWNOTES parameter 66  
   XENOS section 173  
   Xenos transforms 173

ARSXENOS.INI file 176

ASCII output  
   annotations 204  
   document hit list 203  
   error message 205  
   format 201  
   generated by OnDemand 201  
   logon 202  
   messages 205  
   search criteria 202  
   view annotations 204

ATTACHMENT IMAGES section 55

attachments 55, 56

AUTODOCRETRIEVAL parameter 60

**B**

BEGIN parameter 56  
 BMP attachments 55  
 BMP documents  
   media type 52  
   MIME content type 52  
 browser options  
   browser section 66  
   DEFAULT BROWSER section 57, 174, 175  
 browser section 66  
 browsers  
   cookies 76

browsers (*continued*)

- Java Virtual Machine 76
- JVM 76
- supported 76

## C

cache directory 39

cache documents 39

cache size 40

cache storage 39, 40, 41

CACHEDIR parameter 39

CACHEDOCS parameter 39

CACHEMAXTHRESHOLD parameter 40

CACHEMINTHRESHOLD parameter 40

CACHESIZE parameter 40

CACHEUSERIDS parameter 41

cancelling a search 135

CGI

- deploying 25

CGI API

- reference 85

change password

- API 89

- function description 7

- parameters 89

- sample function call 90

changing passwords 158

classes 119

code page 42

CODEPAGE parameter 42

communications protocols 36, 37

CONFIGFILE parameter 47, 49, 174

configuration

- AFP2HTML configuration file 185

- AFP2PDF configuration file 189

- ARSWWW.INI file 35

CONFIGURATION section 37

connecting to a server 124, 125

connection type

- Java API 125

cookies 61, 76

CREDIT.HTM 72

## D

data security 9

debug section 67

default browser options 57, 174, 175

DEFAULT BROWSER section 57, 174, 175

delimited ASCII output

- annotations 204

- delimiters 56

- document hit list 203

- error message 205

- format 201

- generated by OnDemand 201

delimited ASCII output (*continued*)

logon 202

messages 205

search criteria 202

view annotations 204

delimiters 56

diagnostic information 122

directory permissions 17

disconnecting from a server 125

display document location 65

display values, Java API 130

DJDE (metacode) documents

converting 175

viewing 175

DOCSIZE parameter 42

document hit list

API 91

delimited ASCII output 203

function description 8

Java API 130, 143, 145

parameters 91

sample function call 94

document location 65

document type, Java API 130

documents

AFP 59, 174

cache storage 39

converting 59, 61, 62, 174, 175

DJDE (metacode) 175

EMAIL 61

line data 62

links 64

media type 50

metacode/DJDE 175

MIME content type 50

printing with Java API 148

retrieving 60

updating with Java API 155

viewing 59, 61, 62, 174, 175

documents, Java API 143, 145

## E

EMAIL documents

converting 61

media type 52

MIME content type 52

viewing 61

EMAILVIEWING parameter 61

ENCRYPTCOOKIES parameter 61

encryption 61, 62

ENCRYPTURL parameter 62

END parameter 57

environment variables, Java API 121

error message

delimited ASCII output 205

errors 122, 207  
examples 119  
exception handling 123

## F

folder description, Java API 141  
folder name, Java API 141  
folder, listing application groups in with Java API 128  
folder, searching with Java API 130, 135, 137, 143  
FOLDERDESC parameter 62  
fonts

  AFP 199  
  mapping 199  
  TrueType 199

functions  
  add annotation 7  
  annotations 7, 9  
  change password 7  
  document hit list 8  
  logoff 8  
  logon 8  
  print document 8  
  retrieve document 8  
  search criteria 8  
  server print document 8  
  update document 8  
  view annotations 9

## G

GET method 9  
GIF attachments 56  
GIF documents  
  media type 52  
  MIME content type 52

## H

help 207  
host name 36  
HOST parameter 36  
HP-UX  
  installation 20

## I

image directory 42  
Image Web Viewer  
  about 1  
  installing 75  
  requirements 76  
IMAGEDIR parameter 42  
inactivity time out 45  
installation

  AFP Web Viewer 75  
  AFP2HTML Java applet 75  
  AIX 19  
  applets 75  
  ARSWWW.INI file 35

installation (*continued*)

  CGI 25  
  customizing 193  
  HP-UX 20  
  Image Web Viewer 75  
  Java applets 75  
  Java servlet 27  
  line data Java applet 75  
  Linux 21  
  plug-ins 75  
  requirements 15  
  servlet 27  
  Solaris 21  
  user PC 75  
  user-defined files 193  
  Windows server 22  
INSTALLDIR parameter 47, 49, 174

## J

Java AFP2HTML applet  
  requirements 76  
Java AFP2HTML viewer  
  about 6  
Java API  
  programming guide 119  
  reference 117  
Java applets  
  about 1, 6  
  APPLETCACHEDIR parameter 38  
  directory 49  
  enabling 49  
  installing 75  
  large object support 47, 188  
  requirements 76  
Java line data applet  
  requirements 76  
Java line data viewer  
  about 6  
  configuring 161  
  ODApplet.jre.path.IE parameter 64  
  ODApplet.jre.path.NN parameter 64  
  ODApplet.jre.version parameter 64  
  ODApplet.version parameter 64

Java servlet

  deploying 27  
  reference 115

Java Virtual Machine 76

JFIF documents

  media type 53  
  MIME content type 53

JVM 76

## L

language 43  
LANGUAGE parameter 43

large objects 47, 188  
line data documents  
  converting 62  
  media type 53  
  MIME content type 53  
  viewing 62  
line data Java applet  
  about 1  
  APPLETCACHEDIR parameter 38  
  installing 75  
line data viewer  
  configuring 161  
  ODApplet.jre.path.IE parameter 64  
  ODApplet.jre.path.NN parameter 64  
  ODApplet.jre.version parameter 64  
  ODApplet.version parameter 64  
LINEVIEWING parameter 62  
links 64  
Linux  
  installation 21  
local directory  
  Java API 125  
log files 68, 207  
LOG parameter 68, 207  
LOGDIR parameter 68  
logging 68, 207  
logoff  
  API 95  
  function description 8  
  parameters 95  
  sample function call 96  
logon  
  API 97  
  delimited ASCII output 202  
  function description 8  
  parameters 97  
  sample function call 98  
LOGON.HTM 71

**M**

mapping AFP fonts 199  
MAXHITS parameter 63  
maximum hits 63  
media type/subtype 50  
messages 43  
  delimited ASCII output 205  
Metacode/DJDE documents  
  converting 175  
  viewing 175  
METAVIEWING parameter 175  
method attribute of form tag 9  
MIME content type 50, 130  
MIMETYPES section 50

## N

NLS 42, 43  
no HTML output 56, 201  
NOHTML section 56  
NOLINKS parameter 64  
notes  
  *See annotations*

## O

ODApplet.jre.path.IE parameter 64  
ODApplet.jre.path.NN parameter 64  
ODApplet.jre.version parameter 64  
ODApplet.version parameter 64  
ODCallback 147  
ODCriteria  
  documents, updating 155  
  name 130  
  operands 130, 135, 137  
  search values 130, 135, 137  
  updating a document 155  
ODCriteria.getFixedValues 137  
ODCriteria.getName 130  
ODCriteria.getOperand 130, 135  
ODCriteria.getType 137  
ODCriteria.getValidOperands 137  
ODCriteria.getValues 137  
ODCriteria.setOperand 137, 155  
ODCriteria.setSearchValue 130, 155  
ODCriteria.setSearchValues 130, 135, 137  
ODFolder  
  application groups 128  
  cancelling a search 135  
  closing 128, 130, 135  
  criteria 130, 135, 137  
  description 130  
  display order 130, 143  
  document, printing 148  
  document, retrieving 145  
  message 130  
  name 130, 143  
  printing documents 148  
  retrieve document 145  
  searching 130, 135, 137, 143, 145  
ODFolder.close 128, 130, 135, 145  
ODFolder.getApplGroups 128  
ODFolder.getCriteria 130, 135, 137  
ODFolder.getDescription 130  
ODFolder.getDisplayOrder 130, 143  
ODFolder.getName 130, 143  
ODFolder.getNumApplGroups 128  
ODFolder.getSearchMessage 130  
ODFolder.printDocs 148  
ODFolder.retrieve 145  
ODFolder.search 130, 135, 143, 145

ODHit  
 annotations 151, 153  
 display value 143  
 display values 130  
 document list 143  
 document location 130  
 document type 130  
 document, retrieving 145  
 document, updating 155  
 MIME content type 130  
 notes 151, 153  
 retrieve document 145  
 updating documents 155  
 ODHit.addNote 153  
 ODHit.getDisplayValue 130, 143, 155  
 ODHit.getDisplayValues 130  
 ODHit.getDocId 130, 145  
 ODHit.getDocLocation 130  
 ODHit.getDocType 130  
 ODHit.getMimeType 130  
 ODHit.getNotes 151, 153  
 ODHit.retrieve 145  
 ODHit.update 155  
 ODNote  
 annotations 151, 153  
 color 151  
 date 151  
 group name 151  
 page 151  
 position 151  
 text 151  
 time 151  
 userid 151  
 ODNote.getColor 151  
 ODNote.getTime 151  
 ODNote.getGroupName 151  
 ODNote.getOffsetX 151  
 ODNote.getOffsetY 151  
 ODNote.getPageNum 151  
 ODNote.getText 151  
 ODNote.getUserid 151  
 ODNote.isOkToCopy 151, 153  
 ODNote.isPublic 151, 153  
 ODNote.setGroupName 153  
 ODNote.setText 153  
 ODServer  
 application name 125  
 cancelling a search 135  
 changing passwords 158  
 connecting to 125  
 connecting to a server 124  
 connection type 125  
 disconnecting from 125  
 document, retrieving 145  
 folder description 141

ODServer (*continued*)  
 folder name 141  
 folder, opening 145  
 local directory 125  
 open folder 145  
 opening a folder 137  
 password 125, 158  
 port 125  
 printers 148  
 retrieve document 145  
 server 125  
 server printers 148  
 setting and getting passwords 125  
 setting and getting userids 125  
 setting passwords 158  
 userid 125  
 ODServer.cancel 135  
 ODServer.changePassword 158  
 ODServer.getConnectType 125  
 ODServer.getFolderNames 141  
 ODServer.getFoldersDescription 141  
 ODServer.getLocalDir 125  
 ODServer.getNumFolders 141  
 ODServer.getPassword 125  
 ODServer.getPort 125  
 ODServer.getServerName 125  
 ODServer.getServerPrinters 148  
 ODServer.getUserId 125  
 ODServer.logoff 125  
 ODServer.logon 125  
 ODServer.openFolder 137, 145  
 ODServer.retrieve 145  
 ODServer.setApplicationName 125  
 ODServer.setConnectType 125  
 ODServer.setLocalDir 125  
 ODServer.setPassword 125  
 ODServer.setPort 125  
 ODServer.setServer 125  
 ODServer.setUserid 125  
 ODServer.terminate 125  
 OnDemand Internet Connection  
 about 1  
 OnDemand server options  
 @SRV@\_DEFAULT section 35  
 @SRV@\_server section 36  
 defaults 35  
 HOST parameter 36  
 parameters 36  
 PORT parameter 35, 37  
 PROTOCOL parameter 36, 37  
 operands, Java API 130  
 output delimiters 56  
 overview 1

## P

package hierarchy, Java 119  
parameter file  
  Xenos transforms 178  
parameters  
  @SRV@\_DEFAULT section 35  
  @SRV@\_server section 36  
  ADDEXTRACTION 58  
  ADDFIELDSTODOCID 58  
  ADDNOTES 59  
  AFP2HTML section 46  
  AFP2PDF section 48  
  AFPVIEWING 59, 174  
  APPLETCACHEDIR 38  
  APPLETDIR 38  
  ATTACHMENT IMAGES section 55  
  AUTODOCRETRIEVAL 60  
  BEGIN 56  
  CACHEDIR 39  
  CACHEDOCS 39  
  CACHEMAXTHRESHOLD 40  
  CACHEMINTHRESHOLD 40  
  CACHESIZE 40  
  CACHEUSERIDS 41  
  CODEPAGE 42  
  CONFIGFILE 47, 49, 174  
  CONFIGURATION section 37  
  DOCSIZE 42  
  EMAILVIEWING 61  
  ENCRYPTCOOKIES 61  
  ENCRYPTURL 62  
  END 57  
  FOLDERDESC 62  
  HOST 36  
  IMAGEDIR 42  
  INSTALLDIR 47, 49, 174  
  LANGUAGE 43  
  LINEVIEWING 62  
  LOG 68, 207  
  LOGDIR 68  
  MAXHITS 63  
  METAVIEWING 175  
  NOLINKS 64  
  ODApplet.jre.path.IE 64  
  ODApplet.jre.path.NN 64  
  ODApplet.version 64  
  PORT 35, 37  
  PROTOCOL 36, 37  
  REPORTSERVERTIMEOUT 45  
  SECURITY section 45  
  SEPARATOR 57  
  SERVERACCESS 46  
  SERVERPRINT 64  
  SERVERPRINTERS 65  
  SHOWDOCLOCATION 65

## parameters (*continued*)

  TEMPDIR 44  
  TEMPLATEDIR 45  
  USEEXECUTABLE 48, 49  
  VIEWNOTES 66  
  XENOS section 173

## passwords

  Java API 125, 158

## PCX documents

  media type 54  
  MIME content type 54

## PDF documents

  media type 54  
  MIME content type 54

## permissions 17

## plug-in

  installing 75

## plug-ins

  about 1

## port

  Java API 125

## port number 35, 37

## PORT parameter 35, 37

## POST method 9

## preparing to use the OnDemand Internet

  Connection 1

## print document

  API 99

  function description 8

  Java API 148

  parameters 99

  sample function call 102

## printing

  Java API 148

  server 64, 65

## privileges 17

## problem determination 207

## programming guide

  API 119

  Java API 119

## PROTOCOL parameter 36, 37

## protocols 36, 37

## Q

## query results 63

## R

## reference

  API 85, 115, 117

  CGI API 85

  Java API 117

  Java servlet 115

  servlet 115

## REPORTSERVERTIMEOUT parameter 45

## requirements 15

requirements (*continued*)

- cookies 76
- Java Virtual Machine 76
- retrieve document
  - API 103
  - function description 8
  - parameters 103
  - sample function call 106
  - Xenos transforms 183
- retrieving
  - documents 60
- retrieving a document 145

## S

- sample applications 71
- sample code 119
- script file
  - Xenos transforms 181
- search criteria
  - API 107
  - delimited ASCII output 202
  - function description 8
  - Java API 130, 137
  - parameters 107
  - sample function call 109
- search values, Java API 130
- searching a folder 130, 135, 137, 143
- security 9, 45, 61, 62
- SECURITY section 45
- SEPARATOR parameter 57
- server
  - Java API 125
- server access list 46
- server print
  - API 99
  - enabling 64, 65
  - function description 8
  - Java API 148
  - parameters 99
  - sample function call 102
- server security 9, 45
- SERVERACCESS parameter 46
- SERVERPRINT parameter 64
- SERVERPRINTERS parameter 65
- servlet
  - deploying 27
  - reference 115
- setting passwords 158
- SHOWDOCLOCATION parameter 65
- Solaris
  - installation 21
- system environment, Java API 121

## T

- TCP/IP communications protocol 36, 37
- TEMPDIR parameter 44
- template file 73
- TEMPLATE.HTM 73
- TEMPLATEDIR parameter 45
- temporary storage 44
- temporary work directory 44
- TIFF documents
  - media type 54
  - MIME content type 54
- time out 45
- tracing and diagnostic information 122
- tracing problems 207
- transforms
  - AFP2PDF 189
  - AFP2WEB 185
  - Xenos 167
- TrueType fonts
  - mapping AFP fonts to 199
- TXT attachments 56

## U

- update document
  - API 110
  - function description 8
  - Java API 155
  - parameters 110
  - sample function call 111
- USEEXECUTABLE parameter 48, 49
- user-defined files
  - installing 193
- userids
  - cache storage 41
  - Java API 125

## V

- view annotations
  - API 112
  - delimited ASCII output 204
  - function description 9
  - parameters 112
  - sample function call 113
- VIEWNOTES parameter 66

## W

- Web server options
  - AFP2HTML section 46
  - AFP2PDF section 48
  - AFP2PDF Transform 48
  - AFP2WEB Transform 46
  - APPLETDIR parameter 38
  - ATTACHMENT IMAGES section 55
  - BEGIN parameter 56
  - browsers 57, 66, 174, 175

Web server options (*continued*)  
  CACHEDIR parameter 39  
  CACHEDOCS parameter 39  
  CACHEMAXTHRESHOLD parameter 40  
  CACHEMINTHRESHOLD parameter 40  
  CACHESIZE parameter 40  
  CACHEUSERIDS parameter 41  
  CODEPAGE parameter 42  
  CONFIGFILE parameter 47, 49, 174  
  CONFIGURATION section 37  
  debug 67  
  default browser 57, 174, 175  
  END parameter 57  
  IMAGEDIR parameter 42  
  INSTALLDIR parameter 47, 49, 174  
  LANGUAGE parameter 43  
  MIMETYPES section 50  
  NOHTML section 56  
  REPORTSERVERTIMEOUT parameter 45  
  SECURITY section 45  
  SEPARATOR parameter 57  
  SERVERACCESS parameter 46  
  TEMPDIR parameter 44  
  TEMPLATEDIR parameter 45  
  USEEXECUTABLE parameter 48, 49  
  XENOS section 173

Windows server  
  installation 22

## X

  XENOS section 173  
  Xenos transforms  
    about ix, 167  
    APIs for ODWEK 183  
    ARSWWW.INI file 173  
    ARSXENOS.INI file 176  
    CONFIGFILE parameter in ARSWWW.INI file 174  
    INSTALLDIR parameter in ARSWWW.INI file 174  
    installing 12, 17  
    ODWEK APIs 183  
    parameter file 178  
    script file 181  
  XENOS section in ARSWWW.INI file 173





**IBM**<sup>®</sup>

Program Number: 5697-G34

SC27-1000-02



Spine information:



IBM Content Manager  
OnDemand for Multiplatforms

Web Enablement Kit Implementation Guide  
Version 7.1