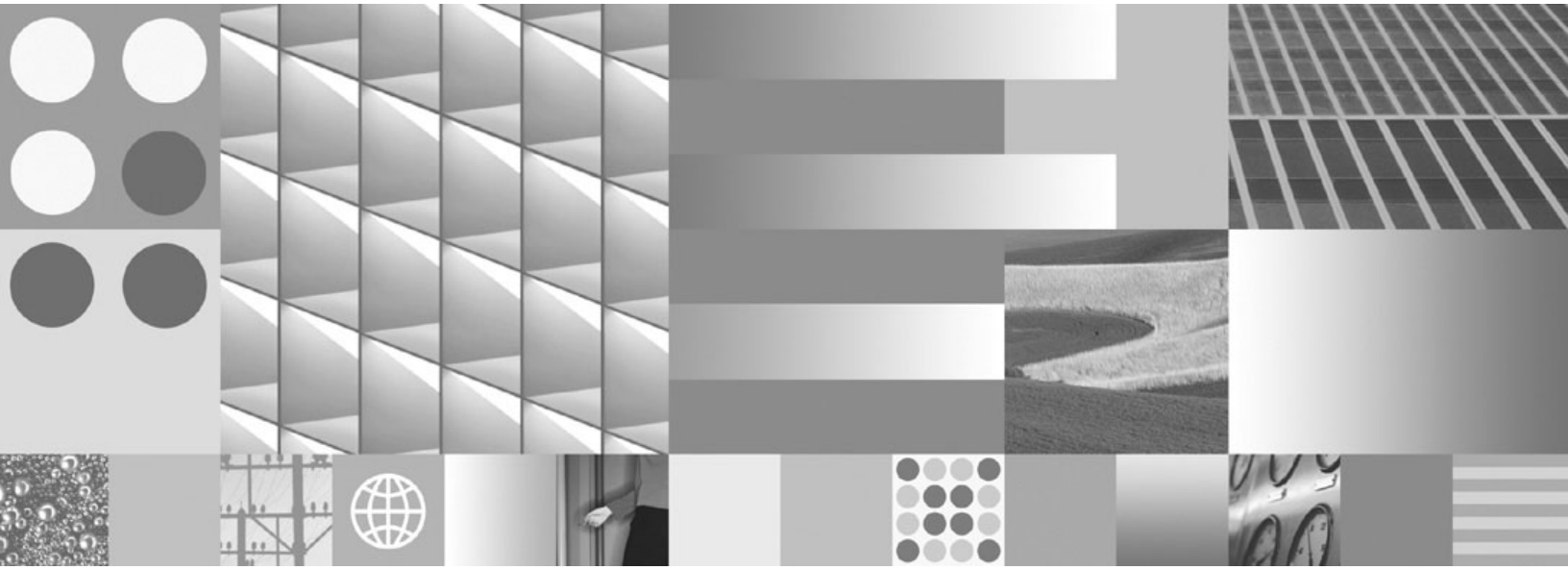
Content Manager OnDemand for i

Version 7 Release 1

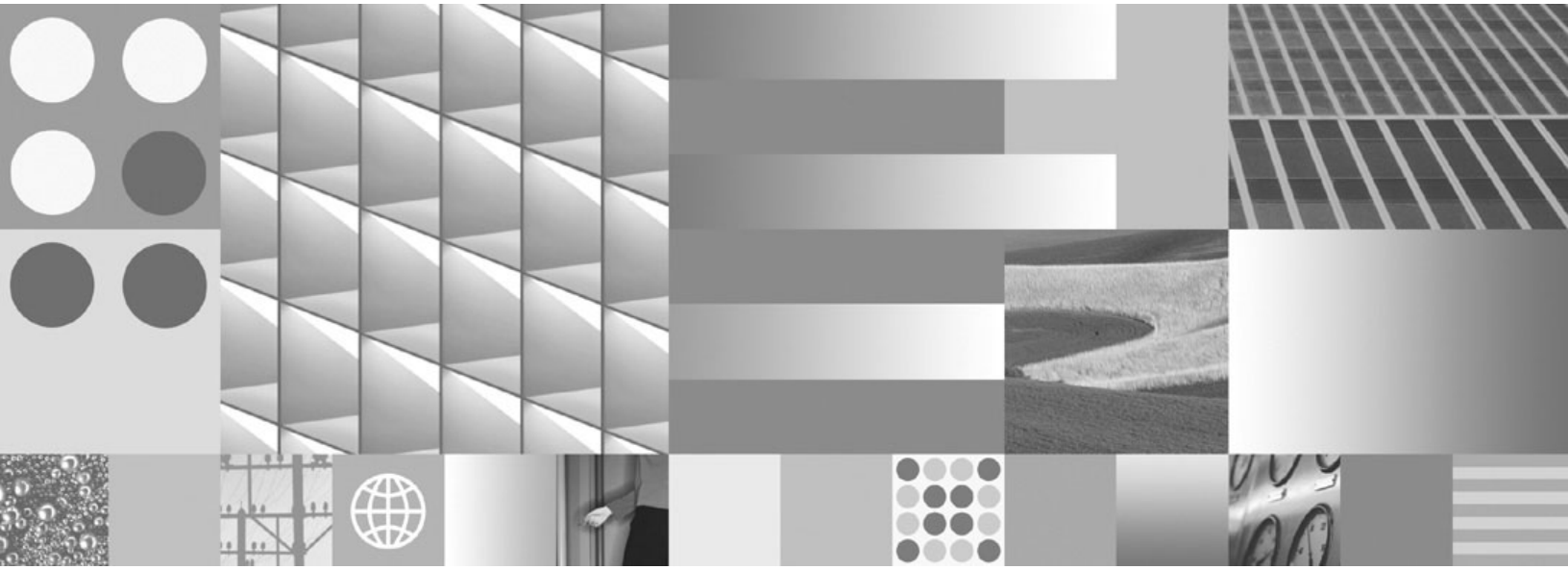


Common Server ODWEK Installation and Configuration Guide

SC19-2791-00

Content Manager OnDemand for i

Version 7 Release 1



Common Server ODWEK Installation and Configuration Guide

SC19-2791-00

Note
Before using this information and the product it supports, read the information in "Notices" on page 177.

This edition applies to version 7, release 1 of IBM Content Manager OnDemand for i (product number 5770-RD1) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About IBM OnDemand for i Common Server Web Enablement Kit -- Installation and Configuration Guide (SC19-2791)

This book provides information that you can use to plan for, install, configure, and use IBM® Content Manager OnDemand for i Version 7 Release 1 Common Server (OnDemand) Web Enablement Kit.

Who should read this book

This book is intended primarily for system administrators who need to implement, install, and maintain the OnDemand Web Enablement Kit (ODWEK) software and applications. It can also be used by programmers who need to integrate OnDemand with Web applications.

How this book is organized

This book provides the information that you need to install and configure ODWEK and plan for users to access data from an IBM Content Manager OnDemand for i Common Server system with a Web browser. This publication contains the following sections:

Prerequisite and related information

Use the IBM i Information Center as your starting point for looking up IBM i technical information.

You can access the Information Center two ways:

- From the following Web site: `http://www.ibm.com/systems/i/infocenter/`
- From CD-ROMs that ship with your IBM i order:
  IBM i Information Center SK3T-4091-07

The IBM i Information Center contains:

- Updated and new information, including IBM i installation and upgrades, data migration, service and troubleshooting, availability, System i® integration, connecting to System i, database, Linux®, WebSphere®, Java™, CL commands, system APIs, and manuals.
- Advisors and other interactive tools to assist in troubleshooting and configuring your IBM i software.

Other information available on the World Wide Web

More IBM i information is available on the World Wide Web. You can access general information from the IBM i home page, which is at the following Web site: `http://www.ibm.com/systems/i/`

To access workshops on advanced IBM i functions, use the Technical Studio, located at: `http://www.redbooks.ibm.com/tstudio/`

Worldwide, you can read about, select, order and take delivery of IBM i program temporary fixes (PTF) over the Internet. IBM i Internet PTFs (downloads) and Preventive Service Planning (PSP) information are available at the following Internet location: `http://as400service.ibm.com`

Product documentation has been moved from the library page to the support page on the CM OnDemand for i product Web site. To see a list of all available OnDemand for i product documentation, go to http://www.ibm.com/software/data/ondemand/400/support.html. Look in the left hand column, under the ″Self help″ section, ″Learn″ subheading.

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 you are familiar with Hypertext Markup Language (HTML), Common Gateway Interface (CGI) and Java programming, that you provide content for Web pages, that you know how to configure and operate a 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 use the Java AFP2HTML Viewer, you must obtain the AFP2WEB Transform, and install and configure it on the server on which your ODWEK application resides. See your IBM representative for more information about the AFP2WEB Transform. You must also provide configuration options for the Advanced Function Presentation (AFP) documents and resources you plan to process with the AFP2WEB Transform. See Appendix E, “AFP to HTML transform,” on page 151 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, you must obtain the AFT2PDF Transform, and install and configure it on the Web server. See you IBM representative for more information about the AFP2PDF Transform. You must also provide configuration options for the AFP documents and resources you plant to process with the AFP2PDF Transform. See Appendix F, “AFP to PDF transform,” on page 155 for more information about the configuration file.

OnDemand Information Center

In addition to the IBM i Information Center (previously mentioned), be sure to visit the OnDemand Information Center, which focuses only on information pertaining to CM OnDemand. The OnDemand Information Center provides fast, online centralized access to product information. It is a task-based documentation repository that allows you to search across the entire product library for commands, error codes, or any other topic of interest. You can bookmark pages of interest or common reference, allowing them to easily be retrieved for future reference.

To access the OnDemand Information center, go to http://publib.boulder.ibm.com/infocenter/cmod/v8r4m1//index.jsp

Accessibility information for OnDemand

For complete information about accessibility features that are supported by this product, see the *IBM Content Manager OnDemand for i: Common Server Administration Guide*.

IBM i Navigator

IBM i Navigator is a powerful graphical interface for managing your IBM i servers. IBM i Navigator functionality includes system navigation, configuration, planning capabilities, and online help to guide you through your tasks. IBM i Navigator makes operation and administration of the server easier and more productive and is the only user interface to the new, advanced features of IBM i. It also includes Management Central for managing multiple servers from a central system.

You can find more information on IBM i Navigator in the IBM i Information Center and at the following Web site: `http://www.ibm.com/servers/eserver/iseries/navigator/`

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. Visit the IBM Data Management Online Reader's Comment Form (RCF) page at www.ibm.com/software/data/rcf.

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 i: Common Server ODWEK Installation and Configuration Guide* contains new technical information. There might be some instances where changes were made, but change bars are missing. Significant changes to note are:

New functions

- New commands were added to replace program calls for a number of different OnDemand functions:
  - Use the new Create Instance for OnDemand (CRTINSTOND) command instead of calling the QRLMINST program to create new OnDemand instances. The new command provides additional parameters beyond what the QRLMINST program provided. The command allows you to specify Port, Autostart, Security, and Auxiliary Storage Pool (ASP)-related parameters on the command so that the `ars.ini` and `ars.cfg` configuration files do not need editing in many cases. Note that the program call interface is no longer supported. The command interface is the only supported interface in version 7.1.
  - Use the new Merge Spooled Files (MRGSPLFOND) command instead of the old MRGSPLFOND sample command (shipped in previous releases) or calling the QRLMQMRGF program to merge small spooled files into one larger file before archiving. The new MRGSPLFOND command shipped with version 7.1 contains new and enhanced parameters that provide significantly more function than the previous sample command. Any of your existing programs that use the previous sample command must be changed to use the version 7.1 parameters.
  - Use the Migrate Media (MGRMEDRDAR) command instead of calling the QRLCSFAMMF program to migrate OnDemand data from one media type to another. (This command was available in OnDemand version 6.1, but is listed here to note that the program call is no longer supported. Only the command interface is supported in version 7.1.)
  - Use the Change Policy Level Date (CHGPLDOND) command instead of calling the QRLCASMCLD program if you need to change migration policy level dates for archived data. (This command was available in OnDemand version 6.1, but is listed here to note that the program call is no longer supported. Only the command interface is supported in version 7.1.)
- A new System i Navigator function has been added to replace the program call for setting up Network File System (NFS) disk pools for use with OnDemand.
  - Use the new Network File System (NFS) panels of the OnDemand System i Navigator plug-in instead of calling the QRLCASMNFS program.
  - See the Content Manager OnDemand for i: Common Server Administration Guide for instructions on using NFS with OnDemand.
- Enhanced retention management capabilities are now available as a separately priced feature of OnDemand for i at 7.1, which provide new hold and release functions for the OnDemand end-user client, and HOLD-ADD and HOLD-RELEASE parameters for the ARSDOC API, and the ODWEK ODHold and ODHit Java APIs. Expiration for held documents can be prevented until the

hold is released. See the new ARS_SUPPORT_HOLD entry for the ARS.CFG file in the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide.

IMPORTANT NOTE: When using enhanced retention management, the OnDemand Disk Storage Manager (DSM) must be in complete control of expiration processing. If you are using the OnDemand Archive Storage Manager (ASM) or Tivoli® Storage Manager, you must disable the ability for either of these storage managers to expire data. For example, in ASM, this means disabling or deleting any Expire levels in migration polices that are used with OnDemand application groups that have enhanced retention management enabled.

- All of the storage management data was moved from the QUSRRDARS library to the individual instance library with which it is associated. A few objects remain in the QUSRRDARS library. This change removes the need to move the QUSRRDARS library to an Independent Auxiliary Storage Pool (IASP).
- An instance library can be located in a user Auxiliary Storage Pool (ASP 2 through 32).
- A new System Load application group was added, which contains one entry for each input file that is loaded into OnDemand. Using the OnDemand client, you can verify what data was successfully loaded into OnDemand.
- A new Java-based ARSSUPPORT utility is available to gather diagnostic information such as log entries. This tool is helpful when you need to report problems to IBM Software Support. See the IBM Content Manager OnDemand for i: Common Server Administration Guide for details.
- The Xerces2 Java Parser Version 2.6.2 is included in this version of Content Manager OnDemand for i. See the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide for instructions on using this version of the parser.
- Numerous OnDemand Web Enablement Kit (ODWEK) changes are included in version 7.1. Look for change bars throughout the IBM Content Manager OnDemand for i: Common Server ODWEK Installation and Configuration Guide to identify the changes.
- Updates were made to the batch administration function (ARSXML API), including new, renamed, and removed attributes. For example, the name attribute value of _ALL was removed for update and delete. (_ALL is still supported for export operations.) See the IBM Content Manager OnDemand for i: Common Server Administration Guide for details.
- New sample programs were added and existing samples have been updated. See QSAMPLES2 source file in the QRDARS library.
- Documentation was added for Archive Storage Manager (ASM)-based expiration, which might eliminate the need to run Disk Storage Manager (DSM). This capability became available in OnDemand 6.1, but is listed here to note that additional documentation was added to the IBM Content Manager OnDemand for i: Common Server Administration Guide on this topic. Also see the IMPORTANT NOTE above, under enhanced retention management, regarding existing OnDemand storage management functions and the new retention management capabilities.
- The AFP2WEB Transform (which includes both AFP2PDF and AFP2HTML) that allows AFP data to be viewed using an OnDemand Web Enablement Kit (ODWEK) interface is available as a separately priced feature of OnDemand for i at version 7.1. See your IBM representative or business partner for more information on the AFP2WEB Transform.

- The PDF indexer supports resource grouping and removing unused resources. After you enable resource grouping, common resources across documents in a single input file are grouped and stored as a single object, and unused resources from an input file can be removed before indexing. New PDF indexer parameters (RESTYPE and REMOVERES) have been added to support this new capability. For more information, see the IBM Content Manager OnDemand for i: Common Server Indexing Reference.
- Integration with FileNet® P8 platform is now supported as a separately priced feature at version 7.1, allowing you to send Content Manager OnDemand for i metadata to FileNet and take advantage of FileNet's Business Process Management (BPM) and FileNet Records Manager (RM) capabilities. For more information on this Content Federation Services for OnDemand feature, see the new ARS_SUPPORT_CFSOD entry for the ARS.CFG file in the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide and the new CFSOD-FED function for ARSDOC in the IBM Content Manager OnDemand for i: Common Server Administration Guide
- Documentation on various data areas that control OnDemand functions was consolidated and added to the IBM Content Manager OnDemand for i: Common Server Administration Guide.
- A new column header was added to the OnDemand post processor program input file. This new header begins with the "<" character and ends with the ">" character. Your post processor programs must be tested to ensure compatibility with this addition.

Command enhancements

- A new *DIR2 monitor type was added to the Start Monitor for OnDemand (STRMONOND) command, triggered by .ARD files (like the ARSLOAD API) instead of the .IND files like the current *DIR monitor type. Note that the *DIR monitor type on the End Monitor for OnDemand (ENDMONOND) command ends either type of monitor, regardless of whether it was started as a *DIR or *DIR2 monitor.
- Two new parameters were added to the Start Monitor for OnDemand (STRMONOND) command. The End server (ENDSVR) parameter allows you to end the instance server when monitor ends. The Monitor job name (JOB) allows you to specify the name you want to use for the monitor job.
- Two new parameters were added to the Add Report to OnDemand (ADDRPTOND), Print Report from OnDemand (PRTRPTOND), Start Monitor for OnDemand (STRMONOND), Remove Report from OnDemand (RMVRPTOND), Start Import into OnDemand (STRIMPOND), Start Archive Storage Mgmt (STRASMOND), Start Disk Storage Management (STRDSMOND), Merge Spooled Files (MRGSPLFOND), Change Policy Level Date (CHGPLDOND), and Migrate Media (MGRMEDRDAR) commands.
  - Using INSTANCE(*DFT), which is the new default rather than INSTANCE(QUSROND), a default instance name can be retrieved from a data area named QDFTINST so that the instance name does not need to be explicitly specified on each command. See online help and the IBM Content Manager OnDemand for i: Common Server Administration Guide for details on the QDFTINST data area.
  - The Start server (STRSVR) parameter can be specified to start the instance server when the command is executed. (The STRSVR parameter does not apply for the CHGPLDOND command.) See online help for more information.

- A new VALIDATE parameter has been added to the Start Disk Storage Management (STRDSMOND) command to ensure that all disk storage files are correctly linked and contain the proper file permissions.
- The Maximum number of monitors to start parameter on the Start Monitor for OnDemand (STRMONOND) command when starting an output queue monitor was reduced from 99 to 9 to avoid locking delays that can occur if a large number of monitors are started against the same output queue.

Additional functions added previously and included in OnDemand version 7.1

- The INSTANCE parameter of the Start TCP/IP Server (STRTCPSVR) and End TCP/IP Server (ENDTCPSVR) commands is now supported when specifying *ONDMD (for OnDemand) for the SERVER parameter. You can name a specific instance to start, or use one of three special values (*DFT, *ALL, *AUTOSTART). Note that calling the QRLMCTL program to start or end an instance is still supported, but using the STRTCPSVR and ENDTCPSVR commands is recommended. See the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide and online help for more information.
- Lightweight Directory Access Protocol (LDAP), an open industry standard that shares information between distributed applications on the same network, can be used to manage basic login authentication directly on the server. See the IBM Content Manager OnDemand for i: Common Server Administration Guide for more information.
- Support for the Internet Protocol Version 6 (IPv6) addressing format, which is a revision of the IPv4 addressing scheme for TCP/IP, is included in the 7.1 version of Content Manager OnDemand for i. See the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide for more information.
- *ASM can now be specified as the target (TGT) destination on the Migrate Media MGRMEDRDAR command, allowing data to be moved from the Spool File Archive architecture (managed by RMC) to the Common Server architecture (managed by ASM).
- The PDF indexer now allows part of the input file name to be used as an index value when storing data using the ARSLOAD API.

When you upgrade to version 7.1, be aware of the following

- If you are upgrading from a previous version of OnDemand, you must be running OnDemand server version 7.1.2.8 or higher prior to upgrading to Content Manager OnDemand for i Version 7 Release 1. To determine your current server version, see the IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide for instructions.
- Version 7.1 of Content Manager OnDemand for i does not support OnDemand client software prior to version 7.1.2.0. This includes, but is not limited to, the OnDemand Windows® (end-user) client, ODWEK CGI/Servlet/Java APIs, CICS®, and II4C (eClient).
- The OnDemand Administrator client must be at the same version or higher as the OnDemand server. For version 7.1 of Content Manager OnDemand for i, the OnDemand Administrator client must be at version 8.4.1.3 or higher.
- Starting an instance or starting the Archive Storage Management (ASM) process for the first time after your upgrade to version 7.1 takes longer than usual due to file conversions and movement of instance-specific data and objects from QUSRRDARS into the instance libraries.

  - Do not end the server job or ASM if you are concerned that it is not progressing.
  - Status messages are written to the job log during the file conversions and the data movement from QUSRRDARS to the instance library which you can check to confirm that the job is making progress.
- If a job description (object type *JOBD) exists in the QUSRRDARS library that matches an OnDemand instance name defined to your system, the job description is moved from QUSRRDARS to the instance library. When OnDemand looks for a job description to use to start the instance server, OnDemand looks first for a *JOBD object that is named the same as the instance name in the instance library. If one is not found, OnDemand looks for a *JOBD object that is named the same as the instance name in the QUSRRDARS library. If one is not found, OnDemand uses the QOND400 job description located in the QRDARS library.
- The following program calls were replaced by new commands or a new System i Navigator interface in version 7.1:
  - Replace the MRGSPLF sample program and the previous MRGSPLFOND sample command with the new MRGSPLFOND command.
  - Replace the QRLMINST program with the CRTINSTOND command.
  - Replace the QRLCASMNFS program with the System i Navigator plug-in interface.
  - Replace the QRLCSFAMMF program with the MGRMEDRDAR command.
  - Replace the QRLCASMCLD program with the CHGPLDOND command.
- The OnDemand Commands (CMDOND) menu, which includes all of the OnDemand for i Common Server commands, is the only 5250 menu that is shipped with OnDemand version 7.1. The following OnDemand 5250 menus were removed:
  - CMDRDAR
  - ONDEMAND
  - RDARS
  - RDARSDEF
  - RDARSM
  - RDARSOBJ
  - RDARSRLA
  - RDARSRPT
  - RDARSUTL
- If you installed fonts for use with the PDF indexer, you should verify the location of the fonts and move them to the directories required by the PDF indexer if necessary. For specific details, see the IBM Content Manager OnDemand for i: Common Server Indexing Reference.
- The QPRLR133 printer file in the QRDARS library is no longer part of the product, because it was a component of Spool File Archive, which is no longer available. The printer file is removed when version 7.1 is installed.
- The unmount file system program (QRLCASMUFS) no longer supports *ALL for the instance name, because all instance-specific files are now in the individual instance libraries. You must name a specific instance when you call the program.

- Before using the latest version of the OnDemand Web Enablement Kit (ODWEK) CGI/Servlet, you must delete all of the files from the Web Enablement Kit cache and temp directories. The directories are specified by the CACHEDIR and TEMPDIR entries in the arswww.ini file.
- Existing Spool File Archive implementations must be migrated from the Spool File Archive environment to Common Server before the system on which they are running is upgraded to 7.1. Content Manager OnDemand releases 5.3 and 5.4 included the Common Server environment, as well as the legacy environments of Spool File Archive, AnyStore, Record Archive, and Object Archive. All of these environments are fully supported through and including i5/OS® 5.4.

  As stated in IBM Announcement Letter #206-030 dated February 14, 2006, 5.4 was the last release that Spool File Archive, AnyStore, Record Archive, and Object Archive would be shipped and supported. Beginning with Content Manager OnDemand 5.3, a Spool File Archive migration utility was available as part of the Content Manager OnDemand licensed program product. The migration utility provides the capability to migrate report definitions and indexes from the legacy Spool File Archive environment to the Common Server environment. Detailed information on the migration utility can be found in Appendix A of the IBM Content Manager OnDemand for iSeries®: Common Server Planning and Installation Guide for version 5.4. Existing Spool File Archive implementations must be migrated from the Spool File Archive environment to Common Server before the system is upgraded to version 7.1.

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 or user-written program. For example, you can provide some people with the Uniform Resource Locator (URL) of a Web page that permits them to log on to an OnDemand server; you can provide other people 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 workstation with a Web browser that is being used to access data from an OnDemand server.

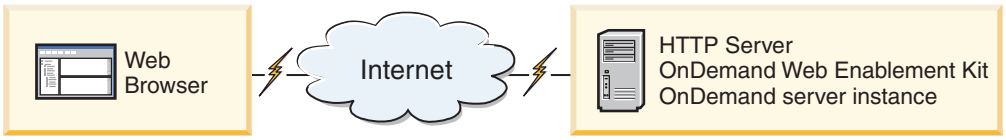


Figure 1. Accessing data stored in OnDemand by using ODWEK

ODWEK can search for and retrieve documents from OnDemand servers that are running IBM Content Manager OnDemand for i Common Server.

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 can use one of the following programming interfaces to control ODWEK:
  - Common Gateway Interface (CGI) program. The CGI program provides a way to access OnDemand data from a Web browser. The CGI program runs on a system that is running an Hypertext Transfer Protocol (HTTP) server, such as the IBM HTTP Server.
  - Java servlet. The CGI program provides a way to access OnDemand data from a Web browser. The servlet runs on a Java-enabled HTTP server that is running a Java application server, such as the IBM WebSphere Application Server.
  - Java API. The Java API provides a way to access OnDemand data from a user-written program. The Java API requires Java version 1.4 or later.
- The IBM OnDemand Advanced Function Presentation (AFP) Web Viewer. The AFP Web Viewer lets users search, retrieve, view, navigate, and print AFP documents from a Web browser.
- The IBM OnDemand Image Web Viewer. The Image Web Viewer lets users search, retrieve, view, navigate, and print BMP, GIF, JPEG, PCX, PNG, and TIFF documents from a Web browser.

- The Line Data Java applet. The Line Data applet lets users view line data documents from a Web browser. An administrator enables the Line Data applet by configuring the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API.
- The AFP2HTML Java applet. The AFP2HTML applet lets users view the output generated by the AFP2WEB Transform. The AFP2WEB Transform converts AFP documents and resources into HTML files that can be displayed with the AFP2HTML applet. After installing and configuring the AFP2WEB Transform, an administrator enables the AFP2HTML applet by configuring the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API.

Important: 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 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 ARSWWW.INI file for CGI and Java servlet or ARSWWW.PROPS file for Java API,.and ODWEK programming interface, which other instances cannot access. There are three ODWEK programming interfaces:

- CGI program, an interface between a Web browser and an OnDemand server
- Java servlet, an interface between a Web browser and an OnDemand server
- Java API, a set of methods that may be used to access OnDemand data from a user-written program

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 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.

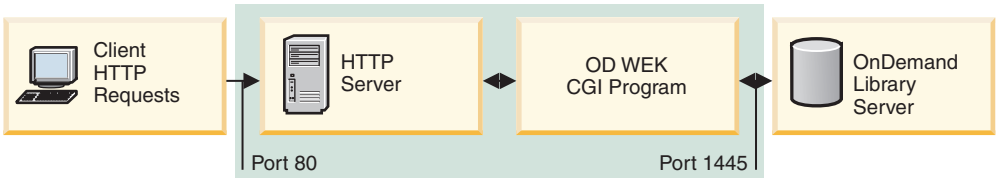


Figure 2. Single instance using CGI interface

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

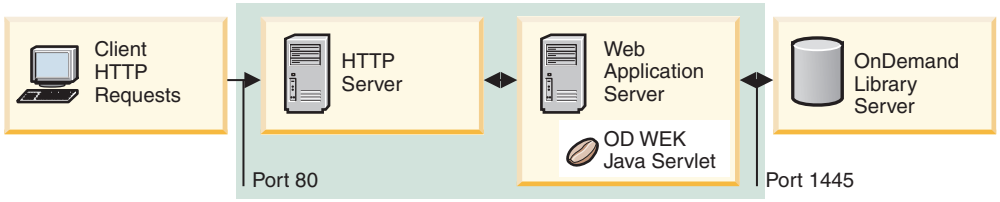


Figure 3. Single instance using Java interface

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

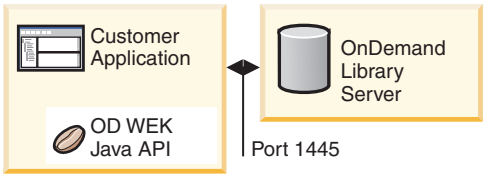


Figure 4. Single instance using Java API interface

You can configure multiple instances of ODWEK on the same system. For CGI and Java servlet, 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. For Java API, the unique port number is specified in your Java code that accesses the ODWEK Java API functions. Each instance also requires its own storage and security. The multiple instance configuration is typically for customers who 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.

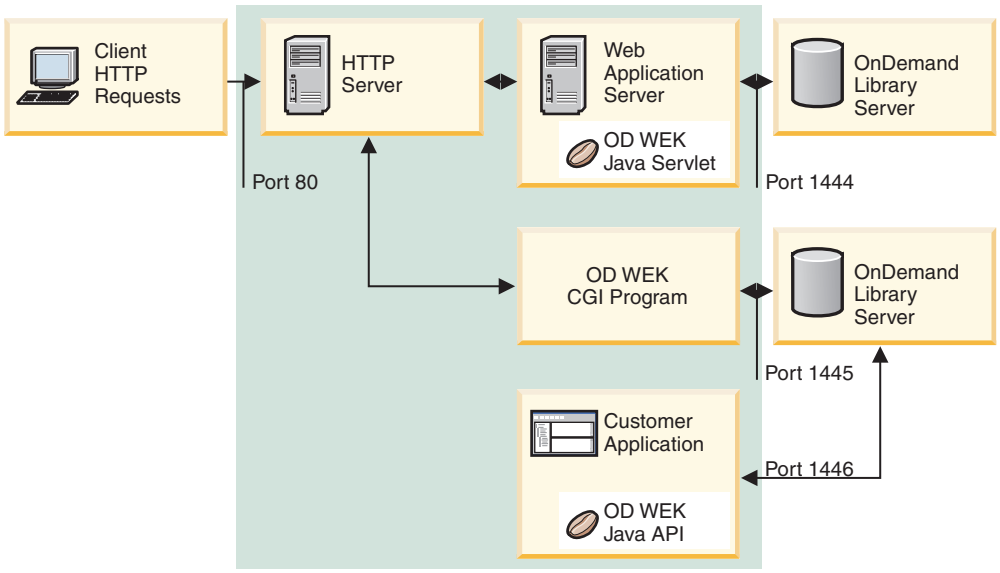


Figure 5. Multiple instance topology

Use cases of ODWEK Java APIs from a business perspective

You can access the data that is stored in OnDemand in two ways: Internet access and intranet access.

To access data through the internet, users (users who are external to the organization) are given access to a specific subset of information. For example, in banking applications, internet users can log on and view current statements that cover the last twelve months. This is the only access that the user has to the OnDemand archive, and the user is presented with a very limited set of menus to obtain this data. Typical internet usage involves tens of thousands of users accessing the OnDemand archive concurrently. The OnDemand architecture lets the system scale to the limits of the available hardware and network resources and enables businesses to grow their system as their usage increases.

To access data through an intranet, users (users who work for an organization) are given access to a wide variety of data based on their user ID access privileges in the OnDemand archive. The user is presented with a selection criteria that allows him or her to access any of the stored data based on the security profile. The flexibility that is provided by the ODWEK Java APIs allows for the design of customized interfaces that are capable of meeting the specific requirements of an organization.

A common customer use case

A common use case across many different business segments involves allowing registered customers to select a range of documents to view. For the banking industry, these documents can be bank statements that can be selected from a predefined number of months. For health insurance companies, the documents can be explanations of benefit statements. For utility companies, the documents can be bills or invoices.

The customer is usually a user who is registered to the company's Web site. The registration process provides credentials, for example, a user ID and password. To gain access to any of the company applications, the user submits the credentials from a Web page such as a portal to be authenticated by the company's Web application. After the user is authenticated, the application restricts what the user can perform when the user navigates the Web site.

OnDemand requires that a user ID be defined for any user who needs access. Administrators cannot create a user ID for each registered Internet user. The user population might reach thousands or millions. Therefore, most applications perform a search on behalf of a user ID that has the permissions to search a specific folder and retrieve a document. The assumption is that if the user was not authenticated, and the constraints are placed on the search values to guarantee unique results, then one or more user IDs can be defined to have full permissions to the folder that is searched.

Regardless of the industry that has OnDemand statements to present to customers, customers can perform two types of transactions:

No user interaction
: One transaction type involves the application displaying a list of documents, usually within some predefined date range. When the user is authenticated, the application chooses a unique key or key combination to ensure that the search results are for that user, and performs the search without any user interaction.

User interaction

Another transaction type involves the user selecting a document from a search results list. The data type of the document affects the way the content is displayed to the users. For example, if the document is stored in OnDemand as an AFP data stream, the user needs an AFP viewer program that is locally installed on the user's machine, or some data conversion needs to take place to provide a data stream that is better for viewing. In many cases, the application transforms the files, such as AFP to PDF, to make it easier for the user to view. The PDF viewer is easy to install. However, the AFP viewer, such as a plug-in, might be more difficult to obtain and manage by the company that provides the Web service.

The steps involved in this use case are:

1. Users sign on to Web site with their credentials. The Web application authenticates the users and retrieves OnDemand connections from a pool of connections.
2. The Web application selects a predefined folder to search.
3. The Web application assigns one or more OnDemand folder field values to perform a search that is unique to each user.
4. The Web application performs a search, for example, by account number and date range, and returns a search results list to a user.
5. The Web application closes the folder and releases the connection back to the pool of connections.
6. The user selects a document to view from the search results list. Optionally, the user is authenticated again before the user is allowed to retrieve the document.
7. The Web application retrieves a OnDemand connection from a pool of connections.
8. The Web application retrieves the document and optionally performs a data transformation before it releases the data to the user.
9. The Web application releases the OnDemand connection back to the pool of connections.

About the viewers

ODWEK provides the following viewers:

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

The AFP Web Viewer and the Image Web Viewer are software programs that extend the capabilities of a Web browser. The AFP Web Viewer lets users view AFP documents. The Image Viewer lets users view BMP, GIF, JPEG, PCX, PNG, and TIFF documents. The viewers can display documents in the browser window. Each viewer adds a toolbar to the top of the display window. Users can add the viewer toolbar to the browser's toolbar. The plug-in toolbar provides controls that can help users work with documents. The people in your organization who plan to use the Web viewers to view documents must install them on their workstations.

The Line Data applet lets users view SCS and line data documents that are stored in OnDemand. The Line Data applet displays line data documents in the browser window and adds a toolbar to the top of the display window. The Line Data applet toolbar provides controls that can help users work with documents. An

administrator enables the Line Data applet by configuring the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API.

The AFP2HTML applet lets users view the output generated by the AFP2WEB Transform. 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 AFP2HTML applet by configuring the ARSWWW.INI file for CGI and Java servlet, or the ARSWWW.PROPS file for Java API. The AFP2HTML applet provides a toolbar with controls that can help users work with documents, including controls for large objects.

One advantage of the applets is that your users never have to install or upgrade software on the workstation to use them, unlike the Web viewers, which must be installed on the workstation. Also, if IBM provides a new version of a Web Viewer, then you must distribute the updated Web Viewer to your 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 client. The client 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 client 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 provided with the product. The LOGON.HTM sample application supports users that are permitted access to several folders. You first modify the LOGON.HTM page with information about your OnDemand server. You 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 client. Make sure that your users 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 Application Programming Interface (API). See Appendix A, "CGI API reference," on page 67 for details.

Add Annotation

The Add Annotation function enables users to add an annotation to the specified document. When later displayed, the annotation will show the annotation text, as well as a date and time stamp and the user who created the annotation. To add an annotation, the user must be given the Annotation Add permission for each application group that contains documents to be annotated. (The Application Group Access permission lets users add annotations.)

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.

Important: The percent sign (%) and colon (:) characters in index data will cause a failure or unpredictable results when retrieving documents in the ODWEK interface.

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 Document Print permission for each application group that contains documents that the user needs to print. (The Application Group Access permission lets users print documents.) At least one server printer must be defined on the OnDemand server.

Update Document

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

View Annotations

The View Annotations function enables users to view the annotations attached to the specified document. To view annotations, the user must be given the Annotation View permission for each application group that contains annotations that the user needs to view. (The Application Group Access permission lets users view annotations.)

Delete Annotations

The Delete Annotations function enables users to delete the annotations attached to a specified document.

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 HTTP 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 HTTP server. For example, many HTTP 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 HTTP server, that requires users to enter a userid and password before accessing the Web pages. However, even though HTTP 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 HTTP 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 client 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.

Important: If you must use the GET method, then you can encrypt the parameters and values by specifying the ENCRYPTURL parameter in the ARSWWW.INI file. See "ENCRYPTURL" on page 38 for more information.

IBM Web Interface for Content Management (WEBi)

IBM Web Interface for Content Management (WEBi) uses ODWEK Java APIs. WEBi is a fully functional Web 2.0 technology based client. You can use WEBi as your only browser or as a second browser implementation in addition to one or more custom developed browser implementations. Like all other Web implementations accessing the OnDemand server, WEBi is built on the ODWEK Java APIs.

Chapter 2. Installation checklist

Setting up your OnDemand Web Enablement Kit environment typically requires that you perform these tasks:

1. Contact IBM Software Support for the latest PTFs for OnDemand. The list of current PTFs can be found in Informational APAR number II14497.
2. Contact IBM Software Support for the latest IBM i HTTP Server Group PTFs. The product number for the HTTP Server is 5770-DG1. Go to http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view and click on the appropriate group PTF number to view the latest list of HTTP Sever PTFs.
3. Contact IBM Support for the latest Database Group PTFs. Go to http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view and click on the appropriate group PTF number to view the latest list of DB2® PTFs.
4. Obtain a copy of the latest OnDemand *Read This First* document from the Web at http://www.ibm.com/software/data/ondemand/400/support.html. You can find it on the "Documentation" page along with the other Version 7 Release 1 documentation. Print and read the entire file before you begin.
5. Check the ODWEK prerequisites. See Chapter 3, "Installing and configuring the HTTP server," on page 13.
6. Install the OnDemand software on the IBM i server. See "Installing on IBM i" on page 14
7. Configure the ARSWWW.INI file for CGI and Java servlet. See "Specifying the ARSWWW.INI file" on page 15
8. Configure the Apache HTTP server. See Appendix G, Appendix G, "HTTP server configuration files," on page 157 for an example of an HTTP server configuration file.
9. Update the QONDADM and QRDARS400 authorization lists, if needed. See Chapter 3, "Other requirements" on page 13.
10. Set up your workstation browser. To do this perform the following tasks:
    a. Download and install the appropriate viewer plug-in files. See Chapter 5, "Installing the Web viewers," on page 51.
    b. To use the newest version of the Line Data Viewer Applet, you must download and install the latest Java Runtime Environment plug-in from http://www.java.com.
    c. Make sure your browser accepts all cookies. Select **Tools > Internet options**, then the Privacy tab.
    d. Make sure UTF-8 is selected for Internet Explorer. Select **Tools > Internet options**, then the Advanced tab, and then make sure **Always send URLs as UTF-8** is selected.
    e. Make sure the Java Runtime Environment is activated. Select **Tools > Internet options**, then Advanded tab, and then look under the Java (Sun) section. Ensure that **Use Java n vx.y.x for <applet> (requires restart)** is selected. The version shown in the panel will reflect the version that you currently have installed for your browser.

Chapter 3. Installing and configuring the HTTP server

This section defines the installation requirements and explains how to install the ODWEK software on the HTTP server and modify the ODWEK configuration file.

You must install the ODWEK software on an IBM i system that is running the current version of the IBM HTTP Server. In addition, if you plan to use the Java servlet, then make sure that you have the current version of the IBM i Web Application Server (WebSphere) up and running.

ODWEK can search for and retrieve documents from OnDemand servers that are running IBM Content Manager OnDemand for i Common Server.

Installation requirements

Before you install the ODWEK software, you should make sure that your system meets the necessary hardware and software requirements. For detailed information on requirements, see http://www.ibm.com/support/docview.wss?rs=152&uid=swg27016180

Other requirements

ODWEK can *cache* (temporarily store) documents on the HTTP server. 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 18 for details.

By default, ODWEK caches data in the `/QIBM/UserData/OnDemand/WWW/CACHE` directory. You can specify a different cache directory by modifying the ARSWWW.INI file. See “CACHEDIR” on page 18 for details.

Make sure that the processes that run ODWEK programs can read from the directory that contains the programs and can write to the cache directory. When ODWEK is installed, all of the objects are secured by authorization list QONDADM and user profiles QTMHHTTP, QTMHHTP1 and QEJBSVR are added to the authorization list with *CHANGE authority. Also, the QRDARS400 authorization list must have QTMHHTTP, QTMHHTP1, and QEJBSVR on it with *USE authority.

ODWEK requires that the end-user browser accept UTF-8 format. In Microsoft® Internet Explorer, select **Tools > Internet options**, then select the Advanced Tab. Under Browsing, select **Always send URLs** as UTF-8.

If you plan to use the AFP2HTML applet, then you must obtain the AFP2WEB Transform and install and configure it on the HTTP server. See your IBM representative for more information about the AFP2WEB Transform. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2WEB Transform. See Appendix E, “AFP to HTML transform,” on page 151 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 and install and configure it on the HTTP server. See your IBM

representative for more information about the AFP2WEB Transform. You must also provide configuration options for the AFP documents and resources that you plan to process with the AFP2PDF Transform. See Appendix F, "AFP to PDF transform," on page 155 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.

Installing on IBM i

Setting up ODWEK typically requires that you perform the following tasks:

1. To install ODWEK, follow the instructions in the book named *IBM Content Manager OnDemand for i: Common Server Planning and Installation Guide* (SC19-2790). The licensed program number is 5770RD1 and the product option is 11.

   Important: The recommended way to install ODWEK is to use the Install licensed programs menu option from the Work with Licensed Programs menu (go licpgm). From the Install licensed programs window, enter a 1 to Add an option, enter 5770RD1 for the Licensed Program and 11 for the Product Option or scroll through the list of Licensed Programs and Product Options until you find ODWEK and enter a 1 before it. *If you install OnDemand with any other method, errors can occur when you attempt to use it.*

2. Recommendation: Order, load, and apply all PTFs available for OnDemand after successful installation of the licensed program. Refer to Informational APAR II14497 for a complete list of OnDemand Version 7 Release 1 PTFs. The informational APAR can be ordered electronically using the SNDPTFORD command, specifying II14497 for the PTF number. Be sure to read the PTF cover letters and follow any special instructions.
3. Load and apply all available PTFs for IBM program product 5770-DG1 (IBM HTTP Server). Current® PTFs for this product are mandatory for ODWEK to function properly.

Your next step

Make sure that you have the current version of the IBM HTTP server up and running on the IBM i system. You will need to configure the HTTP server. See Appendix G, "HTTP server configuration files," on page 157 for an example of an HTTP server configuration file.

If you plan to use the Java servlet, make sure that you have the current version of the IBM i Web Application Server (WebSphere) running. You will need to configure WebSphere. For instructions, see the IBM WebSphere Application Server Documentation Center on the Web at http://www.ibm.com/systems/i/software/websphere/. Follow the links to Installation and Initial Configuration for the appropriate version of WebSphere.

After you install the ODWEK software, configured the HTTP server, and (optionally) configured WebSphere, you can now configure the ODWEK initialization file for your operating environment. See "Specifying the ARSWWW.INI file" on page 15.

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=/tmp/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@_QUSROND]
HOST=Sxxxxxx.mynetwork.com
PORT=1450
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 43 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 can contain the following parameters:

PORT

The TCP/IP port number that OnDemand servers use to communicate with ODWEK. If you do not specify the PORT parameter, then the server uses the port number that is specified for OnDemand in the Service Table (WRKSRVTBLE). If you do not specify the PORT parameter and OnDemand is not listed in the Service Table, then the servers will attempt to use port number `1445`. To specify that the servers use the port number that is specified for OnDemand in the Service Table, specify `0` (zero).

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=0
```

PROTOCOL

The networking protocol that OnDemand servers use to communicate with ODWEK. You must specify `0` (zero) for TCP/IP.

You must specify this parameter once in the default section.

This parameter is optional. If you do not specify this parameter, 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 required.

This section can 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@_gunnar]
HOST=gunnar
```

PORT

The TCP/IP port number that the OnDemand server uses to communicate with ODWEK. If you do not specify the PORT parameter, then the server uses the port number that is specified (or defaulted to) in the default server section.

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.

Example:

```
[@SRV@_gunnar]
PORT=0
```

This port number should match the port number specified in the ars.ini file for the instance.

PROTOCOL

The networking protocol that the OnDemand server uses to communicate with ODWEK. You must specify `0` (zero) for TCP/IP.

You can specify this parameter once in the server section.

This parameter is optional. If not specified, the value specified (or defaulted to) in the default server section is used.

Example:

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

[CONFIGURATION]

The CONFIGURATION section contains parameters that are used by ODWEK on the HTTP 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:

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=/QIBM/UserData/OnDemand/www/cache
```

Notes:

1. The APPLETCACHEDIR parameter has a global scope.
2. The APPLETCACHEDIR 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 AliasMatch:
   - If you specify a directory name, the directory must be relative to the `/QIBM/UserData/OnDemand/WWW` directory. For example, if you specify `appletdir=applets`, the applets must exist in the `/QIBM/UserData/OnDemand/WWW/APPLETS` directory.
   - If you specify an AliasMatch, it must be defined in the HTTP server configuration file. For example, if you specify `appletdir=/applets/`, the HTTP server configuration file must have an AliasMatch for `/applets/`. The

replacement file path of the AliasMatch rule must be set to the full path name of the directory on the server. For example:

```
AliasMatch ^/applets/com/ibm/edmslod/(.*)$ /QIBM/UserData/OnDemand/www/applets/$1

AliasMatch ^/applets/(.*)$ /QIBM/UserData/OnDemand/www/applets/$1
```

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

Use to specify the directory on the HTTP server in which ODWEK temporarily stores (*caches*) documents (see "CACHEDOCS"). By default, ODWEK caches documents in the `/QIBM/UserData/OnDemand/WWW/CACHE` directory.

Important: 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.

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

This parameter is optional.

Example:

```
[CONFIGURATION]
CACHEDIR=/QIBM/UserData/OnDemand/WWW/CACHE
```

CACHEDOCS

Determines whether ODWEK temporarily stores (*caches*) documents on the HTTP server. Cache storage can increase the speed with which previously viewed documents are retrieved from the server. The default value is `1` (one), which means that cache storage for documents is enabled. Specify a `0` (zero) to disable 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 19).

Important: 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. However, in general, most customers should always configure cache storage for documents.

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 the oldest items in 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 the oldest items in 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 HTTP server. Specify the value in megabytes. The default value is `1024`.

Important: To enable cache storage for documents, see "CACHEDOCS" on page 18.

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 HTTP server. Generally, this can increase the speed with which ODWEK sends previously viewed documents to users.

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. 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.
2. The userids listed in the CACHEUSERIDS list can access only those folders whose names and other information are 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.
3. 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=user1,user2,user3
```

CODEPAGE

Identifies the code page of the OnDemand database. By default, ODWEK uses the code page of the HTTP 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 HTTP server is running in a different code page than the database, then you must specify the CODEPAGE parameter.

Example:

```
[CONFIGURATION]
CODEPAGE=37
```

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 CONFIGURATION section.

This parameter is optional.

Example:

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

IMAGEDIR

Identifies the directory that contains image files used by ODWEK.

Notes:

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

   ```
   imagedir=pictures
   ```

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

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

2. You can specify a directory name or an AliasMatch:
   - If you specify a directory name, then the directory must be relative to the `/QIBM/UserData/OnDemand/WWW` directory. For example, if you specify `imagedir=pictures`, then the images must exist in the `/QIBM/UserData/OnDemand/WWW/PICTURES` directory.
   - If you specify an AliasMatch rule, then it must be defined in the HTTP server configuration file. For example, if you specify `imagedir=/pictures/`, then the HTTP server configuration file must have an AliasMatch for `/pictures/`. The AliasMatch rule must be set to the full path name of the directory on the server. For example:

     ```
     AliasMatch   ^/images/(.*)$   /QIBM/UserData/OnDemand/WWW/PICTURES/$1
     ```

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=pictures
```

LANGUAGE

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

Language	Value
Brazilian Portuguese	PTB
Canadian French	FRC
Croatian	HRV
Czech	CZE
Danish	DAN

Language	Value
English	ENU
Finnish	FIN
French	FRA
German	DEU
Greek	ELL
Holland Dutch	NLD
Hungarian	HUN
Italian	ITA
Japanese	JPN
Korean	KOR
Norwegian	NOR
Polish	PLK
Russian	RUS
Simplified Chinese	CHS
Slovak	SKY
Slovenian	SLO
Spanish	ESP
Swedish	SVE
Traditional Chinese	CHT

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

This parameter is optional.

Example:

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

ShowSearchString

Determines whether the Auto Find function is active or inactive. The Auto Find function supports transaction and text searching of line data documents from the Java line data viewer. The Auto Find function will automatically find and highlight the specific line in any document that matches the search criteria specified by the user.

When the Auto Find function is activated and a user performs either a transaction or text search and opens a document from the resulting document list, the system automatically searches the text of the document for the specified search criteria. If the search criteria is found, the line containing the search criteria is highlighted; otherwise the appropriate message is displayed. When the user opens another document for viewing (or reopens a previously viewed document), the search is performed again.

To activate the Auto Find function, set the ShowSearchString parameter to `1` (one). To deactivate the Auto Find function, set the ShowSearchString parameter to `0` (zero).

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

This parameter is optional. If not specified, the default value is `0` (zero; inactive).

Example:

```
[CONFIGURATION]
ShowSearchString=1
```

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 optional. If you do not specify the TEMPDIR parameter, ODWEK will store temporary files in the run-time directory. If you are using the CGI program, the run-time directory is the directory in which the CGI program was installed. If you are using the servlet, the run-time directory is the directory that contains the servlet: for some installations, the run time directory is the location of the java.exe file; for others, the run time directory is the servlets directory, however, the exact location is dependent on the Java application server.

Example:

```
[CONFIGURATION]
TEMPDIR=/QIBM/UserData/OnDemand/WWW/TMP
```

Important: Verify the permissions of the directory that you specify. The processes that run ODWEK programs must write to and read from the temporary directory.

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 `/QIBM/UserData/OnDemand/WWW/SAMPLES` directory.

Important: 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=/QIBM/UserData/OnDemand/WWW/SAMPLES
```

[SECURITY]

The SECURITY section contains the security parameters that are used by ODWEK on the HTTP 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 server. The Inactivity Time Out parameter determines when a 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 you do not specify the REPORTSERVERTIMEOUT parameter, then 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 servers that ODWEK can access. If you specify the SERVERACCESS parameter, then the clients that use ODWEK and the programs that use the APIs are permitted to access only those servers that you specify. You can specify the TCP/IP address, host name alias, 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.

Example:

```
[SECURITY]
SERVERACCESS=QUSROND
```

UPDATETIMESTAMP

Use to specify that ODWEK should update a timestamp after each transaction against a server. If an Inactivity Time Out value as not been set, that value will be compared to the time elapsed since the user's most recent transaction. The intent is to avoid unnecessary additional logons. To specify that ODWEK should update a timestamp after each transaction, set the UPDATETIMESTAMP parameter to 1 (one).

Important: If the same user ID is active in multiple browsers, the UPDATETIMESTAMP parameter may not function properly.

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

UPDATETIMESTAMP is optional. If not specified, or if set to 0 (zero), and if REPORTSERVERTIMEOUT is not set to 1 (one), ODWEK does not update a

timestamp after each transaction against a server. If an Inactivity Time Out value has been set, that value is compared to the time elapsed since the logon for the user, rather than since the user's most recent transaction. As a result, unnecessary additional logons may be performed.

For more information about the Inactivity Time Out parameter, see the online help for the administrative client. The UPDATETIMESTAMP and REPORTSERVERTIMEOUT parameters are similar. If set to 1 (one), both will update a timestamp after each transaction against a server. The difference occurs when the Inactivity Time Out period expires. REPORTSERVERTIMEOUT, causes the user's session to be terminated and an error reported. UPDATETIMESTAMP performs a new logon for the user, and does not report an error. If neither is set to 1 (one), a timestamp will not be updated, and the Inactivity Time Out value will be compared to the time elapsed since logon.

Example:

```
[SECURITY]
UPDATETIMESTAMP=1
```

[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 and install and configure it on the server. See your IBM representative for more information about the AFP2WEB Transform. Someone in your organization must also provide configuration options for the AFP2WEB Transform. See Appendix E, "AFP to HTML transform," on page 151 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 36 for details. (If you plan to use the Retrieve Document API, then you should specify the `_afp=HTML` parameter. See "Retrieve Document" on page 85 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 E, "AFP to HTML transform," on page 151 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 HTTP server.

Important: 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=/QIBM/UserData/OnDemand/www/HTMLbin
```

USEEXECUTABLE

Determines whether ODWEK starts the AFP2WEB Transform by using the shared library (DLL) or the executable (EXE).

Important: ODWEK on IBM i must use the executable. Therefore, this parameter must always be set 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 and install and configure it on the HTTP server. See your IBM representative for more information about the AFP2PDF Transform. Someone in your organization must also provide configuration options for the AFP2PDF Transform. See Appendix F, “AFP to PDF transform,” on page 155 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 36 for details. (If you plan to use the

Retrieve Document API, then you should specify the _afp=PDF parameter. See "Retrieve Document" on page 85 for details.)

3. By default, ODWEK uses the Adobe Acrobat viewer to view converted documents. You must obtain the viewer for the browsers 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 F, "AFP to PDF transform," on page 155 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 HTTP server.

Important: 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=/QIBM/UserData/OnDemand/www/PDFbin
```

USEEXECUTABLE

Determines whether ODWEK starts the AFP2WEB Transform by using the shared library (DLL) or the executable (EXE).

Important: ODWEK on IBM i must use the executable. Therefore, this parameter must always be set 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 43), the MIMETYPES section contains a parameter for each of the standard data types supported by OnDemand (AFP, BMP, EMAIL, GIF, JFIF, LINE, PCX, PDF, PNG, 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 client. 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/vnd.lotus-wordpro
   ```

   Then, when a user retrieves a document from the application, ODWEK sets the MIME content type to `application/vnd.lotus-wordpro` and the system starts Lotus WordPro to open the document.

Table 1 lists the MIME content types for several PC applications:

Table 1. MIME content types for several PC applications

Application	MIME content types
Lotus Applications	WK1=application/vnd.lotus-1-2-3 WK3=application/vnd.lotus-1-2-3 WK4=application/vnd.lotus-1-2-3 123=application/vnd.lotus-1-2-3 APR-application/vnd.lotus-approach VEW=application/vnd.lotus-approach LWP=application/vnd.lotus-wordpro SAM=application/vnd.lotus-wordpro MWP=application/vnd.lotus-wordpro SMM=application/vnd.lotus-wordpro PRE=application/vnd.lotus-freelance PRZ=application/vnd.lotus-freelance
Microsoft Applications	DOC=application/msword XLS=application/vnd.ms-excel PPS=application/vnd.ms-powerpoint PPT=application/vnd.ms-powerpoint MPD=application/vnd.ms-project MPP=application/vnd.ms-project MPT=application/vnd.ms-project MPD=application/vnd.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 36 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 client operating system.

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 37 for more information about processing EMAIL documents before sending them to the client.

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 client operating system.

Example:

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

LINE

The MIME content type for line data documents. See "LINEVIEWING" on page 38 for more information about processing line data documents before sending them to the client.

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 client operating system.

Example:

```
[MIMTYPES]
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 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
```

PNG

The MIME content type for PNG documents.

Notes:

1. By default, PNG documents are displayed by 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/png and starts the program that is associated with the PNG file type on the workstation operating system.

Example:

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

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 client operating system.

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 H, "No HTML output," on page 159 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 specific browser section as detailed in "[browser]" on page 41. (The parameters specified in a browser section override those from the DEFAULT BROWSER section.)

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:

ADDEXTENSION

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.

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

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

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
ADDEXTENSION=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 API 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, and you specify it 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.

Important: Users are permitted or denied the ability to add annotations to documents based on the Annotation permissions in the OnDemand application group.

This parameter has a global scope, and you specify it 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 client. 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 client.

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

ASCII ODWEK converts AFP documents to ASCII text.

HTML ODWEK converts AFP documents to HTML documents with the AFP2WEB Transform.

NATIVE ODWEK extracts and uncompresses AFP documents and their resources from OnDemand.

Important: 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 28 for details.

PDF ODWEK converts AFP documents to PDF documents with the AFP2WEB Transform.

Important: 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 28 for details.

PLUGIN ODWEK does not convert AFP documents (the default).

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=PLUGIN
```

AUTODOCRETRIEVAL

Specifies whether the client 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: Do not enable automatic document retrieval if you plan to use the Microsoft Internet Explorer browser. IBM suggests that you specify AUTODOCRETRIEVAL=0 in any browser sections that you define for Internet Explorer.

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

This parameter is optional.

Example:

```
[DEFAULT BROWSER]
AUTODOCRETRIEVAL=0
```

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 client.

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

NATIVE ODWEK extracts and uncompresses EMAIL documents from OnDemand.

Important: If you specify EMAIL=NATIVE, verify that the MIME content type identifies the viewer that you want to use. See "[MIMETYPES]" on page 28 for details.

HTML ODWEK converts EMAIL documents to HTML documents. This is the default value.

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 `_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, and you specify it 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, and you specify it 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 HTTP server, then you can encrypt these values by specifying ENCRYPTURL=1. See "Server and data security" on page 8 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 `1` (one), meaning that the folder description will be displayed. Specify `0` (zero) if you do not want to display the folder description. If this parameter is not specified or if the parameter has a value other than `1` (one), the folder description will not be displayed.

This parameter has a global scope, unless overridden in a browser section (see "[browser]" on page 41). 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 client.

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).

ASCII ODWEK converts line data documents to ASCII text.

NATIVE ODWEK extracts and uncompresses line data documents from OnDemand.

Important: If you specify LINEVIEWING=NATIVE, verify that the MIME content type identifies the viewer that you want to use. See "[MIMETYPES]" on page 28 for details.

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 `_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 the first value specified to determine the number of items to return to the document list:
   a. 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.
   b. For the Document Hit List API and Print Document API functions, the value of the `_max_hits` parameter, if specified for a function. The value of the `_max_hits` parameter overrides the MAXHITS parameter.
   c. The value of the MAXHITS parameter, if specified.
   d. If none of the above are specified, ODWEK returns a maximum of 50 items to the document list.

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

This parameter is optional.

Example:

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

ODApplet.jre.path.IE

See "Java line data viewer" on page 59.

ODApplet.jre.path.NN

See "Java line data viewer" on page 59.

ODApplet.jre.version

See "Java line data viewer" on page 59.

ODApplet.version

See "Java line data viewer" on page 59.

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 1 (one) and means that users can send documents to a server printer from the document list.

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.
3. When you select documents to print from the document list, only the first document you select will be printed (even if you have selected multiple documents).

This parameter has a global scope, and you specify it 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. The default value is P and means that users can select server printers. 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 H, "No HTML output," on page 159), determines whether the storage location of the document will appear in the output. See "Document Hit List" on page 161 API for details. The default value is `0` (zero) and means that the storage location will not appear in the output.

Important: To display the storage location, you must also set the `Display Document Location` property in the OnDemand folder.

This parameter has a global scope, and you specify it 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 1 (one) and means that annotations can be viewed.

Important: Users are permitted or denied the ability to view annotations to documents based on the Annotation permissions in the OnDemand application group.

This parameter has a global scope, and you specify it 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. 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 any 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.

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 35.

Examples:

```
[IE 5]
AUTODOCRETRIEVAL=0
NOLINKS=0

[Firefox/3.5.1]
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.

Important: The DEBUG section must be the first executable statement in the arswww.ini file.

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:

TRACE

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

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

This parameter is optional. To specify the trace level, use one of the following values:

0 No trace

1 Errors only

2 Errors and Warnings

3 Errors, Warnings, and Information

4 All

TRACEDIR

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

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

This parameter is optional. By default, if tracing is enabled, ODWEK writes the trace file to the /QIBM/UserData/OnDemand/WWW/LOG directory.

Example:

```
[DEBUG]
;Trace:None=0, Error=1, Error+Warn=2, Err+Warn+Info=3, All=4
Trace=4
TraceDir=/QIBM/UserData/OnDemand/www/logs
```

Example ARSWWW.INI file

A sample instance configuration for the default QUSROND instance is shown. The bolded items are configuration lines that need to be changed or added, and sometimes have notes beside them which are bolded and italicized and in parentheses. Comment lines begin with a semicolon. It is important that any directories specified in this file exist. If a directory does not exist, ODWEK will fail.

```
================================================================================
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Server Configuration      ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(Note: DEBUG should be turned off unless needed for problem determination.
This will GREATLY affect performance and should not be turned on unless needed.
To turn DEBUG on, the following three lines should be uncommented.  Remember to
recomment the lines once problem determination is completed.)
[DEBUG]
;Trace:None=0, Error=1, Error+Warn=2, Err+Warn+Info=3, All=4
;Trace=4
;TraceDir=/QIBM/UserData/OnDemand/www/logs

;[@SRV@_<host alias>]
;HOST=<host name>
;PORT=
;PROTOCOL=

[@SRV@_QUSROND]
HOST= (enter the fully-qualified system name or system TCP/IP address
       for the instance to be accessed)
PROTOCOL=0
PORT= (enter the 4-5 digit TCP/IP port address of the instance, for
       example port 1450)

[configuration]
CodePage=37     (change to the code page of the instance configured above)
Language=ENU    (change to the language code of the instance
                                                  configured above)
TemplateDir=/QIBM/UserData/OnDemand/www/SAMPLES
ImageDir=/images
```

```
AppletDir=/applet
TempDir=/tmp
(Note: Caching is recommended for performance; however, if the application group
        or data changes, the cache files should be deleted.)
CacheDir=/QIBM/UserData/OnDemand/www/CACHE
CacheSize=1024
CacheMinThreshold=40
CacheMaxThreshold=80
CacheDocs=1
CacheUserIDs=

[security]
ServerAccess=
ReportServerTimeout=0

;[afp2html]
;InstallDir=/QIBM/UserData/OnDemand/www/HTMLbin
;ConfigFile=/QIBM/UserData/OnDemand/www/HTMLbin/afp2html.ini

;[afp2pdf]
;InstallDir=/QIBM/UserData/OnDemand/www/PDFbin
;ConfigFile=/QIBM/UserData/OnDemand/www/PDFbin/afp2pdf.ini
;UseExecutable=0

[mimetypes]
BMP=image/IBM_OnDemand
GIF=image/IBM_OnDemand
JFIF=image/IBM_OnDemand
PCX=image/IBM_OnDemand
TIFF=image/IBM_OnDemand
PNG=image/IBM_OnDemand
PDF=application/pdf
AFP=application/afp
LINE=application/line
LINE=application/line
EMAIL=text/html
META=application/unknown
DOC=application/msword
123=application/vnd.lotus-1-2-3
LWP=application/vnd.lotus-wordpro
SAM=application/vnd.lotus-wordpro
PRZ=application/vnd.lotus-freelance
XLS=application/vnd.ms-excel
PPS=application/vnd.ms-powerpoint
PPT=application/vnd.ms-powerpoint
HTML=application/html
HTM=application/htm
(Note: Additional mimetypes may need to be added to handle other
document types. The browser must know what application to call to display
the document.)

[attachment images]
TXT=userText.gif
BMP=userBitMap.gif
GIF=userGIF.gif

[no html]
Begin=[
End=]
Separator=^

;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Default Browser    ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;

[default browser]
FOLDERDESC=1       (Set to '1' if you want the folder description
```

```
                  to be displayed)
;AfpViewing=[ascii,html,native,pdf,plugin,xenos]
AfpViewing=plugin  (This requires the afpplgus.exe or afpplgin.exe be
                  installed on your workstation)
;LineViewing=[ascii,applet,native]
LineViewing=applet  (The Line Data Applet version to be used is shown below on
                   the ODApplet.version configuration item)
;EmailViewing=[html,native]
EmailViewing=html
NoLinks=1
ViewNotes=1
AddNotes=1
ServerPrint=1
ServerPrinters=P
AutoDocRetrieval=1
MaxHits=200
ShowDocLocation=1
EncryptCookies=0
EncryptURL=0
ODApplet.version=2   (Use version two of the Line Data Viewer Applet)
ODApplet.jre.path.IE=http://www.java.com/en/download/windows_automatic.jsp
     (Where to find the Java Runtime Environment (JRE), if not already installed)
ODApplet.jre.version-1.4   (Minimum JRE version required)

(The sections below allow you to override the default configuration options for the
 browser types shown below.

[IE]
NoLinks=0
AddExtension=1
AddFieldsToDocid=1    (may need to be set to 0 to retrieve linedata (*SCS) docs)
AutoDocRetrieval=0
ViewNotes=1
AddNotes=1

[Mozilla/4.0 (compatible; MSIE 4.0; Windows 95)]
================================================================
```

Your next step

After you install the ODWEK software and configured the ARSWWW.INI file, you should configure the sample applications. See Chapter 4, “Configuring the sample applications,” on page 47.

Chapter 4. Configuring the sample applications

Setting up your OnDemand Web Enablement Kit environment typically requires that you perform these tasks:

1. Contact IBM Software Support for the latest PTFs for OnDemand. The list of current PTFs can be found in Informational APAR number II14497.
2. Contact IBM Software Support for the latest IBM i HTTP Server Group PTFs. The product number for the HTTP Server is 5770-DG1. Go to http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view and click on the appropriate group PTF number to view the latest list of HTTP Sever PTFs.
3. Contact IBM Support for the latest Database Group PTFs. Go to http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view and click on the appropriate group PTF number to view the latest list of DB2 PTFs.
4. Obtain a copy of the latest OnDemand *Read This First* document from the Web at http://www.ibm.com/software/data/ondemand/400/support.html. You can find it on the "Documentation" page along with the other Version 7 Release 1 documentation. Print and read the entire file before you begin.
5. Check the ODWEK prerequisites. See Chapter 3, "Installing and configuring the HTTP server," on page 13.
6. Install the OnDemand software on the IBM i server. See "Installing on IBM i" on page 14
7. Configure the ARSWWW.INI file for CGI and Java servlet. See "Specifying the ARSWWW.INI file" on page 15
8. Configure the Apache HTTP server. See Appendix G, Appendix G, "HTTP server configuration files," on page 157 for an example of an HTTP server configuration file.
9. Update the QONDADM and QRDARS400 authorization lists, if needed. See Chapter 3, "Other requirements" on page 13.
10. Set up your workstation browser. To do this perform the following tasks:
    a. Download and install the appropriate viewer plug-in files. See Chapter 5, "Installing the Web viewers," on page 51.
    b. To use the newest version of the Line Data Viewer Applet, you must download and install the latest Java Runtime Environment plug-in from http://www.java.com.
    c. Make sure your browser accepts all cookies. Select **Tools > Internet options**, then the Privacy tab.
    d. Make sure UTF-8 is selected for Internet Explorer. Select **Tools > Internet options**, then the Advanced tab, and then make sure **Always send URLs as UTF-8** is selected.
    e. Make sure the Java Runtime Environment is activated. Select **Tools > Internet options**, then Advanded tab, and then look under the Java (Sun) section. Ensure that **Use Java n vx.y.x for <applet> (requires restart)** is selected. The version shown in the panel will reflect the version that you currently have installed for your browser.

Important: If you are using the Java API to control ODWEK, see Appendix D, "Java API programming guide," on page 99 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 who are permitted to access several folders. Each user is defined to the OnDemand library server. After the user logs on to the server, ODWEK displays the list of folders they are permitted to open. ″LOGON.HTM″ contains instructions for customizing this application.
- CREDIT.HTM. This application supports casual use of OnDemand. Users are presented with search criteria for a specific folder. The OnDemand server name, user ID and password, folder name, and folder fields are coded in the application. See “CREDIT.HTM” on page 48for information about customizing CREDIT.HTM.
- FCREDIT.HTM. This application is 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 you might need.

Important: IBM recommend that, in addition to modifying the sample applications, you also 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 49 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. (For example: /www/HttpInstanceName/htdocs.)
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 lines:

```
<h4>Please enter your logon information:</h4>
<FORM METHOD=POST ACTION="/od/odservlet">
```

CREDIT.HTM

Customize the CREDIT.HTM sample application by making a copy of the file for each folder you want users to be able 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 /QIBM/UserData/OnDemand/www/samples directory.)
2. Change or delete the background image specified in the <body> statement (line 11).
3. If you want, change the background color specified in the <body> statement (line 11).
4. Change or delete the product image specified in the <img> statement (line 13).
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). Replace it with "/scripts/arswww.cgi" for CGI.
8. Replace the value specified in the <input> statement (line 30). This is a comma-separated string containing 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 matching 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 QUSROND.
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 user ID. The user ID you specify must have permission to open the folder and access the 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 TEMPLATE.HTM with a copy containing user-defined content. However, the template file must contain the following HTML comment line `<!---AOI#Marker--->`. The location of the comment line determines where the 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 23 for more information.

Your next step

After you have configured the sample applications, go to Chapter 5, "Installing the Web viewers," on page 51.

Chapter 5. 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 the people in your organization need to use.

- To view line data documents, you should use the Line Data applet. The Line Data applet is stored on the HTTP server. After you enable the Line Data applet, 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 for CGI and Java servlet or the ARSWWW.PROPS file for Java API specifies the viewer that your users will be using.
- To view AFP documents, you can use the IBM OnDemand AFP Web Viewer, the AFP2HTML applet, or the Adobe Acrobat viewer.
  - To view AFP documents with the IBM OnDemand AFP Web Viewer, users must install it on their workstations.
  - To view AFP documents with the AFP2HTML applet, an administrator must install and configure the AFP2WEB Transform on the HTTP server and configure the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API. The AFP2HTML applet is stored on the HTTP server. After an administrator enables the AFP2HTML applet, it is automatically loaded into memory on the workstation when the user selects to view an AFP document.
  - To view AFP documents with the Adobe Acrobat viewer, an administrator must install and configure the AFP2PDF Transform on the HTTP server and configure the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API. After an administrator enables the transform, by default, the browser attempts to start the Adobe Acrobat viewer when the user selects to view an AFP document. The user must install the Adobe Acrobat viewer on the workstation.

  Verify that the AFPVIEWING parameter in the ARSWWW.INI file for CGI and Java servlet or the ARSWWW.PROPS file for Java API specifies the viewer that your users will be using.
- To view BMP, GIF, JPEG, PCX, PNG, and TIFF documents, users should install the IBM OnDemand Image Web Viewer on their workstations; otherwise, they should use some other viewer that handles these types of documents. (For example, most browsers have built-in viewers that can display GIF and JPEG files.) 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 for CGI and Java servlet or the ARSWWW.PROPS file for Java API specify the viewers that your users will be using.

Note: 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.

If you want to define one user ID in OnDemand for multiple users to log on to OnDemand, and you want each user to access only their own information, you must configure the system as follows:

1. Provide a logon validation process as part of a Web application.
2. The logon validation process must take place before a query is sent to OnDemand.
3. Use the results of a successful logon to provide the account number to OnDemand.
4. Use the ODWEK APIs to send a SQL query to the OnDemand server. The SQL query should contain a specific account number.

For example: The user opens a "welcome to your company" home page. To access account information, the user must enter a user ID and PIN. These values are verified by your company's Web application, not by OnDemand. After a successful logon, the Web application then presents the user with the account summary page. That page contains a link for viewing account statements. When the user clicks "view account statement" the Web application invokes the ODWEK APIs, including a SQL query that contains the account number that was derived from the logon process. The APIs log onto the OnDemand server, using the user ID and password that you created for ODWEK connections to the server, search for and retrieve the user's statements, and send the document back to the browser. The browser launches the viewer that is appropriate for the type of data contained in the statement.

Requirements

For viewer requirements, see http://www.ibm.com/support/docview.wss?rs=152&uid=swg27016180.

Installation

Important: 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 “Distributing user-defined files” on page 53 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 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 `/QIBM/ProdData/OnDemand/www/plugins` directory:

- afpplgin.exe - IBM OnDemand AFP Web Viewer - All languages including DBCS support
- afpplgin.zip - IBM OnDemand AFP Web Viewer - Zip format for all languages including 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 18 MB of space on the workstation. The Image Web Viewer requires approximately 2 MB of space on the workstation. Remind your users to restart their browser if it is active during the installation process.

AFP Web viewer

The following settings may be applied from logical views on the server to the AFP Web Viewer.

- Background Color. The following colors are supported. No other colors are supported.

  Green Bar (displayed with a white background)

  Green

  Red

  Yellow

  Black

  White

  Grey
- Image Color. The following colors are supported. No other colors are supported.

  Yellow

  Blue

  Red

  Magenta

  Green

  Cyan

  Default (should display as black)
- Zoom.

Important: Selected Area Color is not applied to the AFP Web Viewer. The selected area is always displayed with white text and a black background.

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.
- 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.

Important: 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 HTTP 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 a 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 55 and "Storing user-defined files" on page 55 for details.
  5. If distributing user-defined Adobe Type 1 font files, then create a font configuration file. See "Configuring font files" on page 55 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 56 for details.
  7. Replace the EXE file provided by IBM for a Standard Install with the self-extracting EXE file that you built.
  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 client installation directory. For example, you could name the main client installation directory \ONDEMAND\AFP32.

To configure the main client installation directory to hold user-defined files:

1. Create a CUSTOM directory under the main client installation directory. For example:

   ```
   \ondemand\afp32\custom
   ```

   Important: The CUSTOM directory can hold other[1] 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 TRUETYPE 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.

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” for more information.

1. Other than AFP font files, Adobe Type 1 font files, and Windows TrueType font files.

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 55 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.

   Important: 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.

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.

Important: 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.

Mapping AFP fonts

AFP fonts that a document was created with need to be mapped to fonts that can be displayed using the AFP plug-in. 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 code resides.

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.

Displaying AFP reports

The FTDPORT2.INI file, which resides in the AFP Web Viewer installation directory, contains modifiable parameters that can affect how AFP reports display. This section describes these parameters and their values.

- Rules and lines

  If rules or lines do not display correctly when you view an AFP report, the problem might be the result of display driver differences. Use another method for displaying rules. In the Misc section of the FTDPORT2.INI file, change:

  ```
  RuleFix=FALSE
  ```

  to:

  ```
  RuleFix=TRUE
  ```

- Text fidelity

  If fonts are not substituted correctly and the text alignment is not correct, especially if the Text Fidelity parameter is set to Character, it might be because your report was created with 300-pel metrics instead of 240-pel metrics. If you specify 240Fidelity=FALSE, the report is displayed using 300-pel metrics. If you specify 240Fidelity=TRUE, the report is displayed using 240-pel metrics. The default is 240-pel metrics.

- Print dialog box default

  When the Print dialog box displays, the default is to print the current page of the report. You can change the default to print all of the pages of the report by specifying PrintAllPages=TRUE in the settings section of the FTDPORT2.INI file.

- User-defined page sizes

  You can define two page sizes for viewing reports that contain non-standard page sizes. These two user-defined page sizes are added to the list of other page sizes that can be selected when viewing a report. To define the two page sizes, modify the following two lines in the FTDPORT2.INI file:

  ```
  PaperSize1=width, length
  PaperSize2=width, length
  ```

  Specify the width and length, respectively, of each page in the report. All values must be in units of 1440ths of an inch.

  – If the page size is in inches, multiply it by 1440.
  – If the page size is in millimeters, multiply it by 56.7, and round the result to the nearest whole number.

  If no values are set for PaperSize1 and PaperSize2, the default page size for reports that contain non-standard page sizes is 8.5x11 inches.

- True Type fonts

  If you want to view reports using True Type fonts:

  1. If Adobe Type Manager (ATM) is installed on your workstation, disable or remove it. If Type 1 fonts are installed, they must be removed.
  2. Add the following line to the Misc section of the FTDPORT2.INI file: `TTONLY=TRUE`.

Displaying overlays

If the standard OnDemand end-user client view of an AFP data stream displays an overlay, and the ODWEK AFP Web Viewer does not display the overlay, the overlay resource probably was not found by the AFP Web Viewer.

To configure the AFP Web Viewer to display the overlay, specify the resource directory in the FTDPORT2.INI file. Using an ASCII text editor, such as Windows Notepad, open the file and find the entry named ResourceDataPath under [Preferences]. For example:

```
[Preferences]
DefaultView=DEFAULT
ViewDataPath=C:\Program Files\IBM\OnDemand AFP Web Viewer\Data
ResourceDataPath=C:\Program Files\IBM\OnDemand AFP Web Viewer\Resource
FontDataPath=C:\Program Files\IBM\OnDemand AFP Web Viewer\Font
```

The ResourceDataPath entry that is used for the OnDemand Client should match the same entry used for the AFP Web Viewer. Both the OnDemand Client and the AFP Web Viewer should have an FTDPORT2.INI file.

Important: External overlay resources are not downloaded with the AFP document. If the resource is external (if it is not stored in the same file as the AFP document), the resource must be downloaded with the AFP document. If the resource is external, it must be stored in the directory specified by the ResourceDataPath parameter.

The AFP Web Viewer does not download overlays to the resource directory specified by the ResourceDataPath; therefore, if the resource cannot be downloaded to the client workstation by some other means, the AFP data stream must be modified to include the resource so that the AFP document and the AFP resource are in the same file.

Image Web viewer

The following information applies when using the Image Web Viewer to view multi-page images.

Important: The following procedure requires that you edit the registry on the computer. You should not edit the registry unless it is absolutely necessary. If there is an error in the registry, the computer may not function properly. Before you proceed, you should make a backup copy of the registry and you should be familiar with how to restore the registry to the same version you were using when you last successfully started the computer. For instructions, see your Windows information.

For multi-page images, when the vertical scroll bar tab is dragged a small window will appear next to the tab. This window will display the page number corresponding to the tab position and the number of pages in the image. For example, a display of `5 / 10` indicates that there are ten pages in the image and that, if the tab is released, page number five will become the current page.

This behavior can be suppressed by a registry setting within this key:

HKEY_LOCAL_MACHINE\Software\IBM\OnDemand Image Web Viewer\Preferences

If the string value `PageNumberScroll` is set to `0` (zero), the page number window will not be displayed when the scroll bar tab is dragged.

Within the same registry key, if the string value `PageNumberToolbar` is set to `1` (one), page number information will be displayed on the toolbar for multi-page images. For example, a display of `3 / 5` indicates that there are five pages in the image and that page number three is the current page.

Java line data viewer

ODLineDataViewer2.jar is the Java line data viewer.

For Java line data viewer requirements, see http://www.ibm.com/support/docview.wss?rs=152&uid=swg27016180.

For CGI and Java servlet, an additional parameter in the ARSWWW.INI file determines the location of the Java plug-in installation file for Internet Explorer users who do not have the required version of the Java plug-in installed on their workstations.

Table 2 on page 60 describes the parameters in the ARSWWW.INI file that support the Java line data viewer.

Table 2. Parameters in ARSWWW.INI File for Java Line Data Viewer

Parameter	Value	Comments
ODApplet.jre.path.IE	http://java.sun.com/getjava/installer.html	For Internet Explorer. Specifies to automatically download and install the latest version of the Java plug-in from the java.sun.com Web site. See http://java.sun.com/getjava/install-windows.html for a preview of what happens when users automatically download and install the Java plug-in. The user might need to restart the browser after installing the plug-in.
	<location>	For Internet Explorer. Specifies the location of the Java plug-in installation file within a company's intranet. The location must include a valid browser protocol, such as http, file, or ftp. For example: file://shareName/java/plugins/plugin.exe An administrator must download the Java plug-in installation file and store it in the specified location. By specifying the location of the installation file, the browser will automatically install the Java plug-in on the workstation. The user might need to restart the browser after the installation completes.
ODApplet.jre.version	<version>	For Internet Explorer. Specifies the version of the Java plug-in to use. You must specify version 1.4 or later. Specify a major version number (for example, 1.4) to support any release of the plug-in 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 plug-in. Obtain valid version numbers, such as 1.4 or 1.4.1_01, from the java.sun.com Web site.

The following example shows how to configure the ARSWWW.INI file to support Version 1.4 or later of the Java plug-in. For Internet Explorer, users can automatically download and install the latest version of the Java plug-in from the java.sun.com Web site. Only users who do not have Version 1.4 or later of the Java plug-in installed on their workstations will be prompted to download and install the plug-in.

```
[DEFAULT BROWSER]
ODApplet.jre.path.IE=http://java.sun.com/getjava/installer.html
ODApplet.jre.version=1.4
```

Your next step

After you install 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 must verify the installation before you begin using ODWEK. For verification procedures, see Chapter 6, "Verifying the installation," on page 63.

Chapter 6. Verifying the installation

At this point, you should have completed all 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 the section "Verifying the CGI Program" on page 63. If you are using the Java servlet, go to the section "Verifying the servlet" on page 64.

Important: If you are using the Java API, refer to Appendix D, "Java API programming guide," on page 99 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 you have made to the configuration files.

1. Verify the HOST, PORT, and PROTOCOL parameters in the [@SVR@_default] section of the ARSWWW.INI file. The default location of the ARSWWW.INI file is /QIBM/UserData/OnDemand/www. If you are using multilingual support, review "Implementation" on page 173 to get additional information.

   Important: 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 on which the IBM HTTP server listens for requests from the client.

2. Launch a client (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 ODWEK should invoke. In this example, ODWEK invokes 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 4, "Configuring the sample applications," on page 47 for instructions on how to deploy sample applications.)
4. If the system is configured correctly, ODWEK displays the logon screen.
5. If the logon screen did not appear, see "Troubleshooting" on page 64.
6. From the Logon screen, type a user ID and password that is valid on the OnDemand library server. Click Submit to proceed to the Open a Folder screen.
7. At this point, the basic installation is successful. However, you may need to continue the validation process by retrieving different types of documents to text any transforms you may have integrated with ODWEK.

Verifying the servlet

Before you proceed, if you have not already done so, stop and restart the Web Application Server. For detailed information on configuring the servlet, go to http://www.ibm.com/software/data/ondemand/400/support.html, and search on "ODWEK" and "WebSphere."

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/odservlet`, where *server* is the hostname of the system on which you are deploying the servlet, *od* is the Content Root, and *odservlet* is the Servlet Mapping set in the WebSphere Application Server.

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

Troubleshooting

This section describes common errors that occur when you attempt to verify the installation, and contains possible solutions for those errors.

Problem	Solution
The Logon screen does not appear.	If the Logon screen does not appear, the most likely cause is that the mapping rules for the logon.htm file are incorrect. The mapping rules are specified in the httpd.conf file. See your HTTP server information for directions. If the mapping rules are correct: 1. Verify that the permissions to the samples directory, /QIBM/UserData/OnDemand/www/samples, or in whatever directory the logon script resides, are correct. 2. Verify that the permissions for the logon.htm file are correct. 3. Verify that the HTTP server is operational. Make any necessary corrections, and then restart the HTTP or Web Applications Server, and try logging on again.
Error 404. File not found.	If the Logon screen appeared, but you received an Error 404 message when you tried to log onto the server, verify the file mappings in the httpd.conf file. See your HTTP server documentation for instructions. Make any necessary corrections, and then restart the HTTP or Web Application Server, and attempt to log on again.

Problem	Solution
Error 500. Server Error	If the Logon screen appeared successfully, but you receive an Error 500 message after entering the logon information, check the HTTP or Web Application Server error logs for additional information. This is most likely a configuration error (for example, missing directories under /QIBM/UserData/OnDemand/www). Make any necessary corrections, then restart the HTTP or Web Application Server, and attempt to log on again.

Your 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 E, "AFP to HTML transform," on page 151
- Appendix F, "AFP to PDF transform," on page 155

These sections provide information for administrators:

- "Mapping AFP fonts" on page 57
- "AFP Web viewer" on page 53
- "Image Web viewer" on page 59
- "Java line data viewer" on page 59

These sections provide information for programmers:

- Appendix A, "CGI API reference," on page 67
- Appendix B, "Java servlet reference," on page 95
- Appendix C, "Java API reference," on page 97
- Appendix D, "Java API programming guide," on page 99
- Appendix H, "No HTML output," on page 159

For information on problem determination tools, hints, and tips, see Appendix J, "Problem determination tools," on page 167.

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.

Important: Parameter values are standard text. It is 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 the Add Annotation permission in the OnDemand application group. (The Access permission also lets users add annotations.)

Parameters

Table 3. Add Annotation function

Name=Value	Purpose
_function=addnote	Add an annotation.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid. The user must be given the Annotation Add permission for each application group that contains documents to be annotated. (The Application Group Access permission lets users add annotations.)
_password=*value*	The password for the user.
_folder=*value*	The name of the folder.
_perm=*value*	Determines whether the annotation is Public (`0`), Private (`1`), or Private for Group (`2`). Public annotations can be viewed by any user with View Annotation permission for 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 **_group** parameter contains the name of the group. The default value is `0` (Public).
_group=*groupName*	If the **_perm** parameter is set to `2` (Private for Group), names the group.
_copy=*value*	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.
_text=*value*	The text of the annotation.

Table 3. Add Annotation function (continued)

Name=Value	Purpose
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 add an annotation function.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_docid=*documentID*	The identifier of the document to which the annotation is to be attached. The document identifier is returned by the Document Hit List function.
_port=*value*	The port number for the OnDemand server. The default value, 0 (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number 1445. Any value that you specify overrides the value of the PORT parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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.
_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).

Usage

The following parameters are required:

_function
_server
_user
_password
_text
_docid

The following parameters are optional:

_perm
_group (required if _perm specifies Private for Group)
_html
_nohtml
_port
_codepage
_logoff

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=addnote
&_server=od400&_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 4. Change Password function

Name=Value	Purpose
_function=chgpassword	Change the OnDemand password for the userid.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid.
_password=*value*	The password for the userid.
_new_password=*value*	The new password for the userid.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 change password function.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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 4. Change Password function (continued)

Name=Value	Purpose
_cgibin=*program*	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 *program* can name a directory that is relative to the `ServerRoot` directive or name an *alias* that is defined in the HTTP 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).

Usage

The following parameters are required:

_function
_server
_user
_password
_new_password

The following parameters are optional:

_html
_nohtml
_port
_codepage
_logoff
_cgibin

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=chgpassword
&_server=od400&_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 a document, ODWEK retrieves the document from the server and displays it in the browser window using the appropriate viewer.

Parameters

Table 5. Document Hit List function

Name=Value	Purpose
_function=dochitlist	Display list of documents that match the search criteria.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid.
_password=*value*	The password for the userid.
_folder=*value*	The name of the folder.
*folder field name***=***value*	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.
*folder field name***2=***value*	For folder search fields that use the BETWEEN or NOT BETWEEN search operators, the upper value with which to search the field.
*folder field name***OP=***value*	The operator to be used to override the default operator for a folder search field. The *value* must be one of the following: **1** to indicate Equal **2** to indicate Not Equal **4** to indicate Less Than **8** to indicate Less Than or Equal **16** to indicate Greater Than **32** to indicate Greater Than or Equal **64** to indicate In **128** to indicate Not In **256** to indicate Like **512** to indicate Not Like **1024** to indicate Between **2048** to indicate Not Between
_display_fields=*value[,value,...]*	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.
_sort_field=*value[,value,...]*	Determines the folder search field 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: `_sort_field=Account,Account+Balance,Date` . The default sort fields are defined on the Field Information page for the folder.
_sort_order=*value[,value,...]*	For each folder search field that is specified in the sort_field 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: _sort_order=A,D,A` . The default sort order is determined by the sort order that is defined on the Field Information page for the folder.

Table 5. Document Hit List function (continued)

Name=Value	Purpose
_max_hits=*value*	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. ODWEK uses the first value specified to determine the number of items to return to the document list: 1. The value of the Maximum Hits field (specified on the folder Permissions page). This value overrides all other values. 2. The value of the **_max_hits** parameter, if specified. This value overrides the `MAXHITS` parameter from the ARSWWW.INI file. 3. The value of the `MAXHITS` parameter, if specified. 4. If none of the above are specified, ODWEK returns a maximum of 200 items to the document list.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 document hit list function.
_frame=*value*	The output of this command will include a `target=value` attribute. This parameter makes building HTML frames simpler. This is an optional parameter.
_datefmt=*value*	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 *IBM Content Manager OnDemand for i: Common Server Administration Guide* for details about date formats supported by OnDemand.
_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), ODWEK generates delimited ASCII output. See Appendix H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.

Table 5. Document Hit List function (continued)

Name=Value	Purpose
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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.
_sql=*string*	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. 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 command to list the internal date value for a given date. 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.
_date1=*value*	Use to specify the beginning date in a range of dates to search. If you specify the **_date1** and **_date2** 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.)
_date2=*value*	Use to specify the ending date in a range of dates to search. If you specify the **_date1** and **_date2** 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.)
_cgibin=*program*	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 *program* can name a directory that is relative to the `ServerRoot` directive or name an *alias* that is defined in the HTTP server configuration file. By default, ODWEK retrieves the CGI program from the /QIBM/Proddata/OnDemand/www/bin directory.
_or=*value*	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).
_logoff=1	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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder

The following parameters are optional:

folder field name
*folder field name*2
*folder field name*OP
_display_fields
_sort_field
_sort_order
_max_hits
_frame
_datefmt
_sql
_date1
_date2
_or
_html
_nohtml
_port
_codepage
_logoff
_cgibin

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=dochitlist
&_server=od400&_user=web&_password=web
&_folder=credit%20card%20statements
&account%20number=1000100010009999&date=1%2f1%2f96&date2=12%2f31%2f96
&nameOP=256&name=%AA
&_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 6. Logoff function

Name=Value	Purpose
_function=logoff	Log off an OnDemand server.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 logoff function.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.

Usage

The following parameters are required:

_function

The following parameters are optional:

_html
_nohtml
_port

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 7. Logon function

Name=Value	Purpose
_function=logon	Logon to an OnDemand server.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid.
_password=*value*	The password for the userid.
_new_password=*value*	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.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - –> 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 logon function.
_frame=*value*	The output of this command will include a `target=value` attribute. This parameter makes building HTML frames simpler. This is an optional parameter.
_datefmt=*value*	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 *IBM Content Manager OnDemand for i: Common Server Administration Guide* for details about date formats supported by OnDemand.

Table 7. Logon function (continued)

Name=Value	Purpose
_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 H, "No HTML output," on page 159 for details about the delimited ASCII output.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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=*program*	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 *program* can name a directory that is relative to the `ServerRoot` directive or name an *alias* that is defined in the HTTP server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.

Usage

The following parameters are required:

_function
_server
_user
_password

The following parameters are optional:

_new_password
_frame
_datefmt
_html
_nohtml
_port
_codepage
_logoff
_cgibin

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=logon
&_server=od400&_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 be given the Print Document permission in the OnDemand application group. (The Access permission also lets users print documents.) At least one server printer must be defined on the specified OnDemand server.

Parameters

Table 8. Print Document function

Name=Value	Purpose
_function=printDocuments	Print documents.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid. The user must be given the Document Print permission for each application group that contains documents to be printed. (The Application Group Access permission lets users print documents.)
_password=*value*	The password for the user.
_folder=*value*	The name of the folder.
_printer=*value*	The name of the OnDemand server printer. When the specified printer is a FAX or a Printer with Information, then you can specify the following additional parameters: **_recv_name=***value* The receiver's name. **_recv_comp=***value* The receiver's company name. **_recv_fax=***value* The receiver's fax number. **_send_name=***value* The sender's name. **_send_comp=***value* The sender's company name. **_send_tel=***value* The sender's telephone number. **_send_fax=***value* The sender's fax number. **_send_cover=***value* 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. **_subject=***value* A string that represents the subject of the document. **_notes=***value* A string that represents a note about the document.

Table 8. Print Document function (continued)

Name=Value	Purpose
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 print documents function.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_docids=*documentIDList*	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. **Important:** If the number of document identifiers exceeds 200, then you must specify the **_max_hits** parameter.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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 8. Print Document function (continued)

Name=Value	Purpose
_max_hits=*value*	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 **_docids** parameter. **Important:** 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 **_max_hits** parameter. If you do not specify the **_max_hits** 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 **_docids** parameter. ODWEK uses one of the following values to determine the number of document identifiers to process: • The value of the **_max_hits** parameter, if specified. This value overrides the value of the MAXHITS parameter. • The value of the MAXHITS parameter, if specified. • If none of the above are specified, ODWEK processes a maximum of 200 document identifiers.
_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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder
_printer
_docids

The following parameters are optional:

_recv_name
_recv_comp
_recv_fax
_send_name
_send_comp
_send_tel
_send_fax
_send_cover
_subject
_notes
_max_hits
_html
_nohtml
_port
_codepage
_logoff

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=printDocuments
&_server=od400&_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, viewer, or other program that is associated with the document type.

Parameters

Table 9. Retrieve Document function

Name=Value	Purpose
_function=retrieve	Retrieve the selected document.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid.
_password=*value*	The password for the userid.
_folder=*value*	The name of the folder.
folder field name=*value*	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.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - –> 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 retrieve function.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.

Table 9. Retrieve Document function (continued)

Name=Value	Purpose
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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=*program*	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 *program* can name a directory that is relative to the `ServerRoot` directive or name an *alias* that is defined in the HTTP server configuration file. By default, ODWEK retrieves the CGI program from the CGI-BIN directory.
_or=*value*	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).
_afp=*value*	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 client. 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 `_afp=HTML` so that ODWEK will convert the AFP document before sending it to the client. The *value* can be: **ASCII** ODWEK converts the AFP document to ASCII text. **HTML** ODWEK converts the AFP document to HTML with the AFP2WEB Transform. **NATIVE** ODWEK extracts and uncompresses the AFP document and its resources from OnDemand. **Important:** If you specify `_afp=NATIVE`, verify that the MIME content type identifies the viewer that you want to use (see "[MIMETYPES]" on page 28 for more information). **PDF** ODWEK converts the AFP document to PDF with the AFP2WEB Transform. **PLUGIN** ODWEK does not convert the AFP document (the default).
_email=*value*	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 client. The *value* can be: **NATIVE** ODWEK extracts and uncompresses the EMAIL document from OnDemand. **Important:** If you specify `_email=NATIVE`, verify that the MIME content type identifies the viewer that you want to use (see "[MIMETYPES]" on page 28 for more information). **HTML** ODWEK converts the EMAIL document to HTML.

Table 9. Retrieve Document function (continued)

Name=Value	Purpose
_line=*value*	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 client. The *value* can be: **APPLET** ODWEK converts the line data document for viewing with the Line Data applet (the default). **ASCII** ODWEK converts the line data document to ASCII text. **NATIVE** ODWEK extracts and uncompresses the line data document from OnDemand. **Important:** If you specify `_line=NATIVE`, verify that the MIME content type identifies the viewer that you want to use (see "[MIMETYPES]" on page 28 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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder

The following parameters are optional:

folder field name
_docid
_or
_afp
_email
_line
_html
_nohtml
_port
_codepage
_logoff
_cgibin

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=retrieve
&_server=od400&_user=web&_password=web
&_folder=credit%20card%20statements
&account%20number=10001000010009999&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 10. Search Criteria function

Name=Value	Purpose
_function=searchcrit	Display search criteria for a specific folder.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid.
_password=*value*	The password for the userid.
_folder=*value*	The name of the folder to search.
_html=*value*	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. The overall content of the HTML file is defined by the customer. However, the file must contain the following comment line: <!– - -AOI# Marker– - -> 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 search criteria function.
_frame=*value*	The output of this command will include a `target=value` attribute. This parameter makes building HTML frames simpler. This is an optional parameter.
_datefmt=*value*	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 *IBM Content Manager OnDemand for i: Common Server Administration Guide* for details about date formats supported by OnDemand.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.

Table 10. Search Criteria function (continued)

Name=Value	Purpose
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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=*program*	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 *program* can name a directory that is relative to the `ServerRoot` directive or name an *alias* that is defined in the HTTP 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 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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder

The following parameters are optional:

_frame
_datefmt
_html
_nohtml
_port
_codepage
_logoff
_cgibin

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=searchcrit
&_server=od400&_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.

Parameters

Table 11. Update Document function

Name=Value	Purpose
_function=updatedoc	Update the database.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid. The user must have Update document permission for the application group.
_password=*value*	The password for the user.
_folder=*value*	The name of the folder.
folder field name=*value*	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.
_html=*value*	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: <!– - -AOI# Marker– - -> 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.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_docid=*documentID*	The identifier of the document to be updated. The document identifier is returned by the Document Hit List function.

Table 11. Update Document function (continued)

Name=Value	Purpose
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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.
_logoff=1	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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder

The following parameters are optional:

folder field name
_docid
_html
_nohtml
_port
_codepage
_logoff

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=updatedoc
&_server=od400&_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 be given the View Annotation permission in the OnDemand application group. (The Access permission also lets users view annotations.)

Parameters

Table 12. View Annotations function

Name=Value	Purpose
_function=getnotes	View annotations.
_server=*value*	The name of the OnDemand server.
_user=*value*	The OnDemand userid. The user must be given the Annotation View permission for each application group that contains annotations to be viewed. (The Application Group Access permission lets users view annotations.)
_password=*value*	The password for the user.
_folder=*value*	The name of the folder.
_html=*value*	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: <!– - -AOI# Marker– - -> 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.
_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 H, “No HTML output,” on page 159 for details about the delimited ASCII output.
_docid=*documentID*	The identifier of the document that contains the annotations to be viewed. The document identifier is returned by the Document Hit List function.

Table 12. View Annotations function (continued)

Name=Value	Purpose
_port=*value*	The port number for the OnDemand server. The default value, `0` (zero), means that the server uses the port number that is specified in the Service Table (WRKSRVTBLE). If there is no port number specified in the Service Table, then OnDemand attempts to use port number `1445`. Any value that you specify overrides the value of the `PORT` parameter in the ARSWWW.INI file.
_codepage=*value*	The code page of the OnDemand database. The default code page is the code page of the HTTP 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.
_logoff=1	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).

Usage

The following parameters are required:

_function
_server
_user
_password
_folder
_docid

The following parameters are optional:

_html
_nohtml
_port
_codepage
_logoff

Sample Function Call

```
http://www.company.com/cgi-bin/arswww.cgi?_function=getnotes
&_server=od400&_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 67 for a reference of the functions, descriptions, and parameters.

For examples of Java servlet configurations, see http://www-306.ibm.com/software/data/ondemand/400/support.html, and search on "ODWEK" and "WebSphere." When you upgrade to a new release of OnDemand, you must update your EAR file as described in the examples document so that your CLASS and JAR files are updated.

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 `/QIBM/ProdData/OnDemand/www/api` directory. 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.
- Client/server implementation for Java application users

Source code for many of the examples shown in this section can be found in a zip file named ODApiSamples.zip in the /QIBM/ProdData/OnDemand/www/api directory on your IBM i server.

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.ODApplication
This class represents an OnDemand application. An instance of the ODApplication object provides an application developer access to information specified for an OnDemand application.

com.ibm.edms.od.ODApplicationGroup
This class represents an OnDemand application group. An instance of the ODApplicationGroup object provides an application developer access to information specified for an OnDemand application group. An instance of the ODApplicationGroup object is generated through an instance of the ODServer object by using the `ODServer.openFolder()` method.

com.ibm.edms.od.ODApplicationGroupFields
This class represents an OnDemand application group field. It contains application group field information.

Warning: It is required that access to all ODWEK objects only be done in a single-threaded environment.

com.ibm.edms.od.ODCabinet
This class represents a collection of OnDemand folders. Cabinets are defined and managed on the OnDemand server by the administrator.

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.ODConfig
The ODConfig Java object is the preferred method with which to configure system parameters. For more information, see "Configuring system parameters" on page 101.

com.ibm.edms.od.ODConstant
ODConstant is the public interface.

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.ODException
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.ODHitProperties
Use this class to obtain the OnDemand internal property values for a hit.

com.ibm.edms.od.ODHold
This class represents an OnDemand hold definition. This object is returned from a successful call to `(ODServer.getHolds())`.

Important: You must access all ODWEK objects in a single threaded environment.

com.ibm.edms.od.ODLogicalView
This class represents an OnDemand logical view.

com.ibm.edms.od.ODNamedQuery
This class represents an OnDemand named query. It contains the details of a named query, and enables the system to retrieve existing named queries and to save new named queries to the OnDemand server.

com.ibm.edms.od.ODNamedQueryCriteria
This class represents the criteria of an OnDemand named query. It contains search criteria details that are stored in a named query.

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.

com.ibm.edms.od.ODUser
This class represents an OnDemand user. From this class you can gather user information, such as address and phone number, that is stored in the OnDemand server.

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.props` file to a user-defined runtime directory.
2. Specify the name of the user-defined directory when you run the Java interpreter on the application. See "Running an ODWEK application" on page 104 for an example.

The Java API shared library (service program) is ARS3WAPI64 and is found in the QRDARS library.

Configuring system parameters

Using the ODConfig Java object is the preferred method for configuring system parameters.

There are eight system parameters required for a working ODServer instance. Use the ODConfig default constructor to set these parameters to their default values:

```
<pre>
     try{
          ODConfig cfg = new ODConfig();
          ODServer srvr = new ODServer(cfg);
          srvr.initialize("MyCustomApp");
          cfg.printConfig();
     }
     catch(ODException e) {
          System.out.println("Exception " + e);
     }
</pre>
```

This sample code configures the following default parameters:

```
<pre>
 AfpViewing     PLUGIN
 LineViewing    APPLET
 MaxHits        200
 AppletDir      /applets
 Language       ENU
 TempDir        The temp path as defined by the Java System.getProperty("java.io.tmpdir") method.
 TraceDir       The temp path as defined by the Java System.getProperty("java.io.tmpdir") method.
 TraceLevel     0
</pre>
```

For information on these parameters see "Specifying the ARSWWW.INI file" on page 15.

The TraceLevel parameter reflects the level of tracing that is used for ODWEK problem determination. For more information about Tracing, see "Tracing" on page 103.

You can also explicitly set these parameters by using the following sample code. This sample uses a different ODConfig constructor:

```
<pre>
     try{
          ODConfig cfg = new ODConfig(ODConstant.PLUGIN, //AfpViewer
                                      ODConstant.APPLET, //LineViewer
                                      null,              //MetaViewer
                                      500,               //MaxHits
                                      "c:\\applets",     //AppletDir
                                      "ENU",             //Language
                                      "c:\\temp",        //TempDir
                                      "c:\\temp\\trace", //TraceDir
                                      1);                //TraceLevel
            ODServer srvr = new ODServer(cfg);
            srvr.initialize("MyCustomApp");
            cfg.printConfig();
          }
          catch(ODException e){
               System.out.println("Exception " + e);
          }
</pre>
```

This constructor sets parameters with zero or null values to their defaults.

Important: This object has no methods to set parameters, except during construction. The object cannot be modified after it has been constructed.

Example ARSWWW.PROPS file

The following sample shows an ARSWWW.PROPS file, which is used instead of the ARSWWW.INI file for Java API:

```
#AfpViewing=[ascii,html,native,pdf,plugin]
AfpViewing=plugin

#LineViewing=[ascii,applet,native]
LineViewing=applet

MaxHits=50
Language=ENU
TempDir=/QIBM/UserData/OnDemand/www/tmp

#Trace:None=0, Error=1, Error+Warn=2, Err+Warn+Info=3,All=4
Trace=4
TraceDir=/QIBM/UserData/OnDemand/www/logs

AppletVersion=2
AppletJREVersion=
AppletJREDownloadPathIE=
AppletJREDownloadPathNN=
AppletJREDownloadPathNN=

AppletDir=/QIBM/UserData/OnDemand/www/applets
AppletCacheDir=
AppletImageDir=

AppletFitWidth=
AppletServerSearch=
CompressMemSize=
DocMemSize=
ShowAllFolderCriteria=
AddFieldsToDocid=1
ServerAccessList=

#Afp2HtmlInstallDir=/QIBM/UserData/OnDemand/www/binhtml
```

```
#Afp2HtmlConfigFile=/QIBM/UserData/OnDemand/www/binhtml/afp2web.ini
#Afp2PdfInstallDir=/QIBM/UserData/OnDemand/www/binpdf
#Afp2PdfConfigFile=/QIBM/UserData/OnDemand/www/binpdf/afp2pdf.ini
```

Tracing and diagnostic information

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

Tracing

ODWEK tracing writes trace statements to an ARSWWW.TRACE file located in the trace directory specified in your ARSWWW.PROPS file. ODWEK tracing is intended to assist in problem determination. As with any form of tracing, there is a decline in performance when ODWEK tracing is enabled.

You must manually clean the trace file periodically. The trace file is not circular, and requires sufficient file space.

To enable ODWEK tracing, perform the following steps:

1. Modify the ODConfig default constructor. Follow the instructions in "Configuring system parameters" on page 101.
2. Modify the following debug stanza to reflect the following:

   ```
   ;Trace:None=0, Error=1, Error+Warn=2, Err+Warn+Info=3, All=4
   Trace=4
   TraceDir=/QIBM/UserData/OnDemand/www/logs
   ```

   Tip: Tracing can be set to many different levels with the Trace parameter. When you troubleshoot an ODWEK issue, set the trace level to the highest level, unless otherwise specified by IBM Software Support. Then, for monitoring an ODWEK application that is in a steady state, you might want to set the trace at a lower level. For example, Trace=1 creates minimal overhead and alerts you to error conditions only. If you are using an ARSWWW.PROPS file from a previous release of ODWEK, delete the old debug section. Having multiple debug sections might prevent ODWEK tracing.
3. The arswww.trace file is created in the directory referenced by the TraceDir parameter.
4. Your ODWEK application must be restarted for the changes you made to the ARSWWW.PROPS file to take effect.
5. If you are enabling tracing to troubleshoot a problem, recreate the issue after tracing is enabled, and then send the ARSWWW.TRACE file to IBM support.

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

```
TraceDir=/QIBM/UserData/OnDemand/www/logs
```

Important: 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 J, "Problem determination tools," on page 167 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 `ODException` class or one of its subclasses.

When a `ODException` is created, the API logs diagnostic information into a log file, assuming that logging is enabled. See "Tracing" on page 103 for more information about the log file used by the Java APIs.

When a `ODException` 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 ( ODException e )
 {
   System.out.println( "ODException: " + 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 97 for more information.

Running an ODWEK application

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

1. You need to set up exports prior to compiling and running the application:

   ```
   export LIBPATH=/QSYS.LIB/QRDARS.LIB
   export PATH=/usr/bin:/your_program_path
   export -s CLASSPATH=/QIBM/ProdData/OnDemand/www/api/ODApi.jar:/your_program_path
   ```

2. 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[ ] )
  {
    .
    .
    .
  }
}
```

3. Compile your ODWEK application file (`.java`) with `javac` to produce the `.class` file. See your Java reference publication for instructions on compiling Java applications.

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 97 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( );
odServer.initialize( "AppName" );

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

char directory = "/QIBM/UserData/OnDemand/www";
int port = Integer.parseInt('1450');

odServer.logon( "LIBSRVR1", "ADMIN", "PASSWD", CONNECT_TYPE_TCPIP, port, directory );
  .
  .
  .
odServer.logoff( );
odServer.terminate( );
```

See "Working with an OnDemand server" on page 106 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.initialize( "AppName" );

odServer.setServerName("LIBSRVR1");
odServer.setUserId("ADMIN");
odServer.setPassword("PASSWD");

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

char directory = "/QIBM/UserData/OnDemand/www";
int port = Integer.parseInt('1450');

odServer.logon( odServer.getServerName( ),
```

```
                odServer.getUserId( ),
                odServer.getPassword( ),
                ODConstant.CONNECT_TYPE_TCPIP,
                port,
                directory );
.
.
.
odServer.logoff( );
odServer.terminate( );
```

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, 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
- getPassword
- getPort
- getServerName
- getUserId
- setConnectType
- setPassword
- setPort
- setServerName
- setUserId

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Configuration directory (location of `arswww.props` file)

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 < 5 )
```

```
{
    System.out.println( "usage: java TcServerMisc <server> <port> <userid> <password> <config dir>" );
    return;
}

try
{
    //----------
    // Set the stage
    //----------

    System.out.println( "Testcase TcServerMisc started." );
    System.out.println( "This testcase should:" );
    System.out.println( "  Use ODServer methods setServer, setUserId, 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
    //----------
    ODConfig odConfig = BuildODConfig.build(argv[4] + "/arswww.props");
    if(odConfig == null)
        System.out.println("BuildODConfig Failed.");

    else
    {
        odServer = new ODServer(odConfig );
        odServer.initialize(  "TcListFolders.java" );
        odServer.setServerName( argv[0] );
        odServer.setUserId( argv[2] );
        odServer.setPassword( argv[3] );
        odServer.setConnectType(ODConstant.CONNECT_TYPE_TCPIP);
        odServer.setPort(Integer.parseInt(argv[1]));
        System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
            odServer.logon( );

        //----------
        // Test miscelaneous methods
        //----------
        System.out.println( "Server Name: " + odServer.getServerName( ) );
        System.out.println( "User Id: " + odServer.getUserId( ) );
        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 );

        System.out.println( "Setting connect type to ODConstant.CONNECT_TYPE_TCPIP..." );
        odServer.setConnectType( ODConstant.CONNECT_TYPE_TCPIP );

        System.out.println( "Setting install directory to  /QIBM/ProdData/OnDemand/www ..." );
        odServer.setInstallDir( "/QIBM/ProdData/OnDemand/www" );

        //----------
        // Cleanup
        //----------

        System.out.println( "Logging off..." );

        odServer.logoff( );
        odServer.terminate( );
        System.out.println( "" );

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

catch ( ODException e )
{
    System.out.println( "ODException: " + 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 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;
    }
}
```

Connecting to a non-default port using the Java APIs

In some instances, you may have to access a non-default port using the ODWEK Java APIs. For example, you may have two instances on the OnDemand server; one instance uses the default port, and the other uses a different port. Unless you configure your system properly, when you run your Java program, you will receive the following error: "A connection cannot be established to the instance2 server."

To implement this arrangement, use the `ODServer.setPort()` method just before the logon within your Java source. Then specify the host name for your server name (instead of the instance alias). The methods do not use that part of the ARSWWW.PROPS file to resolve instances.

Building ODConfig

The following example refers to the Java program called `BuildODConfig`. The example provides a guideline for constructing a program to set the configuration parameters for the Java code.

Example of using ODConfig to set configuration parameters:

```
import java.io.FileInputStream;
import java.util.Properties;

import com.ibm.edms.od.ODConfig;


public class BuildODConfig{

        static public ODConfig build(String propsFile){
        ODConfig odConfig = null;
        try{
         FileInputStream fileinput = new FileInputStream(propsFile);
         Properties props = new Properties();
         props.load(fileinput);
         String AfpV = (String)props.get("AfpViewing");
         String LineV = (String) props.get("LineViewing");
         String MetaV = (String) props.get("MetaViewing");
         long maxHits = Long.parseLong((String)props.get("MaxHits"));
         String appletDir = (String)props.get("AppletDir");
         String lang = (String)props.get("Language");
         String tempDir = (String)props.get("TempDir");
         String traceDir = (String)props.get("TraceDir");
         int tracelvl = Integer.parseInt((String)props.get("Trace"));

         props.remove("AfpViewing");
         props.remove("LineViewing");
         props.remove("MetaViewing");
         props.remove("MaxHits");
         props.remove("AppletDir");
         props.remove("Language");
         props.remove("TempDir");
         props.remove("TraceDir");
```

```
            props.remove("Trace");

            odConfig = new ODConfig(AfpV,
                    LineV,
                    MetaV,
                    maxHits,
                    appletDir,
                    lang,
                    tempDir,
                    traceDir,
                    tracelvl,
                    props);
          }
          catch ( Exception e2 )
          {
            System.out.println( "exception: " + e2 );
            System.out.println( "Cause: " + e2.getCause());
            e2.printStackTrace( );
          }
          return odConfig;
      }
}
```

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:
- getNumApplGroups
- getApplGroupNames
- 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
- getApplicationGroup
- logoff
- terminate

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.props` file)

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;
    String[ ] appls;
    int j;
    long agid=0;
    char conntype = 'T';
    String directory = "";

    //----------
    // If too few parameters, display syntax and get out
    //----------
    if ( argv.length < 6 )
```

```
{
  System.out.println( "usage: java TcApplGrp <server> <port> <userid> <password> <folder> <config dir>" );
  return;
}

try
{
  //----------
  // Set the stage
  //----------
  System.out.println( "Testcase TcApplGrp started." );
  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( "  Get application group by name and display" );
  System.out.println( "  Get application group by application group id and display" );
  System.out.println( "" );
  System.out.println( "--------------------------------------------------" );
  System.out.println( "" );

  //----------
  // Logon to the specified server
  //----------
  ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");

  if (odConfig != null)
  {
     odServer = new ODServer(odConfig );
     odServer.initialize(  "TcApplGrp.java" );
     System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
     odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory );

     //----------
     // Open the specified folder
     //----------
     System.out.println( "Opening " + argv[4] + " folder..." );
     odFolder = odServer.openFolder( argv[4] );

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

     System.out.println("\nGet the Application Group by name: " + agname);
     ODApplicationGroup odAG1 = odServer.getApplicationGroup(agname);
     System.out.println("Application Group Name = " + odAG1.getName());
     System.out.println("Application Group Description = " + odAG1.getDescription());
     System.out.println("Applicaiton Group ID = " + odAG1.getId());
     System.out.println("Mapped Applications:");
     appls = odAG1.getApplicationNames();
     for (int i = 0; i < appls.length; i++)
         System.out.println("   " +appls[i]);
     agid =  odAG1.getId();
     System.out.println("Application Group ID = " + agid);

     //----------
     // Cleanup
     //----------
     odFolder.close( );
     odServer.logoff( );
     odServer.terminate( );
  }
  else
  {
     System.out.println( "" );
     System.out.println( "Testcase TcApplGrp failed  -" );
     System.out.println( "  ODConfig could not be initialized.  " );
     System.out.println( "  Probable cause: " );
     System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
     System.out.println( "" );
  }

  System.out.println("Re-Logon and attempt to get the Application Group using Application Group ID saved above.");
  odConfig = BuildODConfig.build(argv[5] + "/arswww.props");

  if (odConfig != null)
  {
     odServer = new ODServer(odConfig );
     odServer.initialize(  "TcApplGrp.java" );
     System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
     odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory );
     System.out.println( "Opening " + argv[4] + " folder..." );
     odFolder = odServer.openFolder( argv[4] );

     System.out.println("\nGet the Application Group for id " + agid);
     ODApplicationGroup odAG = odServer.getApplicationGroup(agid);
     System.out.println("Application Group Name = " + odAG.getName());
     System.out.println("Application Group Description = " + odAG.getDescription());
     System.out.println("Application Group ID = " + odAG.getId());
     System.out.println("Mapped Applications:");
     appls = odAG.getApplicationNames();
     for (int i = 0; i < appls.length; i++)
```

```
            System.out.println("   " +appls[i]);

          //----------
          // Cleanup
          //----------
          odFolder.close( );
          odServer.logoff( );
          odServer.terminate( );
          System.out.println( "" );
          System.out.println( "-------------------------------------------------" );
          System.out.println( "" );
          System.out.println( "Testcase TcApplGrp completed. - analyze results if required" );
        System.out.println( "" );
       }
       else
       {
          System.out.println( "" );
          System.out.println( "Testcase TcApplGrp failed  -" );
          System.out.println( "  ODConfig could not be initialized.  " );
          System.out.println( "  Probable cause: " );
          System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
          System.out.println( "" );
       }

     }

     catch ( ODException e )
     {
         System.out.println( "ODException: " + 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( );
     }
   }
}
```

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
- setMaxHits
- search
- getSearchMessage
- close

This example demonstrates these `ODCriteria` methods:
- getName
- setOperator
- 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
- logoff
- terminate

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Maximum hits
- 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.props` file)

Restriction: The number of hits can be restricted by the MAXHITS parameter in the `arswww.props` 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 server, userid, password, folder, directory;
        String crit = "", operator = "", value1 = "", value2 = "";
        String header, line1, line2, hit_value, useable_value;
        boolean mismatch_detected, use_default_values;
        int j, k, opr;
        int port, maxHits;

        //----------
        // If too few parameters, display syntax and get out
        //----------
        if ( argv.length <  11 && argv.length != 7 )
        {
            System.out.println( "usage: java TcSearch <server> <port> <userid> <password> <folder> <MaxHits (-1 for default)> " );
            System.out.println( "                                                    <criteria> <opr> <value1> <value2> <config dir>" );
            System.out.println( "      or, to use deafult search criteria" );
            System.out.println( "java TcSearch <server> <port> <userid> <password> <folder> <MaxHits (-1 for default)> <config dir>" );
            return;
        }

        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TcSearch started." );
            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( "  If specified:" );
            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.props 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
//----------
use_default_values = argv.length == 7;
server   = argv[0];
port     = Integer.parseInt(argv[1]);
userid   = argv[2];
password = argv[3];
folder   = argv[4];
maxHits  = Integer.parseInt(argv[5]);
if ( use_default_values )
    directory = argv[6];
else
{
    crit      = argv[6];
    operator  = argv[7];
    value1    = argv[8];
    value2    = argv[9];
    directory = argv[10];
}

ODConfig odConfig = BuildODConfig.build(directory + "/arswww.props");
if(odConfig == null)
{
   System.out.println( "" );
   System.out.println( "Testcase TcListFoldersByCrit failed  -" );
   System.out.println( "  ODConfig could not be initialized.  " );
   System.out.println( "  Probable cause: " );
   System.out.println( "    File arswww.props is not found in directory " +  directory );
   System.out.println( "" );
}
else
{
    odServer = new ODServer(odConfig );
    odServer.initialize(  "TcSearch.java" );
    String localdir = "";
    System.out.println( "Logging on to " + server  + " server with user " + userid + "..." );
    odServer.logon( server, userid, password, ODConstant.CONNECT_TYPE_TCPIP, port, localdir);

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

    System.out.println( "Name='" + odFolder.getName( ) + "' Desc='" + odFolder.getDescription( ) + "'" );

    //----------
    // If we are not using the default search values:
    //----------
    if ( !use_default_values )
    {
        //----------
        // Find the requested criteria
        //----------
        System.out.println( "Getting " + crit + " criteria..." );
        odCrit = odFolder.getCriteria( crit );
        if ( odCrit == null )
            System.out.println( "  *** " + crit + " criteria does not exist - NullPointerException will be reported" );

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

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

```
                    //----------
                    // Set the search values
                    //----------
                    if ( opr == ODConstant.OPBetween || opr == ODConstant.OPNotBetween )
                    {
                        odCrit.setSearchValues( value1, value2 );
                        System.out.println( "  " + odCrit.getName( ) + " " + getOperatorName( opr ) + " " + value1 + " and " + value2 );
                    }
                    else
                    {
                        odCrit.setSearchValue( value1 );
                        System.out.println( "  " + odCrit.getName( ) + " " + getOperatorName( opr ) + " " + value1 );
                    }
                }
                //----------
                // Set Max Hits limit if specified
                //----------
                if (maxHits != -1)
                {
                    System.out.println("Setting MaxHITS to " + maxHits + ".");
                    odFolder.setMaxHits(maxHits);
                }
                else
                    System.out.println("No MaxHits passed in. Will use the MaxHits in arswww.props/ODConfig, or ");
                    System.out.println("   the Folder defined max, which ever is less.");

                //----------
                // Search the folder
                //----------
                System.out.println( "  Searching " + folder + ( use_default_values ? " using default values" : "" ) + "..." );

                long startTime  = System.currentTimeMillis();
                hits = odFolder.search( );
                long estimatedTime = System.currentTimeMillis() - startTime;
                System.out.println( "Elapsed Search Time: " +  estimatedTime + " ms");
                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( "        " + 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( "" );
                System.out.println( "Testcase TcSearch 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 ( ODException e )
        {
            System.out.println( "ODException: " + 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 = "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.OPGreaterThanEqual:
            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 = "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.FileTypeSCS:
            str = "SCS";
            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;
    }
}
```

Searching a folder using an SQL string

The following example uses `ODFolder` methods to open the specified folder, search the folder with the specified SQL string, and close the folder. This example uses `ODHit` methods to display the number of items that match the query and to display the document list.

This example demonstrates these `ODFolder` methods:

- getName
- getDescription
- getDisplayOrder
- setApplGroupForSearchWithSQL
- setMaxHits
- search
- close

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
- logoff
- terminate

This example uses these run-time parameters:

- Server name
- Port
- User Id
- Password
- Folder name
- SQL string
- Max hits

- Configuration directory (location of `arswww.props` file)
- Date 1 (optional)
- Date 2 (optional)
- Date format (optional)
- Application group (optional)

Example of searching a folder using an SQL string:

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

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

        //----------
        // If too few parameters, display syntax and get out
        //----------
        if ( argv.length < 8)
        {
            System.out.println( "usage: java TcSQLSearch <server> <port> <userid> <password> <folder> <SQL> );
            System.out.println( "                        <MaxHits (-1 to use default)> <config dir>" );
            System.out.println( "                        <opt begin Date> <opt end Date> <opt Date Format> <opt ApplGrpName>" );
            return;
        }
        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TCSQLSearch started." );
            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( "  Set the application group (if specified)" );
            System.out.println( "  Search the folder using specified Sql" );
            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.props 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
            //----------
            ODConfig odConfig = BuildODConfig.build(argv[7] + "/arswww.props");
            if(odConfig == null)
                System.out.println("BuildODConfig Failed.");
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize(  "TcSQLSearch.java" );
                System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
                String directory = "";
                odServer.logon( argv[0], argv[2], argv[3], ODConstant.CONNECT_TYPE_TCPIP, Integer.parseInt(argv[1]), directory);

                //----------
                // Open the specified folder and find the requested criteria
                //----------
                System.out.println( "Opening " + argv[4] + " folder..." );
                odFolder = odServer.openFolder( argv[4] );
                System.out.println( "Name='" + odFolder.getName( ) + "' Desc='" + odFolder.getDescription( ) + "'" );
                System.out.println( "Getting " + argv[5] + " criteria..." );

                //----------
                // Seearch with Application Group limiters if specified
                //----------
                if(argv.length == 11 )
                {
                    System.out.println("**Search is limited to Application Group " + argv[9]);
                    odFolder.setApplGroupForSearchWithSQL(argv[10]);
                }
                //Set MaxHits
                int maxHits  = Integer.parseInt(argv[6]);
                if (maxHits != -1)
                {
                    System.out.println("Setting MaxHITS to " + maxHits + ".");
                    odFolder.setMaxHits(maxHits);
                }
                else
                    System.out.println("No MaxHits passed in. Will use the MaxHits in arswww.props/ODConfig, or ");
                    System.out.println("   the Folder defined max, which ever is less.");
```

```
//----------
// Seearch with date limiters if specified
//----------
if(argv.length >= 10 ) //call search(sql,date1,date2,date format)
{
    String date1 = argv[8];
    if(date1.compareTo(" ") != 0) //Only go in here if date was actually spec'd
    {
        System.out.println("Searching " + argv[4] +" with Date Range limits of " + argv[8] + " and " + argv[9]);
        System.out.println("Date format is " + argv[10]);

        hits = odFolder.search(argv[5],
                argv[8],  //date1
                (argv.length > 10) ?  argv[9] : " ",   //date2
                        (argv.length >= 11) ? argv[10] : " "  ); //date format
    }
    else
    {
        //----------
        // Search the folder
        //----------
        System.out.println( "  Searching " + argv[4] + "with SQL " + argv[5] + "..." );
        hits = odFolder.search(argv[5] );
    }
}
else if(argv.length > 8 && argv.length < 11 ) //call search(sql,date1,date2)
{
    String date1 = argv[8];
    if(date1.compareTo(" ") != 0) //Only go in here if date was actually spec'd
    {
        System.out.println("Searching " + argv[4] +" with Date Range limits of " + argv[8] + " and " +argv[9]);

        hits = odFolder.search(argv[5],
                argv[8],  //date1
                (argv.length > 10) ?  "" : argv[9]);   //date2
    }
    else //call search(sql)
    {
        //----------
        // Search the folder
        //----------
        System.out.println( "  Searching " + argv[4] + "with SQL " + argv[5] + "..." );
        hits = odFolder.search(argv[5] );
    }
}
else //call search(sql)
{
    //----------
    // Search the folder
    //----------
    System.out.println( "  Searching " + argv[4] + "with SQL " + argv[5] + "..." );
    hits = odFolder.search(argv[5] );
}

//----------
// Display the hits
//----------
System.out.println(" Number of Hits Found = " + hits.size());
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( "        " + 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( "" );
```

```
            System.out.println( "Testcase TcSQLSearch 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 ( ODException e )
    {
        System.out.println( "ODException: " + 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 = "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.OPGreaterThanEqual:
        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 = "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.FileTypeSCS:
                str = "SCS";
            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:
- getName
- getCriteria
- search
- close

This example demonstrates these `ODCriteria` methods:
- setOperator
- setSearchValues

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name

- Configuration directory (location of `arswww.props` file)

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 ( ODException e )
        {
            System.out.println( "ODException: " + 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( );
        }
    }
}
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 < 6 )
        {
            System.out.println( "usage: java TcCancelSearch <server> <port> <userid> <password> <folder> <config dir>" );
            return;
        }
        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TcCancelSearch started." );
            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 1970-2005" );
            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 .1 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 arswww.props 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
            //----------
            ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");
            if(odConfig == null)
                System.out.println("BuildODConfig Failed.");
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize( "TcCancelSearch.java" );
                char conntype = 'T';
                String directory = "";
                System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
                odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);

                //----------
                // Open the specified folder and display its name and description
                //----------
                System.out.println( "Opening " + argv[4] + "..." );
                odFolder = odServer.openFolder( argv[4] );
```

```
                odCrit = odFolder.getCriteria( "Report Date" );
                odCrit.setOperator( ODConstant.OPBetween );
                odCrit.setSearchValues( "01/01/1970", "01/01/2005" );

                //----------
                // 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)..." );
                System.out.println( "  Searching " + odFolder.getName( ) + "..." );
                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 .1 seconds..." );
                ( Thread.currentThread( ) ).sleep( 100 );
                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.logoff( );
                odServer.terminate( );
                System.out.println( "" );
                System.out.println( "--------------------------------------------------" );
                System.out.println( "" );
                System.out.println( "Testcase TcCancelSearch completed. - Ensure that the second search," );
                System.out.println( "  which was cancelled, yielded fewer hits than the first" );
                System.out.println( "" );
            }
        }
        catch ( ODException e )
        {
            System.out.println( "ODException: " + 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 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` method. Fixed search values are listed for any search fields that are defined as `FixedChoice` or `Segment` .

This example demonstrates these `ODCriteria` methods:

- getName
- getOperator
- getValidOperators
- getType
- getDefaultFmt
- getDisplayFmt
- getDisplayFmtQual
- getMaxEntryChars
- getMaxDisplayChars
- getMinSearchValue
- getMaxSearchValue
- getDBFieldNames
- getDBFieldMask
- getSearchValues
- getFixedValues
- isUpdateable
- isRequired

- isDefaultValueAvailable
- isDefaultValueFixed

This example demonstrates these `ODServer` methods:
- initialize
- logon
- openFolder
- logoff
- terminate

This example demonstrates these `ODFolder` methods:
- getName
- getDescription
- getNumApplGroups
- getApplGroupNames
- getNumCriteria
- getCriteria
- close

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.props` file)

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;
        String[ ] value_vec;
        String[ ] search_values, fixed_values, database_fields;
        Object[ ] appl_grps;
        int[ ] valid_oprs;
        int j, k, opr;
        char field_type;

        //----------
        // If too few parameters, display syntax and get out
        //----------
    if ( argv.length < 6 )
    {
          System.out.println( "usage: java TcListCriteria <server> <port> <userid> <password> <folder> <config dir>" );
            return;
        }
        try
        {
          //----------
          // Set the stage
          //----------
          System.out.println( "Testcase TcListCriteria started." );
          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 and names of folder application groups" );
          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( "    Required, DefaultValueAvailable, and DefaultValueFixed (true/false) flags" );
          System.out.println( "    Maximum Entry Characters" );
          System.out.println( "    Maximum Display Characters" );
          System.out.println( "    Number and names of database fields" );
          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( "" );

  ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");
  if(odConfig == null)
  {
     System.out.println( "" );
     System.out.println( "Testcase TcListCriteria failed  -" );
     System.out.println( "  ODConfig could not be initialized.  " );
     System.out.println( "  Probable cause: " );
     System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
     System.out.println( "" );
  }
  else
  {
      odServer = new ODServer(odConfig );
      odServer.initialize(  "TcListCriteria.java" );
      System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
      char conntype = 'T';
      String directory = "";
       odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);

      //----------
      // Open the specified folder and display its name and description
      //----------
      System.out.println( "Opening " + argv[4] + " folder..." );
      odFolder = odServer.openFolder( argv[4] );
      System.out.println( "Name='" + odFolder.getName( ) + "' Desc='" + odFolder.getDescription( ) + "'" );

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

      //----------
      // For each folder criteria,
      //----------
      System.out.println( "There are " + odFolder.getNumCriteria( ) + " 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.getOperator( );
          System.out.println( "  Default operator: " );
          System.out.println( "    " + getOperatorName( opr ) );

          //----------
          // Display valid operators
          //----------
          valid_oprs = odCrit.getValidOperators() ;
          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 field format for Date
          //----------
          if ( odCrit.getDefaultFmt() != null)
          {
              System.out.println("  Date Format:" );
              System.out.println("    DefaultDisplay Fmt " + odCrit.getDefaultFmt());
              System.out.println("    Display Fmt " + odCrit.getDisplayFmt());
          }
          System.out.println("    DisplayFmtQualifier " + odCrit.getDisplayFmtQual());

          //----------
          // Display field id mask info
          //----------
          System.out.println( "  FieldIdMask:" );
          System.out.println( "    Required=" + odCrit.isRequired( ) +
                  " Default=" + odCrit.isDefaultValueAvailable( ) +
                  " Fixed Default=" + odCrit.isDefaultValueFixed( ) );
          //----------
          // Display max entry chars
          //----------
          System.out.println( "  MaxEntryChars:" );
          System.out.println( "    " + odCrit.getMaxEntryChars( ) );

          //----------
          // Display max display chars
          //----------
          System.out.println( "  MaxDisplayChars:" );
          System.out.println( "    " + odCrit.getMaxDisplayChars( ) );

          //----------
```

```
                    // Display min search values
                    //----------
                    System.out.println( "  MinSearchValue:" );
                    System.out.println( "    " + odCrit.getMinSearchValue( ) );
                    //----------
                    // Display max search values
                    //----------
                    System.out.println( "  MaxSearchValue:" );
                    System.out.println( "    " + odCrit.getMaxSearchValue( ) );

                    //----------
                    // Display the database field names
                    //----------
                    System.out.println( "  Database fields:" );
                     for ( j = 0; j < appl_grps.length; j++ )
                     {
                       database_fields = odCrit.getDBFieldNames( (String)appl_grps[j] );
                       if(database_fields == null)
                            System.out.println("No DBFields Defined");
                       else{
                  System.out.println( "    " + database_fields.length + " mapping for ApplGroup '" + (String)appl_grps[j] + "':" );
                       for ( k = 0; k < database_fields.length; k++ )
                       {
                         System.out.println( "      " + "DB Field Name = " + ( database_fields[k].equals( "" ) ? );
                         System.out.println( "[empty string]" : database_fields[k] ) );
                         long fieldMask =  odCrit.getDBFieldMask((String)appl_grps[j],database_fields[k]);
                         System.out.println( "      mask: " + fieldMask);

                        System.out.println( "      ApplID Field = " +
                                  ((fieldMask & ODConstant.OD_FLDMSK_APPL )== 0
                                   ? false : true ));

                        System.out.println( "      Field Can Be Updated? = " +
                                   odCrit.isUpdateable((String)appl_grps[j],database_fields[k]));
                       }
                       }
                     }

                    //----------
                    // Display default value(s) using ODCrit.getValues( )
                    //----------
                    value_vec = odCrit.getSearchValues( );
                    System.out.println("  Default Value(s) (ODCrit.getValues method):");
                    System.out.println( "    '" + value_vec[ 0 ] + "'" );
                    System.out.println( "    '" + value_vec[ 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 TcListCriteria completed -" );
                System.out.println( "  analyze and compare results to Windows Client if required" );
                System.out.println( "" );
            }
        }
        catch ( ODException e )
        {
            System.out.println( "ODException: " + 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.OPGreaterThanEqual:
                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;
            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
- Port
- User Id

- Password
- Configuration directory (location of `arswww.props` file)

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;
        String folder_desc;
        int num_folders;

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

        try
        {
            //----------
            // Set the stage
            //----------

            System.out.println( "Testcase TcListFolders started." );
            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
            //----------
            ODConfig odConfig = BuildODConfig.build(argv[4] + "/arswww.props");
            if(odConfig == null)
            {
               System.out.println( "" );
               System.out.println( "Testcase TcListFolders failed  -" );
               System.out.println( "  ODConfig could not be initialized.  " );
               System.out.println( "  Probable cause: " );
               System.out.println( "    File arswww.props is not found in directory " +  argv[4] );
               System.out.println( "" );
            }
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize(  "TcListFolders.java" );
                System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
                char conntype = 'T';
                String directory = "";
                odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);

                //----------
                // Display the number of folders available.
                //----------
                num_folders = odServer.getNumFolders( );

                System.out.println( "" );
                System.out.println( "There are " + num_folders + " folders available to " + argv[2] + " 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 TcListFolders completed - compare results to Windows Client if required" );
                System.out.println( "" );
            }
        }
```

```
        catch ( ODException e )
        {
            System.out.println( "ODException: " + 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 folder criteria information

The following example uses `ODServer` methods to open a folder on a specified server, displays the folder name and description, and prints the folder criteria information. The folder criteria includes:

- Name
- Default operator
- Valid operators
- Field Type
- Default values
- Fixed values

To display folder criteria information, ensure that:

- None of the operators indicate `Unknown operator`.
- None of the field types indicate `Unknown type`.
- For each method, the default values are the same.
- The folder criteria information that is displayed by ODWEK API matches what is displayed by the OnDemand client.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- getNumFolders
- getFolderNames
- getFolderDescription
- logoff
- terminate

This example uses these runtime parameters:

- Server name
- Port
- User ID
- Password
- Folder criteria
- Configuration directory (contains `arswww.props`)

Example of listing folder criteria information:

```
import java.util.*;
import java.io.*;
import com.ibm.edms.od.*;
public class TcListFoldersByCrit
{
  public static void main ( String argv[ ] )
  {
     ODServer odServer;
     Enumeration folders_enum;
     String folder_name;
     String folder_desc;
     int num_folders;
```

```
//----------
// If too few parameters, display syntax and get out
//----------
if ( argv.length < 6 )
{
    System.out.println( "usage: java TcListFoldersByCrit <server> <port> <userid> <password> <Folder Criteria> <config dir>" );
    return;
}
try
{
     //----------
     // Set the stage
     //----------
     System.out.println( "Testcase TcListFoldersByCrit started." );
     System.out.println( "This testcase should:" );
     System.out.println( "  Log onto the server.");
     System.out.println( "  Display the total number of folders available for the user.");
     System.out.println( "  Display the number of folders available for the user with criteria.");
     System.out.println( "  Display one line for each folder retrieved from search criteria, ");
     System.out.println( "          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
     //----------
     ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");
     if(odConfig == null)
       {
          System.out.println( "" );
          System.out.println( "Testcase TcListFoldersByCrit failed  -" );
          System.out.println( "  ODConfig could not be initialized.  " );
          System.out.println( "  Probable cause: " );
          System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
          System.out.println( "" );
       }
     else
     {
        odServer = new ODServer(odConfig );
        odServer.initialize(  "TcListFolders.java" );
        System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
        char conntype = 'T';

        String directory = "";
        odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);

        //----------
        // Display the total number of folders available.
        //----------
        num_folders = odServer.getNumFolders( );

        System.out.println( "" );
        System.out.println( "There are " + num_folders + " folders available to user " + argv[2]);
        System.out.println( "      on server " + argv[0] + "." );

        //----------
        // Display the number of folders available with search criteria.
        //----------
        num_folders = odServer.getNumFolders(argv[4]);

        System.out.println( "" );
        System.out.println( "There are " + num_folders + " folders with criteria " + argv[4] );
        System.out.println( "      available to user " + argv[2] + " on server " + argv[0] + "." );

        //----------
        // Display the folder names and descriptions
        //----------
        int i = 1;

        // get a limited folder list based on search criteria

        System.out.println( "" );
        System.out.println("List folders found using search criteria... " + argv[4] + ":");
        System.out.println( "" );

        i = 1;
        for (Enumeration folders_enum2 = odServer.getFolderNames(argv[4]); folders_enum2.hasMoreElements( ); )
      {
      folder_name = (String)folders_enum2.nextElement( );
      folder_desc = odServer.getFolderDescription( folder_name );

      System.out.println(i++ + "  " + folder_name + "  ---  " + folder_desc );
    }

        //----------
        // Cleanup
        //----------
        odServer.logoff( );
        odServer.terminate( );

        System.out.println( "" );
        System.out.println( "-----------------------------------------------------" );
        System.out.println( "" );
        System.out.println( "Testcase TcListFoldersByCrit completed - ");
        System.out.println("          compare results to Windows Client if required" );
        System.out.println( "" );
     }
}
catch ( ODException e )
{
  System.out.println( "ODException: " + 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`, `ODHit`, and `ODCriteria` 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:
- getDisplayOrder
- getFolderSortOrder
- getCriteria
- getSortLocation
- setSortLocation
- search
- close

This example demonstrates these `ODHit` methods:
- getDisplayValue

This example demonstrates these `ODCriteria` methods:
- getName
- setSortOrder
- getAscending
- setAscending

This example also demonstrates these `ODServer` methods:
- initialize
- logon
- openFolder
- logoff
- terminate

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- 1st criteria (major sort key)
- Ascending(1) or descending(0) for 1st criteria
- 2nd criteria (less significant sort key)
- Ascending(1) or descending(0) for 2nd criteria
- 1st criteria (least significant sort key)
- Ascending(1) or descending(0) for 3rd criteria
- Configuration directory (location of `arswww.props` 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;
    ODCriteria odCrit;
    Enumeration crit_enum;
    String server, userid, password, folder, crit1, crit2, crit3;
    boolean ascend1, ascend2, ascend3;
    //----------
    // If too few parameters, display syntax and get out
    //----------
if ( argv.length < 12 )
{
  System.out.println( "usage: java TcSortedHitlist <server> <port> <userid> <password> <folder> <criteria1>" +
                      " <ascending1> <criteria2> <ascending2> <criteria3> <ascending3> <config dir>" );
  return;
}
try
{
        //----------
        // Set the stage
        //----------
        System.out.println( "Testcase TcSortedHitlist started." );
        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 default sort order");
        System.out.println("  Leave sort enablement disabled by default");
        System.out.println( "  Search the folder using the default criteria and display the Hitlist" );
        System.out.println("  Display the default sort order");
        System.out.println("  Enable sort");
        System.out.println( "  Search the folder using the default criteria and display the Hitlist" );
        System.out.println("  Set the sort order specified on the command line");
        System.out.println("  Disable sort");
        System.out.println("  Display the new sort order");
        System.out.println( "  Search the folder using the default criteria and display the Hitlist" );
        System.out.println("  Enable sort");
        System.out.println( "  Search the folder using the default criteria and display the Hitlist" );
        System.out.println("");
        System.out.println("-----------------------------------------------------");
        System.out.println("");

        //----------
        // Logon to the server
        //----------
        server = argv[0];
        userid     = argv[2];
        password   = argv[3];
        folder     = argv[4];
        crit1      = argv[5];
        ascend1    = argv[6].equals( "0" ) ?false:true;
        crit2      = argv[7];
        ascend2    = argv[8].equals( "0" ) ?false:true;
        crit3      = argv[9];
        ascend3    = argv[10].equals( "0" ) ?false:true;
        ODConfig odConfig = BuildODConfig.build(argv[11] + "/arswww.props");
        if(odConfig == null)
            System.out.println("BuildODConfig Failed.");
        else
        {
            odServer = new ODServer(odConfig );
            odServer.initialize( "TcSortedHitlist.java" );
            String directory = "";
            System.out.println( "Logging on to " + server  + " server with user " + userid + "..." );
            odServer.logon( server, userid, password, ODConstant.CONNECT_TYPE_TCPIP, Integer.parseInt(argv[1]), directory);

            //----------
            // Open the folder and display default folder sort order
            //----------
            System.out.println("Opening " + folder + "...");
            odFolder = odServer.openFolder(folder);
            System.out.println("Sort Location = " + odFolder.getSortLocation());
            displayFolderSortOrder(odFolder);
            //----------
            // Search the folder with sort disabled by default
            //----------
            System.out.println("With sort disabled by default:");
            System.out.println("  Searching with SORT=NONE...");
            searchAndListHits(odFolder);
            //----------
            // Search the folder after enabling sort
            //----------
            odFolder.setSortLocation(ODConstant.OD_SORT_LOCATION_MIDTIER);
            displayFolderSortOrder(odFolder);
            System.out.println("With Mid-tier sort enabled: USING Default Folder sort settings");
            System.out.println("Sort Location = " + odFolder.getSortLocation());
            System.out.println("  Searching folder with SORT=M and default criteria...");
            searchAndListHits(odFolder);
            //----------
            // Create a new sort order and display it
            //----------
            System.out.println("/n/nSetting specified sort order...");
            System.out
            .println("-----------------------------------------------------------------");
            for (crit_enum = odFolder.getCriteria(); crit_enum.hasMoreElements();)
            {
                odCrit = (ODCriteria) crit_enum.nextElement();
                odCrit.setSortOrder(0);
            }
            if (!(crit1.equals("-")))
            {
                odCrit = odFolder.getCriteria(crit1);
                odCrit.setSortOrder(1);
                odCrit.setAscending(ascend1);
            }
            if (!(crit2.equals("-")))
```

```
                {
                    odCrit = odFolder.getCriteria(crit2);
                    odCrit.setSortOrder(2);
                    odCrit.setAscending(ascend2);
                }
                if (!(crit3.equals("-")))
                {
                    odCrit = odFolder.getCriteria(crit3);
                    odCrit.setSortOrder(3);
                    odCrit.setAscending(ascend3);
                }
                displayFolderSortOrder(odFolder);


                //----------
                // Search the folder after disabling sort
                //----------
                odFolder.setSortLocation(ODConstant.OD_SORT_LOCATION_NONE);
                System.out.println("With sort disabled:");
                System.out.println("  Searching folder with setSortLocation(NONE) ...");
                System.out.println("Sort Location = " + odFolder.getSortLocation());
                searchAndListHits(odFolder);
                //----------
                // Search the folder after enabling sort
                //----------
                odFolder.setSortLocation(ODConstant.OD_SORT_LOCATION_MIDTIER);
                displayFolderSortOrder(odFolder);
                System.out
                .println("  Searching folder with setSortLocation(MIDTIER) and CmdLine sort order (default criteria)...");
                System.out.println("Sort Location = " + odFolder.getSortLocation());
                searchAndListHits(odFolder);
                //----------
                // Search the folder after enabling Server sort
                //----------
                odFolder.setSortLocation(ODConstant.OD_SORT_LOCATION_SERVER);
                displayFolderSortOrder(odFolder);
                System.out
                .println("  Searching folder with setSortLocation(SERVER) and CmdLine sort order (default criteria)...");
                System.out.println("Sort Location = " + odFolder.getSortLocation());
                searchAndListHits(odFolder);
                //----------
                // Cleanup
                //----------
                odFolder.close();
                odServer.logoff();
                odServer.terminate();
                System.out.println("");
                System.out.println("-----------------------------------------------------");
                System.out.println("");
                System.out.println( "Testcase TcSortedHitlist completed - Ensure that:" );
                System.out.println("  1. The default sort order is the same as that shown by the");
                System.out.println("     Windows Client");
                System.out.println("  2. The 1st hitlist is not sorted");
                System.out.println("  3. The 2nd hitlist is sorted according to the default order");
                System.out.println("  4. The new sort order is correct for the command line");
                System.out.println("  5. The 3rd hitlist is not sorted");
                System.out.println("  6. The 4th hitlist is sorted according to the specified order");
                System.out.println("");
            }
        }
        catch ( ODException e )
        {
            System.out.println("ODException: " + 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 void searchAndListHits( ODFolder odFolder )
    {
         ODHit odHit;
         Vector hits;
         String[ ] display_crit;
         String value;
         int j, k;
        try
        {
          System.out.println( "  Searching folder with default criteria..." );
            hits = odFolder.search();
            System.out.println("    Number of hits: " + hits.size());
          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);
            }
          }
        }
        catch ( ODException e )
        {
```

```
            System.out.println("ODException: " + 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 void displayFolderSortOrder( ODFolder odFolder )
    {
         ODCriteria odCrit;
         ArrayList sort_order;
         int j, order;
      try
      {
            sort_order = odFolder.getFolderSortOrder();
            System.out.println("Sort Order:");
        for ( order = -1; order < 100; order++ )
        {
          for( j = 0; j < sort_order.size( ); j++ )
          {
                    odCrit = (ODCriteria) sort_order.get(j);
                    if (odCrit.getSortOrder() == order)
                        System.out.println("  "
                                + order
                                + ". "
                                + odCrit.getName()
                                + (order <= 0 ? "" : " ("
                                    + (odCrit.getAscending() ? "ascending" : "descending")
                                    + ")"));
          }
        }
      }
      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
- logoff
- cancel
- terminate

This example demonstrates these `ODFolder` methods:

- search
- retrieve
- close

This example demonstrates these `ODHit` methods:

- getViewMimeType
- getDocType
- getDocument

- getResources
- retrieve

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Configuration directory (location of `arswww.props` file)

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[ ] res_from_server;
        byte[ ] data_from_folder;
        int j;
        int getDocResSize =0;

        //----------
        // If too few parameters, display syntax and get out
        //----------
        if ( argv.length < 6 )
        {
            System.out.println( "usage: java TcRetrieve <server> <port> <userid> <password> <folder> <config dir>" );
            return;
        }
        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TcRetrieve started." );
            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 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( "" );

            //----------
            // Logon to specified server
            //----------
            ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");
            if(odConfig == null)
            {
               System.out.println( "" );
               System.out.println( "Testcase TcRetrieve failed  -" );
               System.out.println( "  ODConfig could not be initialized.  " );
               System.out.println( "  Probable cause: " );
               System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
               System.out.println( "" );
            }
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize(  "TcDeleteDocs.java" );
                String directory = "";
                odServer.logon( argv[0], argv[2], argv[3], ODConstant.CONNECT_TYPE_TCPIP, Integer.parseInt(argv[1]), directory);

                //----------
                // Open the specified folder and search with the default criteria
                //----------
                System.out.println( "Opening " + argv[4] + " folder..." );
                odServer.cancel();
                odFolder = odServer.openFolder( argv[4] );
                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 );
                hit_to_retrieve = new Vector( );
                hit_to_retrieve.addElement( odHit );

                System.out.println( " View Mime Type = " + odHit.getViewMimeType());
                System.out.println( " Doc Type = " + odHit.getDocType());
                System.out.println( "Retrieving data from first hit using ODHit.retrieve..." );
                data_from_hit = odHit.retrieve(ODConstant.NATIVE);
                FileOutputStream fos1 = new FileOutputStream("1hitR.out");
                fos1.write(data_from_hit);
                fos1.close();

                System.out.println( "Retrieving data from first hit using ODHit.getDocument/getResources..." );
                data_from_server = odHit.getDocument();
                FileOutputStream fos2 = new FileOutputStream("getDoc.out");
                fos2.write(data_from_server);
                fos2.close();
                if(odHit.getDocType() == ODConstant.FileTypeAFP)
                {
                   try{
                   res_from_server = odHit.getResources();
                   FileOutputStream fos8 = new FileOutputStream("getRes.afp");
                   fos8.write(res_from_server);
                   fos8.close();
                   getDocResSize = data_from_server.length + res_from_server.length;
                   }
                   catch(ODException E)
                   {
                     System.out.println("There are no resources for this document.");
                     getDocResSize = data_from_server.length;
                   }
                }
                else
                    getDocResSize = data_from_server.length ;

                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( );

                if ( data_from_folder == null )
                {
                    data_from_folder = new byte[0];
                    System.out.println( "Callback function not called during ODFolder.retrieve" );
                }
                else
                {
                    FileOutputStream fos = new FileOutputStream("3folderR.out");
                    fos.write(data_from_folder);
                    fos.close();
                }
                System.out.println( "Length of data from:" );
                System.out.println( "  ODHit.retrieve=" + data_from_hit.length );
                System.out.println( "  ODHit.getDocument (+getResource) =" + getDocResSize );
                System.out.println( "  ODFolder.retrieve=" + data_from_folder.length );
                if ( data_from_hit.length == getDocResSize  )
                {
                        System.out.println( "ODHit vs. ODHit.getDoc: 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 match" );
                            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. ODHit.getDoc: 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 TcRetrieve completed - analyze the result of the comparisons" );
            System.out.println( "" );
            System.out.println( "If the arswww.props file specifies 'native' for the viewer type, all" );
            System.out.println( "lengths and data should match; otherwise, differences are expected." );
            System.out.println( "" );
        }
    }
    catch ( ODException e )
```

```
        {
            System.out.println( "ODException: " + 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( );
        }
    }
}
```

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
- printDocuments
- close

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Printer name
- Configuration directory (location of arswww.props file)

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;
        Vector hit_to_print;
        String [ ] printers;
        String printer_name;
        boolean match;
        int j;

        //----------
        // If too few parameters, display syntax and get out
        //----------
      if ( argv.length < 7 )
         {
            System.out.println( "usage: java TcPrintHit <server> <port> <userid> <password> <folder> <printer> <config dir>" );
            return;
        }

        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TcPrintHit started." );
            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
            //----------
            ODConfig odConfig = BuildODConfig.build(argv[6] + "/arswww.props");
            if(odConfig == null)
```

```
            {
               System.out.println( "" );
               System.out.println( "Testcase TcListFoldersByCrit failed  -" );
               System.out.println( "  ODConfig could not be initialized.  " );
               System.out.println( "  Probable cause: " );
               System.out.println( "    File arswww.props is not found in directory " +  argv[6] );
               System.out.println( "" );
            }
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize(  "TcPrintHit.java" );
                char conntype = 'T';
                String directory = "";
                System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
                odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);

                //----------
                // If any server printers are available on the server
                //----------
                System.out.println( "Retrieving list of server printers..." );
                printer_name = argv[5];
                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[4] + " folder..." );
                        odFolder = odServer.openFolder( argv[4] );
                        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.printDocuments( hit_to_print, printer_name,2);
                        }
                        else
                            System.out.println( "There is no document to print" );

                        odFolder.close( );
                    }
                    else
                        System.out.println( "The specified printer (" + printer_name + ") is not avilable on this server" );
                }
                else
                    System.out.println( "No printers are avilable on this server" );

                //----------
                // Cleanup
                //----------
                odServer.logoff( );
                odServer.terminate( );
                System.out.println( "" );
                System.out.println( "-----------------------------------------------------" );
                System.out.println( "" );
                System.out.println( "Testcase TcPrintHit completed - Analyze the results" );
                System.out.println( "" );
            }
        }

        catch ( ODException e )
        {
            System.out.println( "ODException: " + 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:
- getNoteStatus
- getNotes

This example uses these run-time parameters:
- Server name
- Port
- User Id
- Password
- Folder name
- Configuration directory (location of arswww.props file)

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;
        String str;
        int j, exist;
        //----------
        // If too few parameters, display syntax and get out
        //----------
```

```
if ( argv.length < 6 )
{
       System.out.println( "usage: java TcListNotes <server> <port> <userid> <password> <folder> <config dir>" );
       return;
}
try
{
    //----------
    // Set the stage
    //----------
    System.out.println( "Testcase TcListNotes started." );
    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 note status for the first hit");
    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
    //----------
    ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");
    if (odConfig == null)
    {
       System.out.println( "" );
       System.out.println( "Testcase TcListNotes failed  -" );
       System.out.println( "  ODConfig could not be initialized.  " );
       System.out.println( "  Probable cause: " );
       System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
       System.out.println( "" );
    }
    else
    {
        odServer = new ODServer(odConfig);
        odServer.initialize("TcListNotes.java");
        System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
        char conntype = 'T';
        String directory = "";
        odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);
        //----------
        // Open the specified folder and search with the default criteria
        //----------
        System.out.println( "Opening " + argv[4] + " folder..." );
        odFolder = odServer.openFolder( argv[4] );
        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);
            System.out.println("  For the first hit:");
            System.out.println("    Note status is: "
                    + odHit.getNoteStatus());
            notes = odHit.getNotes();
            if (notes.size() > 0)
                System.out.println("    There is(are) " + notes.size() + " note(s)");
            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.getDateTime());
                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());
                System.out.println("         Text=" + odNote.getText());
            }
        }
        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 TcListNotes 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.getErrorId( ) );
  System.out.println( "  msg = " + e.getErrorMsg( ) );
  e.printStackTrace( );
}
```

```
        catch ( Exception e2 )
        {
          System.out.println( "exception: " + e2 );
          e2.printStackTrace( );
        }
  }
  static String getExistenceName( int code )
  {
    String str;
    switch( code )
    {
        case ODConstant.NoteStatusUnknown:
            str = "UNKNOWN";
            break;
        case ODConstant.NoteStatusYes:
            str = "YES";
            break;
        case ODConstant.NoteStatusNo:
            str = "NO";
            break;
        case ODConstant.NoteStatusError:
            str = "ERROR";
            break;
        default:
            str = "UNDEFINED VALUE";
    }
        return str;
  }
}
```

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 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 demonstrates these `ODNote` methods:

- getGroupName
- getText
- isOkToCopy
- isPublic
- setGroupName
- setOkToCopy
- setPublic
- setText

This example uses these run-time parameters:

- Server name
- Port
- User Id
- Password
- Folder name
- Text of note
- Configuration directory (location of `arswww.props` file)

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;
    Vector notes;
    int j;

    //----------
    // If too few parameters, display syntax and get out
    //----------
    if ( argv.length < 7 )
    {
      System.out.println( "usage: java TcAddNote <server> <port> <userid> <password> <folder> <note text> <config dir>" );
      return;
    }
    try
    {
      //----------
      // Set the stage
      //----------
      System.out.println( "Testcase TcAddNote started." );
      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
      //----------
      ODConfig odConfig = BuildODConfig.build(argv[6] + "/arswww.props");

      if (odConfig != null)
      {
        odServer = new ODServer(odConfig );
        odServer.initialize( "TcAddNote.java" );
        System.out.println( "Logging on to " + argv[0] + " server with user " + argv[2] + "..." );
        char conntype = 'T';
        String directory = "";
        odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);
        //----------
        // Open the specified folder and search with the default criteria
        //----------
        System.out.println( "Opening " + argv[4] + " folder..." );
        odFolder = odServer.openFolder( argv[4] );
        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 );
          System.out.println("Working with DocID " + odHit.getDocId());
          notes = odHit.getNotes( );
          if(notes.size() > 0)
          System.out.println("  There are " + notes.size( ) + " notes for the first hit" );

          odNote = new ODNote( );
          odNote.setText( argv[5] );
          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("      UserId=" + argv[2] );
          System.out.println("      Page=1" );
          System.out.println("      Color=" + 'Y' );
          System.out.println("      Group=" + odNote.getGroupName( ) );
```

```
          System.out.println("      Offset=(0,0)" );
          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 TcAddNote completed. - Ensure that the new note was correctly" );
        System.out.println( "  added by displaying it with the Windows Client" );
        System.out.println( "" );
      }
      else
      {
         System.out.println( "" );
         System.out.println( "Testcase TcAddNote failed  -" );
         System.out.println( "  ODConfig could not be initialized.  " );
         System.out.println( "  Probable cause: " );
         System.out.println( "    File arswww.props is not found in directory " +  argv[6] );
         System.out.println( "" );
      }
    }
    catch ( ODException e )
    {
      System.out.println( "ODException: " + 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( );
    }
  }
}
```

Deleting a note

The following example uses `ODHit` methods to display the number of notes associated with the document and delete the first note.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- openFolder
- logoff
- terminate

This example demonstrates these `ODFolder` methods:

- search
- getHits
- close

This example demonstrates these `ODHit` methods:

- getDocId
- getNotes
- deleteNote

This example uses these runtime parameters:

- Server name
- Port
- User Id
- Password
- Folder name
- Text of note
- Configuration directory (contains arswww.props)

Example of deleting an annotation:

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

public class TcDeleteNote
{
  public static void main ( String argv[ ] )
  {
    ODServer odServer;
    ODFolder odFolder;
    ODHit odHit;
    ODNote odNote;
    Vector hits;
    Vector notes;
    int j, numNotes1 = 0, numNotes2 = 0;

    //----------
    // If too few parameters, display syntax and get out
    //----------
    if ( argv.length < 6 )
    {
      System.out.println( "usage: java TcDeleteNote <server> <port> <userid> <password> <folder> <config dir>" );
      return;
    }
    try
    {
      //----------
      // Set the stage
      //----------
      System.out.println( "Testcase TcDeleteNote started." );
      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( "  Delete first note" );
      System.out.println( "" );
      System.out.println( "-------------------------------------------------" );
      System.out.println( "" );

      //----------
      // Logon to specified server
      //----------
      ODConfig odConfig = BuildODConfig.build(argv[5] + "/arswww.props");

      if (odConfig != null)
      {
        odServer = new ODServer(odConfig );
        odServer.initialize( "TcDeleteNote.java" );
        System.out.println( "Logging on to " + argv[0] + " server with user " + argv[2] + "..." );
        char conntype = 'T';
        String directory = "";
        odServer.logon( argv[0], argv[2], argv[3], conntype, Integer.parseInt(argv[1]), directory);
        //----------
        // Open the specified folder and search with the default criteria
        //----------
        System.out.println( "Opening " + argv[4] + " folder..." );
        odFolder = odServer.openFolder( argv[4] );
        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);
   System.out.println("Working with DocID " + odHit.getDocId());
   notes = odHit.getNotes();
   numNotes1 = notes.size();
   System.out.println("  There are " + numNotes1 + " notes for the first hit");

   if (numNotes1 > 0)
   {
    odNote = (ODNote)notes.elementAt(0);
    System.out.println("   Delete in the first note...");
    odHit.deleteNote(odNote);

    // destroy hits
    hits.removeAllElements();

    // search again
    System.out.println("Searching second time with difault criteria...");
    odFolder.search();
    hits = odFolder.getHits();
    System.out.println("  Number of hits:  " + hits.size());

    // Retrieve notes again
    if (hits.size() > 0)
    {
```

```
    odHit = (ODHit)hits.elementAt(0);
    System.out.println("Working with DocID " + odHit.getDocId());
    notes = odHit.getNotes();
    numNotes2 = notes.size();
    System.out.println("  There are " + numNotes2 + "notes for the first hit");
   }
   else
    System.out.println("No document - cannot list notes");
  }
  else
   System.out.println("No document - cannot list notes");

  if ((numNotes1 - numNotes2) == 1)
   System.out.println("\nSuccess!");
  else if (numNotes1 == 0)
   System.out.println("\nSuccess!");
  else
   System.out.println("\nFailed!");
 }
 else
  System.out.println("\nNo hits found");

      //----------
      // Cleanup
      //----------
      odFolder.close( );
      odServer.logoff( );
      odServer.terminate( );
      System.out.println( "" );
      System.out.println( "-----------------------------------------------------" );
      System.out.println( "" );
      System.out.println( "Testcase TcDeleteNote completed. - Ensure that the new note was correctly" );
      System.out.println( "  added by displaying it with the Windows Client" );
      System.out.println( "" );
    }
    else
    {
     System.out.println( "" );
     System.out.println( "Testcase TcDeleteNote failed  -" );
     System.out.println( "  ODConfig could not be initialized.  " );
     System.out.println( "  Probable cause: " );
     System.out.println( "    File arswww.props is not found in directory " +  argv[5] );
     System.out.println( "" );
    }
   }

   catch ( ODException e )
   {
     System.out.println( "ODException: " + 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( );
   }
  }
}
```

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 one or more database values.

This example demonstrates these `ODServer` methods:

- initialize
- logon
- openFolder
- logoff
- terminate

This example demonstrates these `ODFolder` methods:

- getFolderSortOrder

- getDisplayOrder
- getCriteria
- search
- close

This example demonstrates these `ODCriteria` methods:
- setOperator
- setSearchValues
- setSearchValue

This example demonstrates these `ODHit` methods:
- getDocId
- getDisplayValue
- updateValuesForHit

This example uses these run-time parameters:
- Server name
- Port
- 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.props` 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, odCrit2;
    ODHit odHit;
    Hashtable hash;
    Vector hits;
    String[ ] display_crit;
    String line;
    String crit1;
    String crit2;
    String value1;
    String value2;
    String new_value;
    int j;

    //----------
    // If too few parameters, display syntax and get out
    //----------
    if ( argv.length < 10 )
    {
           System.out.println( "usage: java TcUpdate <server> <port> <userid> <password> <folder>" );
           System.out.println( "    <criteria1> <value1> <criteria2> <value2> <new value2> <config dir>" );
      return;
    }
    try
    {
      System.out.println( "Testcase TcUpdate started." );
      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( "" );
```

```
//----------
//Setup ODConfig object using defaults and initialize ODServer.
//----------
ODConfig odConfig = BuildODConfig.build(argv[10] + "/arswww.props");
if(odConfig == null)
{
   System.out.println( "" );
   System.out.println( "Testcase TcUpdate failed  -" );
   System.out.println( "  ODConfig could not be initialized.  " );
   System.out.println( "  Probable cause: " );
   System.out.println( "    File arswww.props is not found in directory " +  argv[10] );
   System.out.println( "" );
}
else
{
   odServer = new ODServer(odConfig);
   odServer.initialize( "TcUpdate.java");

   //----------
   // Logon to specified server
   //----------
   System.out.println( "Logging on to " + argv[0]  + " server with user " + argv[2] + "..." );
   String directory = "";
   odServer.logon( argv[0], argv[2], argv[3], ODConstant.CONNECT_TYPE_TCPIP, Integer.parseInt(argv[1]), directory);

   //----------
   // Open the specified folder and set the requested criteria
   //----------
   crit1 = argv[5];
   crit2 = argv[7];
   value1 = argv[6];
   value2 = argv[8];
   new_value = argv[9];
   System.out.println( "Opening " + argv[4] + " folder..." );
   odFolder = odServer.openFolder( argv[4] );

   ArrayList allCrit =odFolder.getFolderSortOrder();
   /*Clear all default search values*/
   for (int a =0; a < allCrit.size(); a++)
   {
       odCrit = (ODCriteria)allCrit.get(a);
       odCrit.setSearchValues("","");
   }
   odCrit = odFolder.getCriteria( crit1 );
   if(odCrit == null)
   System.out.println("ERROR: Search Criteria Not Found [" +crit1+"]");
   else
   {
      odCrit.setOperator( ODConstant.OPEqual );
      odCrit.setSearchValue( value1 );
   }
   odCrit2 = odFolder.getCriteria( crit2 );
   if(odCrit2 == null)
     System.out.println("ERROR: Search Criteria Not Found [" +crit2+"]");
   else
   {
      odCrit2.setOperator( ODConstant.OPEqual );
      odCrit2.setSearchValue( value2 );
   }
   //----------
   // Search the folder
   //----------
   if (odCrit != null && odCrit2 != null)
   {
       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:" );
        odHit = (ODHit)hits.elementAt( 0 );
        System.out.println( "      DOCID = "+ odHit.getDocId() );
        line = "      ";
        display_crit = odFolder.getDisplayOrder( );
        for( j = 0; j < display_crit.length; j++ )
          line = line + display_crit[j] + " ";
        System.out.println( line );
        line = "      ";
        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.updateValuesForHit( hash );
                System.out.println("New display values as set in ODHit");
                odHit = (ODHit)hits.elementAt( 0 );
                line = "      ";
                System.out.println( "      DOCID = "+ odHit.getDocId() );
                display_crit = odFolder.getDisplayOrder( );
                for( j = 0; j < display_crit.length; j++ )
                   line = line + display_crit[j] + " ";
                System.out.println( line );
                line = "      ";
                for ( j = 0; j < display_crit.length; j++ )
                    line = line + odHit.getDisplayValue( display_crit[j] ) + " ";
                System.out.println( line );
              }
             else
                System.out.println( "There were no hits" );
            }
            //----------
            // Cleanup
            //----------
            odFolder.close( );
            odServer.logoff( );
            odServer.terminate( );
            System.out.println( "" );
            System.out.println( "-----------------------------------------------------" );
            System.out.println( "" );
            System.out.println( "Testcase TcUpdate completed - Using the Windows Client," );
            System.out.println( "  ensure that the value has been changed." );
            System.out.println( "" );
          }
        }
        catch ( ODException e )
        {
          System.out.println( "ODException: " + 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( );
        }
      }
    }
```

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
- getPassword
- changePassword
- setUserId
- setPassword
- setServerName
- logoff
- terminate

This example uses these run-time parameters:

- Server name
- Port
- User Id
- Password
- New password
- First logon
- Configuration directory (location of `arswww.props` file)

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 < 7 )
         {
            System.out.println( "usage: java TcChangePassword <server> <port> <userid> <password> " );
            System.out.println( "                                   <new password> <initial logon> <config dir>" );
            return;
        }
        try
        {
            //----------
            // Set the stage
            //----------
            System.out.println( "Testcase TcChangePassword started." );
            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( "" );
            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[2];
            original_password = argv[3];
            new_password = argv[4];
            char conntype = 'T';
            String directory = "";
            int logon_first= Integer.parseInt(argv[5]);

            ODConfig odConfig = BuildODConfig.build(argv[6] + "/arswww.props");
            if(odConfig == null)
            {
               System.out.println( "" );
               System.out.println( "Testcase TcGetFolders failed  -" );
               System.out.println( "  ODConfig could not be initialized.  " );
               System.out.println( "  Probable cause: " );
               System.out.println( "    File arswww.props is not found in directory " +  argv[6] );
               System.out.println( "" );
            }
            else
            {
                odServer = new ODServer(odConfig );
                odServer.initialize( "TcChangePassword.java" );

                if(logon_first == 1)
                {
                    //----------
                    // Logon to the server using the original password
                    //----------
                    System.out.println( "Logging on to " + server + " using original password..." );
                    odServer.logon( server, userid, original_password, conntype, Integer.parseInt(argv[1]), directory);
                }
                else
                {
                    System.out.println("Do NOT logon, but set the pre-req fields of UID,PWD,and Server");
                    odServer.setUserId(userid);
                    odServer.setPassword(original_password);
                    odServer.setServerName(server);
                }
                //----------
                // Change to the new password and logoff
                //----------
                System.out.println( "Changing to new password..." );
                odServer.changePassword( new_password );
                System.out.println( "Current ODServer.password is " + odServer.getPassword() );
                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..." );
                odServer.logon( server, userid, new_password, conntype, Integer.parseInt(argv[1]), directory );

                //----------
                // Change back to the original password and logoff
                //----------
                System.out.println( "Changing back to original password..." );
                odServer.changePassword( original_password );
                System.out.println( "Current ODServer.password is " + odServer.getPassword() );
                System.out.println( "Logging off..." );
                odServer.logoff( );

                //----------
                // Cleanup
                //----------
                odServer.terminate( );
                System.out.println( "" );
                System.out.println( "-----------------------------------------------------" );
                System.out.println( "" );
                System.out.println( "Testcase TcChangePassword completed successfully." );
                System.out.println( "" );
```

```
                }
            }
            catch ( ODException e )
            {
                System.out.println( "ODException: " + 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( );
            }
        }
    }
```

Appendix E. AFP to HTML transform

The AFP to HTML transform process converts AFP documents and resources into HTML documents. The AFP to HTML transform process requires the AFP2WEB Transform. An administrator must install and configure the AFP2WEB Transform on the HTTP server. See your IBM representative for more information about the AFP2WEB Transform. 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.

Important: 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 25.

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

Important: 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 for CGI and Java servlet or ARSWWW.PROPS file for Java API. See "AFPVIEWING" on page 36 for details. (If you plan to use the Retrieve Document API, then you should specify the `_afp=HTML` parameter. See "Retrieve Document" on page 85 for details.) You must also specify the directory that contains the AFP2WEB Transform programs. (See "CONFIGFILE" on page 25.)

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), then 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" briefly describes how they are used by the AFP2WEB Transform.

Options for the AFP2WEB Transform

Table 13 lists the options that you can specify in the AFP2HTML.INI file to convert documents with the AFP2WEB Transform.

Table 13. 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. **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 at the client.
`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.

Table 13. Options for the AFP2WEB Transform (continued)

Option in AFP2HTML.INI file	Description
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 imagfont.cfg and resides in the directory that contains the AFP2WEB Transform programs. See the AFP2WEB Transform documentation for details about the Font Map file.
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.

Important: 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), then 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 F. AFP to PDF transform

The AFP2PDF Transform converts AFP documents and resources into PDF documents. An administrator must install and configure the AFP2PDF Transform on the HTTP server. See your IBM representative for more information about the AFP2PDF Transform. 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.

Important: 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 27.

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

Important: 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 for CGI and Java servlet or ARSWWW.PROPS file for Java API. See "AFPVIEWING" on page 36 for details. (If you plan to use the Retrieve Document API, then you should specify the `_afp=PDF` parameter. See "Retrieve Document" on page 85 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 AFP2PDF Transform programs. To specify the directory that contains the programs for the AFP2PDF Transform, see "INSTALLDIR" on page 27. 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 client. **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 at the client.

Viewing converted documents

To view converted documents with the Adobe Acrobat viewer, you must obtain the viewer for the browsers used by your organization.

Appendix G. HTTP server configuration files

This section contains examples of the following HTTP server configuration files:

- HTTP Apache Server
- WebSphere Application Server

Important: Please consult your HTTP documentation for configuration assistance.

HTTP Apache Server

The following sample shows an HTTP Server configuration file. (You must first create the HTTP server configuration by using the HTTP Administration client.) The sample configuration included is for instance QUSROND and HTTP server ODAPACHE. The configuration items in bold are required for or related to ODWEK. Configuration notes are bolded, italicized, and in parentheses. These notes should not be entered into the configuration file. Comment lines start with #. The file name and path in IFS on the server where this configuration file is located is: /www/odapache/conf/httpd.conf

Notes®:

1. The 'original' HTTP server is no longer supported.
2. ODWEK requires that the end-user browser accept UTF-8 format. In Microsoft Internet Explorer, select **Tools > Internet Options**, then select the Advanced tab. Under Browsing, select Always send URLs as UTF-8.

```
============================================================================================
Listen *:ppppp    (Set to the port that the browser uses to connect to ODWEK.
The default TCP/IP port number is 80.)
DocumentRoot /www/odapache/htdocs
ServerRoot /www/odapache
DefaultType text/plain
Options -ExecCGI -FollowSymLinks -SymLinksIfOwnerMatch -Includes-IncludesNoExec
-Indexes -MultiViews
ErrorLog logs/error_log
LogLevel Warn
DirectoryIndex index.html
HostNameLookups off
RuleCaseSense OFF
LimitRequestBody 102400
ServerName f.q.h.n  (Enter the fully-qualified host name or the TCP/IP address of
the server running ODWEK.)
UseCanonicalName Off
DefaultFsCCSID  0037 (This should be the CCSID of the OnDemand instance.)
DefaultNetCCSID 1208
CGIConvMode EBCDIC   (For DBCS languages, change this to EBCDIC_JCD)
ScriptLogLength 200
ScriptLog /www/odapache/logs/cgi_log
Alias       /logon                                /www/odapache/htdocs/oda_logon.html
                                (The above line points to your logon script name and location.)
AliasMatch ^/images/(.*)$                        /www/odapache/htdocs/images/$1
AliasMatch ^/applets/com/ibm/edms/od/(.*)$  /QIBM/ProdData/OnDemand/www/applets/$1
AliasMatch ^/applets/(.*)$                       /QIBM/ProdData/OnDemand/www/applets/$1
ScriptAliasMatch ^/scripts/arswww.cgi$        /QSYS.LIB/QRDARS.LIB/ARS3WCGI.PGM
ScriptAliasMatch ^/scripts/arswww\.cgi/(.*)$  /QSYS.LIB/QRDARS.LIB/ARS3WCGI.PGM
AddType www/unknown cab
AddType www/unknown jar
AlwaysDirectoryIndex On
DirectoryIndex index.html
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{Cookie}n \"%r\" %t" cookie
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined
```

```
SetEnvIf "User-Agent" "Mozilla/2" nokeepalive
SetEnvIf "User-Agent" "JDK/1\.0" force-response-1.0
SetEnvIf "User-Agent" "Java/1\.0" force-response-1.0
SetEnvIf "User-Agent" "RealPlayer 4\.0" force-response-1.0
SetEnvIf "User-Agent" "MSIE 4\.0b2;" nokeepalive
SetEnvIf "User-Agent" "MSIE 4\.0b2;" force-response-1.0

# Root directory access authority
<Directory />
Order Deny,Allow
   Deny From all
   Options None
   Options +ExecCGI
   AllowOverride NoneLimit
<Except GET HEAD OPTIONS TRACE POST PUT>
</LimitExcept>
</Directory

# Directory access for the HTTP server directory documents
<Directory /www/odapache/htdocs>
Order Allow,Deny
   Allow From all
</Directory>

# Directory access to the ProdData ..... line data applets directory
<Directory /QIBM/ProdData/OnDemand/www/applets/>
Order Allow,Deny
   Allow From all
</Directory>

# Directory access to the ProdData .... images directory
<Directory /QIBM/ProdData/OnDemand/www/images>
Order Allow,Deny
   Allow From all
Options +FollowSymLinks
</Directory>

# Directory access to ProdData .... samples directory
<Directory /QIBM/ProdData/OnDemand/www/samples/>
Order Allow,Deny
   Allow From all
Options +FollowSymLinks
</Directory>

# Directory access to the HTTP server instance directory
<Directory /www/odapache/>
Order Allow,Deny
   Allow From all
</Directory>
# Directory access to the QRDARS library
<Directory /QSYS.LIB/QRDARS.LIB>
Order Allow,Deny
   Allow From all
</Directory>

================================================================================
```

WebSphere Application Server

A sample WebSphere configuration file can be obtained from the IBM Content Manager OnDemand for i support Web page at http://www.ibm.com/software/data/ondemand/400/support.html. Search for ODWEK WebSphere configuration to locate the information.

Appendix H. 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.

Delimited ASCII output

The delimited ASCII output generated by ODWEK is a set of output records. These records contain character string values, keywords, and function, record, and string delimiters and separators:

- Character string values are the output data of a function, other than keywords, delimiters, and separators. For example, character string values include the next function to be called, the name of the folder, the folder field names, search operators, and field 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 (^), 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 can 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:

  ```
  [1^2^4^8^16^32]
  ```

  You can override the default characters for the string delimiters and separators. See "[NO HTML]" on page 34 for details.
- A single null character string value is indicated by the absence of a value inside two double quotation 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]\n
ACTION=searchCriteriaUrl\n
FOLDER=[folderName^folderDesc]\n
FOLDER=[folderName^folderDesc]\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 in double quotation marks.
3. The string `folderDesc` is the description of the folder. The description is not in double quotation marks.

Search Criteria

The following example shows delimited ASCII data generated by the Search Criteria function:

```
[BEGIN]\n
ACTION=hitListUrl\n
DISPLAY_ORDER=[field1^field2^...fieldN]\n
NUMROWS=numberOfRows\n
ROW=[criteriaName^[[validOp]^defOp]^[inpType^inpAssocData]\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 in double quotation marks.
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`. See Table 14.

Table 14. Default operator and input type associated with inpAssocData

defOp	inpType	inpAssocData

Between, Not Between	N	Null: [ ]
		or a list: [defaultField1∧...∧defaultField*N*]
		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 example shows delimited ASCII output generated by the Document Hit List function:

```
[BEGIN]\n
ACTION=hitListURL\n
MSG=Only 20 documents can be listed for this folder.
DOC=[criteria1∧criteria2∧criteriaN∧docid∧fileType∧docLocation]\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 in double quotation marks.
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 example shows delimited ASCII output generated by the View Annotations function:

```
[BEGIN]\n
NOTE 4: 15:42:44 PM Mountain Standard Time Thursday November 19, 1998...\n
Public - Cannot be copied to another server\n
Test note from the OnDemand Internet Client.\n
[END]\n
```

Error Message

The following example shows delimited ASCII output generated when errors occur:

```
[ERROR]\n
ID=nnnn\n
MSG=errorMessageText\n
```

Notes

1. The string `nnnn` is the error message number.
2. The string `errorMessageText` is the error message text.

Appendix I. National language support

Configuring ODWEK for DBCS languages

This section contains information that may help administrators configure ODWEK for DBCS languages.

The CODEPAGE and LANGUAGE parameters in the ARSWWW.INI file for CGI and Java servlet, or the LANGUAGE parameter in the ARSWWW.PROPS file for Java APIs are used to specify National Language (NL) configuration options.

The CODEPAGE parameter identifies the code page of the ODWEK server and needs to be compatible with OnDemand database on the OnDemand library server. The CODEPAGE parameter needs to be specified **only** if the code page of the workstation on which you are running your ODWEK application is different than the code page of the OnDemand database on the OnDemand library server. The system uses the code page of the workstation on which the ODWEK application is running as the default value.

The LANGUAGE parameter determines the message catalog that ODWEK uses to display messages.

Table 15 lists the DBCS code pages and languages supported by OnDemand. The **CODEPAGE=** column lists the value for the code page, and needs to be specified **only** if the code page of the workstation on which you are running your ODWEK application is different than the code page of the OnDemand database. The **LANGUAGE=** column lists the values that are associated with the translated message catalogs.

Table 15. DBCS languages, code pages, and locales

Country or region	LANGUAGE=	CODEPAGE=	Locale
China (PRC)	CHS	1388	ZH_CN
Japan	JPN	5035	JA_5035
Korea, South	KOR	933	KO_KR
Taiwan	CHT	937	ZH_TW

Code page conversion in ODWEK

In contrast to the standard OnDemand client, ODWEK behaves differently when it converts code pages. Because ODWEK is a mid-tier system, there is always an additional presentation layer. In most cases, the layer is a browser; however, it can also be a stand-alone Java application that uses the ODWEK API.

ODWEK internally runs in UTF-8, which leads to a conversion of all index and annotation data from UTF-16 (the format in which the data are sent in TCP/IP) to UTF-8.

When you implement a Java application that uses ODWEK, ensure that you correctly handle the information that you pass to and retrieve from ODWEK API. Because Java works in UTF-16 Unicode internally, you can ignore data that are returned from ODWEK API functions. Java handles the conversion from UTF-8.

When you pass strings to ODWEK methods, you do not need to perform any additional tasks because Java supports only string variables that are in UTF-16. The conversion to UTF-8 is done by native subroutines of ODWEK.

Despite the internal conversion to UTF-8, ODWEK does not do any other conversion on indexes. Therefore, a client that displays index data that is received through the ODWEK API must be capable of handling UTF-8 Unicode data. If you deliver index data to any external applications by using your ODWEK-based Java application, you must ensure that those applications can handle Unicode data or you must convert the data manually. The same implications apply if you want to save any indexes or annotation data to file. If you do not perform any explicit conversion, the data is written as a Unicode data stream. As Web browsers usually are capable of displaying and sending UTF-8 data, it is not a problem when you implement Web applications.

For the document data, you can handle the conversion in different ways. If you request raw native document data, ODWEK returns the data in its unaltered form: in the same code page in which it was archived. When you request the line data to be displayed by using an applet, ODWEK sends the UTF-8 ASCII data to the applet, but only standard HTML code containing applet invocation code is returned. When you request an ASCII conversion, ODWEK returns a UTF-8 ASCII converted representation of the original AFP or line data document. For most other document types, ODWEK works the same as the OnDemand Windows client: it passes the data through in the data's native format.

ICU conversion library

OnDemand might need to convert or map from one code page to another code page. This conversion or mapping is done by using a standard component called International Components for Unicode (ICU).

The International Components for Unicode (ICU) is an open source project developed by IBM and other companies. It is a library that is available for Java and C and is used for internationalization. ICU provides services such as character conversion between different code pages and language-sensitive collation, searching, normalization, and locale information.

Earlier versions of OnDemand used the ICONV library, another character set conversion engine. Starting from Version 7.1.2.1, OnDemand uses ICU. ICONV is a character encoding library that is primarily distributed on UNIX® environments. For example, it is part of the GNU C library in most Linux distributions.

OnDemand uses ICU at different locations for code page conversion and text operations.

Each component uses ICU for different use cases, but all do code page conversions, which enables communication with other parts of the OnDemand infrastructure:

- The OnDemand server uses the ICU facility for index and annotation communication. All TCP/IP traffic is in UTF-16, but the index data is in the database code page format. The OnDemand server uses ICU for data conversion between the database or instance code page and UTF-16.
- ODWEK needs the ICU facility to convert TCP/IP data in UTF-16 to UTF-8. Also, if you use line data in the line data Java applet, then the data gets converted from its original code page to UTF-8, which is used by the applet. Other documents are passed over without code page change.

- The standard OnDemand client uses its ICU facility for converting AFP and line data so that they can be properly viewed in the internal viewers. The data gets converted to the local Windows code page.
- The AFP plug-in, which works similarly to the AFP viewer within the OnDemand client, has its own ICU library for converting AFP data into the local Windows code page.

Appendix J. Problem determination tools

You can use the tools listed in Table 16 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 plug-ins.

Table 16. 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
Content Manager OnDemand Web Enablement Kit System Tracing	Save access information, errors, and server information.	Perform the following tasks: 1. In the DEBUG section of the ARSWWW.INI file for CGI and Java servlet or ARSWWW.PROPS file for Java API, set the `TRACE` parameter to `1`, `2`, `3` or `4`, depending on the level of trace desired. For the Java APIs, use the appropriate ODConfig constructor to specify this. The trace file that is generated by ODWEK is named ARSWWW.TRACE and is written to the directory specified by the `TRACEDIR` parameter. (The default directory is `/QIBM/UserData/OnDemand/WWW/LOGS`.) **Important:** For CGI and Java servlet, if specified, the DEBUG section must be the first executable statement in the ARSWWW.INI file. 2. Configure logging for your HTTP server. Each HTTP server can have a different way to configure logging and can have different logs and options that you can enable to collect more or less detailed information. **Important:** 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.

Table 16. Problem determination tools (continued)

Tool	Purpose	How to enable

AFP Web Viewer Trace Facility	Capture detailed information about AFP documents being viewed with the AFP Web Viewer.	Make sure the following section exists in the FLDPORT2.INI file on the user's workstation:
		`[Misc]`
		`ViewTraceFile=d:\temp\afpplgin.log`
		`Trace=TRUE`
		
		Verify the path of the log file. Remember to turn off logging when you have gathered the information you need.

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.	Perform the following tasks: 1. Enable system and application group logging for the OnDemand server. Update the system parameters for the server by using the administrative client. 2. Enable the specific application group messages that you want to log. Update the message logging options for the application group by using the administrative client.

Java Dump

During the run time of a Java process, some Java Virtual Machines (JVMs) might not respond predictably and seem to freeze for a long time or until a JVM shutdown occurs. It is not easy to determine the root cause of these problems.

By triggering a javacore when a Java process does not respond, you might be able to collect diagnostic information that is related to the JVM and a Java application captured at a particular point. For example, the information can be about the operating system, the application environment, threads, native stack, locks, and memory. The exact contents depend on the platform on which the application is running.

On some platforms and in some cases, a javacore is known as a ″javadump.″ The code that creates a javacore is part of the JVM. You can control it by using environment variables and runtime switches. By default, a javacore occurs when the JVM terminates unexpectedly. A javacore can also be triggered by sending specific signals to the JVM. Although a javacore or javadump is present in Sun Solaris JVMs, much of the content of a javacore is added by IBM and, therefore, is present only in IBM JVMs.

The IBM Thread and Monitor Dump Analyzer for Java analyzes javacore and diagnoses monitor locks and thread activities to identify the root cause of hangs, deadlocks, and resource contention or monitor bottlenecks.

IBM Thread and Monitor Dump Analyzer

This technology analyzes each thread and provides diagnostic information, such as current thread information, the signal that caused the javacore, Java heap information (maximum Java heap size, initial Java heap size, garbage collector

counter, allocation failure counter, free Java heap size, and allocated Java heap size), number of runnable threads, total number of threads, number of monitors locked, and deadlock information.

In addition, IBM Thread and Monitor Dump Analyzer for Java Technology provides the recommended size of the Java heap cluster (applicable only to IBM SDK Version 1.4.2 and Version 1.3.1 SR7 or later) based on the heuristic analysis engine.

IBM Thread and Monitor Dump Analyzer for Java compares each javacore and provides process ID information for threads, time stamp of the first javacore, time stamp of the last javacore, number of garbage collections per minute, number of allocation failures per minute, time between the first javacore and the last javacore, number of hang suspects, and list of hang suspects.

This technology also compares all monitor information in javacore and detects deadlock and resource contention or monitor bottlenecks, if there are any.

For more information on IBM Thread and Monitor Dump Analyzer, see http://www.alphaworks.ibm.com/tech/jca.

Java diagnostic commands

jmap

jmap is a command-line utility that is included in the Solaris Operating Environment and Linux (but not Windows) releases of the Java Development Kit (JDK). This utility prints memory–related statistics for a running JVM or core file. If jmap is used without any command line options, then it prints the list of shared objects loaded, similar to what the Solaris pmap utility outputs. For more specific information, you can use the –heap, –histo, or –permstat options.

-heap Use the –heap option to obtain information that includes the name of the garbage collector, algorithm-specific details, such as the number of threads used for parallel garbage collection, heap configuration information, and a heap usage summary.

-histo Use the -histo option to obtain a class-wise histogram of the heap. For each class, it prints the number of instances in the heap, the total amount of memory consumed by those objects in bytes, and the fully qualified class name. The histogram is useful when you want to understand how the heap is used.

-permstat
Configuring the size of the permanent generation can be important for applications that dynamically generate and load a large number of classes, for example, Java Server Pages and Web containers. If an application loads too many classes, then an `OutOfMemoryError` exception is thrown. Use the –permstat option to the jmap command to get statistics for the objects in the permanent generation.

jstat

The jstat utility uses the built–in instrumentation in the HotSpot JVM to provide information on performance and resource consumption of running applications. You can use the jstat utility to diagnose performance issues, especially issues that

are related to heap sizing and garbage collection. Some of its many options can print statistics regarding garbage collection behavior and the capacities and usage of the various generations.

HPROF: Heap Profiler

HPROF is a simple profiler agent that is shipped with JDK Version 5.0. It is a dynamically linked library that interfaces to the JVM by using the Java Virtual Machine Tools Interface (JVM TI). It writes profiling information either to a file or to a socket in ASCII or binary format. This information can be further processed by a profiler front–end tool.

HPROF can present CPU usage, heap allocation statistics, and monitor contention profiles. In addition, it can output complete heap dumps and report the states of all the monitors and threads in the Java virtual machine. HPROF is useful when you analyze performance, lock contention, memory leaks, and other issues.

HAT: Heap Analysis Tool

The Heap Analysis Tool (HAT) helps debug unintentional object retention. This term is used to describe an object that is no longer needed but is kept alive because of references through some path from a live object. HAT provides a convenient means to browse the object topology in a heap snapshot that is generated by using HPROF. The tool allows a number of queries, including “show me all reference paths from the rootset to this object.

Diagnostic Tool for Java Garbage Collector

This technology is a diagnostic tool for optimizing parameters affecting the garbage collector when using the IBM JavaTM Virtual Machine (JVM).

Applications that run under Java use a Java heap for garbage collection, which serves as a storage manager in IBM’s Java development kits and runtime environments.

An analysis of data reflecting the activity of the garbage collector in Java enterprise or stand-alone applications is critical to optimizing tasks running under a JVM. For example, issues such as the frequency of the garbage collection cycle, the time spent in different phases of the garbage collection, the quantities of heap memory involved in the process, the characteristics of the allocation failures from which the garbage collection originate, and the unwanted presence of stack overflows must be considered in optimization of parameters for Java applications and prevention of bottlenecks.

Diagnostic Tool for Java Garbage Collector helps to examine the characteristics of the garbage collection for an application running under a JVM by reading the output of the ″verbose″ garbage collection and producing textual and graphical visualizations and related statistics. This tool is particularly well suited for looking at the garbage collector activity of a heavily-accessed enterprise application hosted by a WebSphere Application Server. The tool includes a ″multiple file analysis″ modality that allows the loading of two or more files simultaneously, thereby enabling the comparison of the behavior of two or more application servers in a WebSphere cluster.

Diagnostic Tool for Java Garbage Collector Version 1.3 comes with three built-in parsers for IBM JVM Version 1.5.0, IBM JVM Version 1.4.2, and IBM JVM Version 1.2.2, and sample files for evaluating features of the tool and detailed

documentation. Each parser is likely to be able to parse the data that is produced by some other versions of the IBM JVM; for instance, the JVM Version 1.4.2 parser also works well with the JVM Version 1.3.x. For unsupported versions of the IBM JVM, the tool contains detailed documentation that allows one to code a different parser in Java by implementing the GCParser interface. The documentation describes in detail the assumptions and the rules necessary for coding of the new parser

For more additional information and product download, see http://www.alphaworks.ibm.com/tech/gcdiag.

HeapAnalyzer

HeapAnalyzer finds a possible Java heap leak area through its heuristic search engine and analysis of the Java heap dump in Java applications.

Java heap areas define objects, arrays, and classes. When the Garbage Collector allocates areas of storage in the heap, an object continues to be live while a reference to it exists somewhere in the active state of the JVM; therefore, the object is reachable. When an object ceases to be referenced from the active state, it becomes garbage and can be reclaimed for reuse. When this reclamation occurs, the Garbage Collector must process a possible finalizer and also ensure that any internal JVM resources that are associated with the object are returned to the pool of such resources. Java heap dumps are snap shots of Java heaps at specific times.

HeapAnalyzer analyzes Java heap dumps by parsing the Java heap dump, creating directional graphs, transforming them into directional trees, and running the heuristic search engine.

The following are examples of features of HeapAnalyzer:

- List of Java heap leak suspects
- Recommendation of the size of kCluster
- List of gaps among allocated objects/classes/arrays
- Java objects/classes/arrays search engine
- List of objects/classes/arrays by type name
- List of objects/classes/arrays by object name
- List of objects/classes/arrays by address
- List of objects/classes/arrays by size
- List of objects/classes/arrays by size of child
- List of objects/classes/arrays by number of child
- List of objects/classes/arrays by frequency
- List of available heap spaces by size
- Tree view of Java heap dump Loading/saving processed Java heap dumps

HeapRoots

HeapRoots is a tool for debugging memory leaks in Java applications through analysis of heap dumps.

The Java Virtual Machine (JVM) maintains a runtime data area (called a heap) for the allocation of all class instances and array objects. The heap storage for objects is automatically reclaimed by a storage management system known as the garbage collector. If an application requires more heap space than can be made available by the garbage collector, the JVM throws an OutOfMemoryError.

HeapRoots analyzes heap dumps, which are files (typically text files) containing information about the objects in the JVM garbage collected heap.

Some IBM VMs (contained in the IBM Developer Kits for Windows, Java Edition) can produce heap dumps on demand; heap dumps can also be triggered by out-of-memory situations.

HeapRoots loads these heap dump files and provides commands for analyzing the data. These commands run algorithms on the data or query for information about the data. HeapRoots provides a command-line interactive interface where you enter commands and get results. Examples of analysis include:

- Searching or filtering of individual objects
- Summary or tabulation of the various types of objects
- Statistics on heap address space (such as gaps between objects)
- Inward and outward references of an object
- Paths between two objects
- Exploring the heap from source or root objects by following references
- Calculation of objects reachable by an object
- Calculation of objects kept alive by an object

Appendix K. Multilingual support for CGI using the Apache HTTP server

Multilingual support has been added to the OnDemand Web Enablement Kit (ODWEK) Common Gateway Interface (CGI) program. Multilingual support allows you to have Apache HTTP servers supporting OnDemand Common Server instances running in different languages on a single IBM i server.

Software prerequisites

The following software is required for ODWEK multilingual support for CGI:

- OnDemand Common Server environment.

Implementation

Important: For a similar version of the following information that includes screen captures, go to the OnDemand Support web page at http://www.ibm.com/software/data/ondemand/400/support.html, and search for the string "ODWEK multilingual."

To implement multilingual support, perform the following steps. In the following example, we will create a French language HTTP server with the CCSID 1147.

1. Create a new Apache HTTP server.
   a. Open the HTTP Admin interface at http://i_hostname.company.com:2001/HTTPAdmin, where *i_hostname.company.com* is the name of your IBM i server.
   b. Click Create HTTP Server.
   c. Specify a name and description for your new HTTP server, and click Next.
   d. Specify the root directory for the new HTTP server, and click Next.
   e. Specify the document root directory for the new HTTP server, and click Next.
   f. Specify the port number on which the new HTTP server will listen. (This port must not be used by other TCP/IP services on your system.) Click Next.
   g. Specify if you want to keep an access log.
   h. Specify how many days you want to keep log files.
   i. Review your settings, and click Finish to create your new HTTP server. (If you want to change any of the settings, click Back to return to the previous screen.)
2. Run the program to create the ODWEK directory:

   ```
   CALL PGM(QRDARS/QRLMINSTW) PARM('ccsid')
   ```

   For example, the command:

   ```
   CALL PGM(QRDARS/QRLMINSTW) PARM('1147')
   ```

   creates the directory /QIBM/UserData/OnDemand/www/1147. This directory will contain the following items:

   - cache
   - logs

- tmp
- applets
- images
- samples
- arswww.ini

3. Update the httpd.conf file for your new HTTP server to:
   a. Have the same CCSID as the OnDemand instance that will be accessed. (For example, if the OnDemand instance uses the CCSID 1147, the DefaultFsCCSID in the httpd.conf file should be 1147.)
   b. Make other updates to the httpd.conf file as required for ODWEK.

   In the following example the CCSID of the HTTP server is 1147. The HTTP server root directory is /www/odfra. Lines added for ODWEK support are delimited with comments. Text highlighted in ***bold italic*** must be changed to match your environment.

```
# Configuration originally created by Create HTTP Server wizard on Thu Oct 26 09:31:27 EDT 2006
Listen *:2114
DocumentRoot /www/odfra/htdocs
Options -ExecCGI -FollowSymLinks -SymLinksIfOwnerMatch -Includes -IncludesNoExec -Indexes -MultiViews
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{Cookie}n \"%r\" %t" cookie
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined
LogMaint logs/access_log 7 0
LogMaint logs/error_log 7 0
# Added Lines for ODWEK
DefaultFsCCSID 1147
DefaultNetCCSID 1208
CGIConvMode EBCDIC
# End Added Lines for ODWEK
SetEnvIf "User-Agent" "Mozilla/2" nokeepalive
SetEnvIf "User-Agent" "JDK/1\.0" force-response-1.0
SetEnvIf "User-Agent" "Java/1\.0" force-response-1.0
SetEnvIf "User-Agent" "RealPlayer 4\.0" force-response-1.0
SetEnvIf "User-Agent" "MSIE 4\.0b2;" nokeepalive
SetEnvIf "User-Agent" "MSIE 4\.0b2;" force-response-1.0
# Added Lines for ODWEK
LimitRequestBody 102400
ServerName iseries_hostname.company.com
UseCanonicalName Off
AccessFileName .htaccess
ScriptLogLength 200
ScriptLog /www/odfra/logs/cgi_log
Alias       /logon        /QIBM/UserData/OnDemand/www/1147/samples/logon.htm
AliasMatch ^/images/(.*)$ /QIBM/UserData/OnDemand/www/1147/images/$1
AliasMatch ^/applets/com/ibm/edms/od/(.*)$  /QIBM/ProdData/OnDemand/www/applets/$1
AliasMatch ^/applets/(.*)$                  /QIBM/ProdData/OnDemand/www/applets/$1
ScriptAliasMatch ^/scripts/arswww.cgi$        /QSYS.LIB/QRDARS.LIB/ARS3WCGI.PGM
ScriptAliasMatch ^/scripts/arswww\.cgi/(.*)$  /QSYS.LIB/QRDARS.LIB/ARS3WCGI.PGM
AddType www/unknown cab
AddType www/unknown jar
# End Added Lines for ODWEK
<Directory />
Order Deny,Allow
Deny From all
# Added Lines for ODWEK
Options None
Options +ExecCGI
<LimitExcept GET HEAD OPTIONS TRACE POST PUT>
</LimitExcept>
# End Added Lines for ODWEK
</Directory>
<Directory /www/odfra/htdocs>
Order Allow,Deny
Allow From all
</Directory>
# Added Lines for ODWEK
<Directory /QIBM/ProdData/OnDemand/www/applets/>
Order allow,deny
Allow from ALL
</Directory><Directory /QIBM/ProdData/OnDemand/www/1147/samples/>
Order allow,deny
Allow from All
Options +FollowSymLinks
</Directory><Directory /QIBM/ProdData/OnDemand/www/1147/images/>
Order allow,deny
Allow from All
Options +FollowSymLinks
</Directory>
</Directory><Directory /QIBM/ProdData/OnDemand/www/1147/>
Order allow,deny
Allow from All
Options +FollowSymLinks
</Directory>
```

```
<Directory /www/odfra/>
Order allow,deny
Allow from All
</Directory>
<Directory /QSYS.LIB/QRDARS.LIB>
Order allow,deny
Allow from ALL
Options +ExecCGI
</Directory>
# End Added Lines for ODWEK
```

4. Edit the ARSWWW.INI file in the /QIBM/UserData/OnDemand/www/ccsid directory:

   ```
   edtf '/QIBM/UserData/OnDemand/www/ccsid/arswww.ini'
   ```

   where *ccsid* is the required CCSID. For example:

   ```
   edtf '/QIBM/UserData/OnDemand/www/1147/arswww.ini'
   ```

   Change the following lines as appropriate:

   ```
   TraceDir=/QIBM/UserData/OnDemand/www/ccsid/logs
   Language=lang
   CodePage=ccsid
   TemplateDir=/QIBM/UserData/OnDemand/www/ccsid/samples
   CacheDir=/QIBM/UserData/OnDemand/www/ccsid/CACHE
   ```

   For example:

   ```
   TraceDir=/QIBM/UserData/OnDemand/www/1147/logs
   Language=FRA
   CodePage=1147
   TemplateDir=/QIBM/UserData/OnDemand/www/1147/samples
   CacheDir=/QIBM/UserData/OnDemand/www/1147/CACHE
   ```

   Change the following lines to specify the instance to be accessed by this HTTP server:

   ```
   [@SRV@_QUSROND]
   HOST=Sxxxxxxx
   PROTOCOL=0
   PORT=1450
   ```

   For example:

   ```
   [@SRV@_ONDFRA]
   HOST=i_hostname.company.com
   PROTOCOL=0
   PORT=1514
   ```

   where *i_hostname.company.com* is the name of your IBM i system.
5. Modify other configuration sections of the ARSWWW.INI file, if necessary.
6. Edit the logon.htm file to specify the default instance name:

   ```
   edtf '/QIBM/UserData/OnDemand/www/ccsid/samples/logon.htm'
   ```

   For example:

   ```
   edtf '/QIBM/UserData/OnDemand/www/1147/samples/logon.htm'
   ```

   Change the following:

   ```
   <b>Server Name:</b><input type=text name=_server value=QUSROND>
   ```

   For example:

   ```
   <b>Server Name:</b><input type=text name=_server value=ONDFRA>
   ```
7. Modify other configuration sections of the logon.htm file, if necessary.
8. Start the new HTTP server.

9. Test the configuration.
   a. Open the logon page:
      `http://iseries_hostname.company.com:port_number/logon`

      For example:
      `http://rdr400m.raleigh.ibm.com:2114/logon`
   b. Logon.
   c. Open a folder.
   d. Search.
   e. View archived data as appropriate:
      - Line Data
      - AFPDS
      - Images
      - PDF

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:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 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
Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901–7829 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.

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.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks and service marks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

Adobe, Acrobat, Portable Document Format (PDF), and PostScript® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel® and Pentium® are trademarks of Intel 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.

Linux is a registered trademark of Linus Torvalds 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.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Index

Special characters

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Program Number: 5770-RD1

SC19-2791-00