IBM® WebSphere™ Host Publisher Administrator's and User's Guide

Version 2.2

GC31-8728-01

IBM® WebSphere™ Host Publisher Administrator's and User's Guide

Version 2.2

GC31-8728-01

Note

Before using this information and the product it supports, be sure to read the general information under "Appendix C. Notices" on page 119.

2nd Edition (July, 2000)

Contents

Chapter 1. About Host Publisher 1
About this information 1
What is Host Publisher? 1
  Comparing Host Publisher and WebSphere Application Server 3
  Comparing Host Publisher and IBM Host On-Demand 4
  Host Publisher's advantages 4
About Host Publisher Studio 5
About Host Publisher Server 6
What's new with Host Publisher Version 2.2? . 6
Migrating from Host Publisher Version 2.1 to Version 2.2 6
For more information 6
  Information on the Web 9

Chapter 2. Administering Host Publisher 11
Using the Server 11
  Using Host Publisher Server Administration 11
  Deploying applications 11
  Managing licenses 12
  Logging and tracing components 13
  Balancing the server's load 16
  Application administration 17
  Setting up the Display Terminal 18
Securing access to Host Publisher Administration using WebSphere Application Server 20
Using the XML Legacy Gateway 21
  hPubPortalAdmin 22
  Host Publisher XLGW portal page . . . 24
  Enhancing the xmlLegacyGateway servlet 25
  Tracing the xmlLegacyGateway servlet . . 25
Configuring WebSphere Application Server and your Web server for Host Publisher Server 26
  Web Server Configuration 26
  WebSphere Application Server Configuration 27

Chapter 3. Using Host Publisher 29
Overview 29
Using Host Publisher Studio 30
  About macros 30
  Creating an Integration Object for host access 31
  Creating user lists 42
  Creating an Integration Object for database access 43
  Creating Web pages for Integration Objects 45
  Creating composite applications 49
  Importing Java objects 53
  Transferring applications to a Host Publisher Server 53
  Enabling tracing 55
  Remote Integration Objects 55
Using WebSphere Studio 56
  Using WebSphere Page Designer 56
  Inserting a Host Publisher Integration Object into a WebSphere Studio Project . . 56
  Inserting a Host Publisher .jsp into a WebSphere Studio project 56
  Transferring .jar and .jsp files to the Host Publisher Server 56

Chapter 4. Advanced features 59
Using Integration Object chaining 59
Configuring and using Secure Sockets Layer (SSL) support 62

Chapter 5. Troubleshooting 65
Host Publisher problem determination procedure 65
Common problems and limitations 66
  With Host Access and execution of Host Access Integration Objects 66
  With Database Access and execution of database Integration Objects 68
  General problems and limitations 69
  With the Studio and transferring applications 70
  With the Server and execution of Integration Objects 70
Gathering WebSphere Application Server and Web server problem data. 76
  Standard output and standard error output logs 76
  Error logs redirected from Web servers . . 77

Error logs for problems and events discovered by WebSphere Application Server 78
Contacting IBM for service 78

Appendix A. Server error messages and recovery actions 81

Appendix B. Examples for designing custom Web pages. 109
Java access to page parameters 109
Redirecting based on Integration Object results 109
Invoking Integration Objects based on previous Integration Object results 110
Building dynamic HTML based on Integration Object properties 110
Validating user input. 111
Testing for successful database record deletion 111
Testing for successful database record addition 112
Passing Java variables to JavaScript function 112
Using Java to display variables passed into a page 113
Using Java to pad an input value and passing it to an Integration Object 113
Using a function type passed from a hidden HTML form variable to determine page to execute 113
Using Java to prevent blank lines in an HTML table. 114
Using Java to control display of HTML table based on host results. 115
Determining number of page downs and tabs for making a selection. 116
Changing the action value of a form based on the clicked button. 117
Using HTTP Session object to pass values 117
Disabling the browser back button 117

Appendix C. Notices 119
Programming interface information 121

Appendix D. Trademarks. 123

Index 125

Chapter 1. About Host Publisher

About this information

This information is designed to help you, the Web Application Developer, plan for, install, and start using IBM WebSphere Host Publisher. This book includes information about how to record interactions with data sources within an Integration Object, how to include those Integration Objects in Web pages, and how to publish those pages on the Web.

This book helps you get started quickly. Additional information resources are available for learning to use Host Publisher features. These resources include the product README, online help, and product Web pages. (See “For more information” on page 6).

Note: Consult the product README and the Host Publisher Web site, http://www.ibm.com/software/network/hostpublisher, for corrections and additions to this information.

This book is available in hard copy (in the Host Publisher packaging), as an HTML file on the installation CD, as a PDF file on the CD, and as an HTML file on the product Web site. Visit the Web site for the most updated version of this document.

What is Host Publisher?

Web-to-host integration is an integral part of any e-business. 70% of business-critical data and applications reside on IBM host systems, such as S/390, AS/400, and RS/6000. Making this information available to new users and using it in new ways across intranets, extranets and the Internet enables you to reduce costs, improve services, generate new sources of revenue, and establish a competitive advantage.

Host Publisher is a set of tools that enable you to provide access to data on a legacy data source (a host machine with a 3270 or database application, for example) on the World Wide Web or a private intranet. These legacy data sources typically involve proprietary access and complex applications. Host Publisher enables you to present end users with the data they need without exposing the way the data is accessed.

Host Publisher enables you to create *Integration Objects* that contain logic to perform host data access and retrieval tasks. You can use and reuse these Integration Objects in other Java applications, or you can publish the

Integration Objects on the Web. When you publish Integration Objects on the Web, you enable other people to interact with the data. For example, if you have an existing host application that enables you to look up telephone numbers for employees, you can create a Web page that lets a user with a browser enter the name of the person they want, and then displays the telephone number on another page.

Host Publisher enables you to create Integration Objects for common tasks, such as connecting to a host, and reuse them. You can chain Integration Objects together so that the user can log in, complete several tasks, and then log off. You can also create composite applications that integrate multiple sources of data into a single Web page. To the end user, this Web page appears to be a single new application.

Host Publisher uses IBM WebSphere Application Server to provide a consistent, reliable deployment environment for Java applications and HTML/JSP pages across platforms. Applications created in Host Publisher Studio can be accessed with all standard Web browsers, with or without Java.

Host Publisher supports terminal-oriented applications that use 3270, 5250, and VT data streams, as well as databases that provide a JDBC interface, such as IBM DB2 Universal Database®, Oracle®, and Sybase™.

Host Publisher provides the enterprise-class features you expect, including security, load balancing and hot standby. Host Publisher supports Secure Sockets Layer (SSL) encryption and authentication, as well as DES-encrypted passwords, to provide a high level of security. Host Publisher includes the IBM Network Dispatcher, which provides for large enterprises a high level of performance and availability through load balancing and hot standby.

Host Publisher is divided into two major components: Host Publisher Studio and Host Publisher Server. The Studio provides the development environment for creating Web applications. The Server provides the runtime environment for executing Web applications created with the Studio. You create Web-to-host applications using the Studio, publish them to the Server, and provide access to the end user. The Web-to-host applications you build contain Integration Objects. When used with the Server, Integration Objects can:

- Automatically establish a connection with a host
- Navigate to and extract data from an application
- Disconnect from the host and end the connection

You can use Integration Objects as part of fully-customizable Web pages or reuse them with other Java application programs.

Host Publisher provides Integration Object chaining, which involves constructing a set of Integration Objects that depend on each other to drive a connection through several states. Chaining can increase performance and reduce the administration of creating complex applications. For example, you might use chaining in a typical 3270 application that uses multi-level menus. A corporate phone directory might have several menus to step you down to the point where you can list everyone in a particular department. You want to display the office address for an individual, return to the department list and select a new name, and display the second person's office address. Chaining enables you to break the task into steps, each of which will be represented as a single Integration Object, so that the end user does not have to navigate back down through the several menus to reach the department list again.

You can optimize connection establishment for each Integration Object by using connection pooling. Connection pools are defined in the Studio. Connection pools can be used during runtime to cache ready connections to improve response time to Web pages. A user-defined number of connections can remain active in the pool for subsequent requests from any user. Connection pools can eliminate the overhead of establishing new connections to the host or database for every Web page request.

Comparing Host Publisher and WebSphere Application Server

Although Host Publisher and WebSphere Application Server complement each other very well, and Host Publisher integrates WebSphere into its runtime environment, there is a fundamental difference between the primary use of each product.

Host Publisher delivers a quick and easy way for companies to implement e-business applications by extending *existing* applications to the Internet. It focuses on applications with little or no new business logic. In contrast, WebSphere Application Server provides a robust Java infrastructure for the development and execution of Web applications and Servlets. WebSphere Application Server focuses on adding new business logic to existing applications to make them accessible on the Web or deploying totally new Web applications such as those used in business reengineering.

The two products are complementary. Host Publisher uses the WebSphere application server environment to support applications that include Integration Objects created by Host Publisher. You can reuse Integration Objects within new Java applications, or you can use WebSphere Application Server and your favorite Java interactive development environment (IDE), such as Visual Age for Java, to add new logic to Host Publisher Web applications. For information about using Integration Objects in Java servlets and applications, see the Host Publisher Web page: http://www.ibm.com/software/network/hostpublisher/library.

Host Publisher provides WebSphere Application Server Standard Edition. However, if you need or already use the advanced features of WebSphere Application Server Advanced Edition or WebSphere Application Server Enterprise Edition, you can substitute those products to support the Host Publisher runtime environment.

Comparing Host Publisher and IBM Host On-Demand

Host Publisher and Host On-Demand are designed for different end users. Host On-Demand is intended for users who are already familiar with host applications. Host Publisher is intended for users who are not.

Host Publisher is designed primarily for Internet users who are not familiar with typical host screens or how to navigate through legacy applications, and for whom a new, easy-to-use graphical interface is critical. These users might not have Java-enabled browsers and therefore require HTML. As with Web self-service applications, these users typically connect infrequently and for short periods of time. They are familiar with their standard HTML browser, and they are accustomed to Web response times. Applications for these users might need to access multiple hosts. Host Publisher can also be appropriate for extranet users and, to a lesser extent, intranet users where their usage and requirements are similar to the Internet user.

Host On-Demand is IBM’s answer for Java-based host access primarily designed to meet the needs of intranet and extranet users. These users are familiar with the original host application screens and can be considered power users who require a full function emulator. User desktop software is typically well controlled and can include a Java-enabled browser. Users typically connect for extended periods of time, and fast response times are important to maximize productivity.

Host Publisher’s advantages

Host Publisher is built on open-industry standards, such as Java and HTML. Integration Objects are reusable components that can be used in Java applications created outside Host Publisher. Likewise, interactive development environment (IDE) tools can be used to add new business logic to the applications Host Publisher creates. The Studio also generates fully customizable HTML output with embedded JavaServer Pages tags. You can use any HTML editor to enhance and customize the HTML to meet your design guidelines and personal preferences.

Host Publisher Server provides enterprise-class performance, scalability and availability through several key features, such as chaining, connection pooling, load balancing, hot standby, and cross-platform portability. Since the Host Publisher Server runs on OS/390™, AIX™, Windows NT, Solaris, OS/400, and NetWare applications created with the common Host Publisher Studio will run unchanged in all environments.

Host Publisher's use of Network Dispatcher's load-balancing capabilities enable you to balance the load of Integration Object requests over a group of Host Publisher Servers. This provides predictable performance, easy scalability, and hot backup. The ability to move from one operating system platform to another will allow you to move your workload to a higher capacity platform as demands increase.

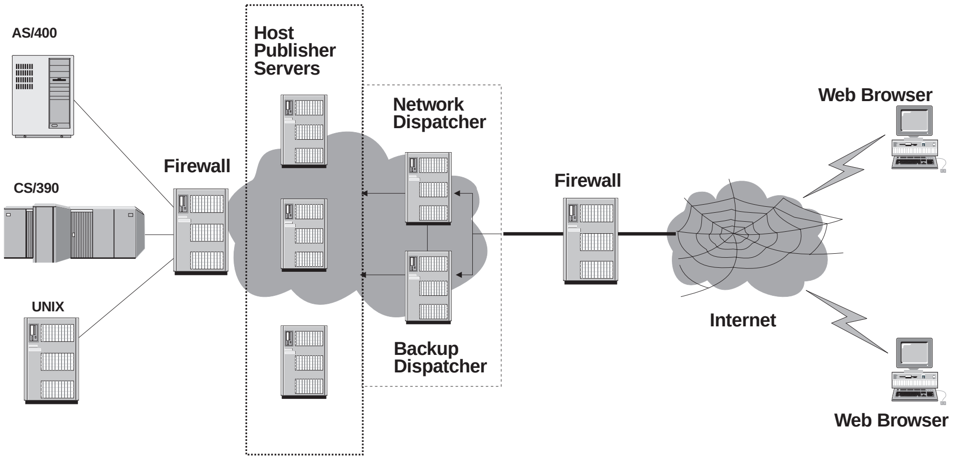

Figure 1. Host Publisher working in a network

About Host Publisher Studio

Host Publisher Studio is a collection of task-oriented, easy-to-use graphical user interfaces that assist the Web application builder in managing and creating Web-to-host publishing applications. It uses task-oriented prompts to guide the user through the creation process— including recording host and database interactions, identifying desired data, and labeling that data for retrieval. Host Publisher Studio automatically generates JavaBeans called Integration Objects, which encapsulate the interactions and data retrieval logic. You can use Host Publisher Studio to generate a fully customizable Web page for modeling interaction with the Integration Objects and rendering the resulting data. You can enhance the generated Web page with your favorite Web authoring tool, such as WebSphere's page designer, to meet corporate guidelines on style and image. Once the Web page is completed, you publish it to a Host Publisher Server for production access by end users.

Host Publisher Studio runs on the Windows 95®, Windows 98, Windows NT, and Windows 2000 operating systems.

About Host Publisher Server

Host Publisher Server provides the run-time environment for supporting Web applications created with Host Publisher Studio. It consists of the IBM WebSphere Application Server and other run-time components such as connection management, license monitoring, run-time administration, and log and trace management.

Host Publisher Server is supported on OS/390, OS/400, AIX, Windows NT, NetWare, and Sun Solaris operating environments. For information about supported Web servers, refer to the *Host Publisher Planning and Installation Guide* for your platform.

What's new with Host Publisher Version 2.2?

Host Publisher Version 2.2 runs on more platforms and provides some new features that broaden its function. New features include:

- Support for the NetWare and Windows 2000 operating systems (in addition to OS/390, OS/400, AIX, Solaris and Windows NT)
- The XML Legacy Gateway, which converts a 3270 or 5250 data stream into an XML format; this format is then available to Java applications or can be automatically displayed as HTML in a browser
- Remote Integration Objects (RIO), which provides a standard XML interface to access data from Integration Objects and enables Java applications or applets running on a remote system to execute Integration Objects as if they were local

Migrating from Host Publisher Version 2.1 to Version 2.2

When migrating from Host Publisher Version 2.1 to Version 2.2, you must install Version 2.2 in the same directory path to ensure access to your existing applications. You can migrate from Host Publisher 2.1 to 2.2 without taking any other specific actions. It is not, however, backwards compatible; that is Integration Objects created with Host Publisher Studio Version 2.2 will not run on a Version 2.1 server.

For more information

To access online documentation installed with Host Publisheruse a Web browser to open the following HTML files on your local system.

For the *IBM WebSphere Host Publisher Administrator's and User's Guide*

AIX	/usr/lpp/HostPublisher/common/doc/*lang*/guide.htm
NetWare	//*ServerName*/_IBM_HP_doc/guide/guide.htm where *ServerName* is the name of your server

OS/400	/QIBM/ProdData/Hostpublisher/doc/guide/guide.htm
Solaris	/opt/HostPublisher/common/doc/*lang*/guide.htm
S/390	/usr/lpp/HostPublisher/common/doc/*lang*/guide.htm
Windows NT	*install_dir*\HostPub\common\doc\guide\guide.htm where *install_dir* is the directory in which Host Publisher is installed.
Windows 2000	*install_dir*\Common\doc\guide\guide.htm where *install_dir* is the directory in which Host Publisher is installed.

For the *IBM WebSphere Host Publisher Planning and Installation Guide*

AIX	/usr/lpp/HostPublisher/common/doc/*lang*/instgd.htm
NetWare	//*ServerName*/_IBM_HP_doc/install/nwhpinst.htm where *ServerName* is the name of your server
OS/400	/QIBM/ProdData/Hostpublisher/doc/install/as4inst.htm
Solaris	/opt/HostPublisher/common/doc/*lang*/instgd.htm
S/390	/usr/lpp/HostPublisher/common/doc/*lang*/instgd.htm
Windows NT	*install_dir*\HostPub\common\doc\install\instgd.htm where *install_dir* is the directory in which Host Publisher is installed.
Windows 2000	*install_dir*\Common\doc\install\instgd.htm where *install_dir* is the directory in which Host Publisher is installed.

For the *IBM WebSphere Host Publisher Programmer's Guide and Reference*

AIX	/usr/lpp/HostPublisher/common/doc/*lang*/proggd.htm
NetWare	//*ServerName*/_IBM_HP_doc/proggd/proggd.htm where *ServerName* is the name of your server
OS/400	/QIBM/ProdData/Hostpublisher/doc/proggd/proggd.htm
Solaris	/opt/HostPublisher/common/doc/*lang*/proggd.htm
S/390	/usr/lpp/HostPublisher/common/doc/*lang*/proggd.htm
Windows NT	*install_dir*\HostPub\common\doc\proggd\proggd.htm

where *install_dir* is the directory in which Host Publisher is installed.

Windows 2000
install_dir\Common\doc\proggd\proggd.htm

where *install_dir* is the directory in which Host Publisher is installed.

For the IBM Host Publisher Readme

AIX	/usr/lpp/HostPublisher/common/doc/*lang*/readme.htm
NetWare	//*ServerName*/_IBM_HP_doc/readme.htm where *ServerName* is the name of your server
OS/400	/QIBM/ProdData/Hostpublisher/doc/readme.htm
Solaris	/opt/HostPublisher/common/doc/*lang*/readme.htm
S/390	/usr/lpp/HostPublisher/common/doc/*lang*/readme.htm
Windows NT	*install_dir*\HostPub\common\doc\readme.htm where *install_dir* is the directory in which Host Publisher is installed.
Windows 2000	*install_dir*\Common\doc\readme.htm where *install_dir* is the directory in which Host Publisher is installed.

lang is the language-specific subdirectory for your language, and is one of the following:

de_DE	German
en_US	English
es_ES	Spanish
fr_FR	French
it_IT	Italian
ja_JP	Japanese
ko_KO	Korean
pt_BR	Brazilian Portuguese
tr_TR	Turkish
zh_CN	Simplified Chinese
zh_TW	Traditional Chinese

To open the book (PDF) versions, substitute the following file names for the .htm file names:

Administrator's and User's Guide
guide.pdf

Planning and Installation Guide
instgd.pdf

Programmer's Guide and Reference
progguid.pdf

Online help, including the HTML version of this book, is available from the product's user interface.

Information on the Web

Find the most up-to-date versions of this document, frequently asked questions (FAQs), white papers, and additional information at the product Web site:

- http://www.ibm.com/software/network/hostpublisher

Chapter 2. Administering Host Publisher

Using the Server

Once you have published pages to the Host Publisher Server, the server administrator can use the administration interface to manage the way connections and applications are handled.

This section contains information to help you with the following tasks:

- Using Host Publisher Server Administration (“Using Host Publisher Server Administration”)
- Deploying applications (“Deploying applications”)
- Managing licenses (“Managing licenses” on page 12)
- Logging and tracing components (“Logging and tracing components” on page 13)
- Balancing the server’s load (“Balancing the server’s load” on page 16)
- Application administration (“Application administration” on page 17)
- Setting up the Display Terminal for AS/400 (“Setting up the Display Terminal” on page 18)

Using Host Publisher Server Administration

The Host Publisher Server provides the runtime environment for supporting Web applications created with the Host Publisher Studio. The runtime environment includes Web-based server administration for controlling the runtime and the applications it serves. To access Server Administration, load this URL in your browser: **http://***server***/***hp_alias***/HostPublisher/HPAdmin/main.jsp** (where *server* is your server name and ***hp_alias*** is the alias chosen during installation of Host Publisher Server).

Deploying applications

Host Publisher deploys applications that have been transferred to the server into production.

By default, applications must be deployed manually. If you want an application automatically deployed when the server starts, you must change the server.properties file. See the *IBM WebSphere Host Publisher Programmer’s Guide and Reference* for instructions on editing the properties file.

To deploy an application:

1. Start Host Publisher Server Administration.

2. In the navigation area on the left, click Administration > Application Administration > Deploy.
3. Check the boxes next to the applications you want to deploy.
4. Click Deploy.

Once you choose the applications you want to deploy, Host Publisher Server scans the staging area for the application files and, if no problems are found, moves the files to the production area for immediate use. The staging area is then cleared.

Some examples of problems are: errors in connection pool configuration files, missing files, conflicting information, or applications already deployed.

If one or more of the applications selected for deployment contain strongly-encrypted user pool data, you must enter the Host Publisher encryption key in the entry field provided. If you do not, only selected applications that do *not* contain strongly-encrypted user pool data are deployed; however, you will receive a message informing you that one or more applications were not deployed because they require the encryption key.

Note: The Host Publisher encryption key is defined during the application creation process in the Host Publisher Studio, and allows Host Publisher Server to encrypt and decrypt data.

Notes:

1. If a previous version of an application, or another application with the same name, is currently deployed, deploying a new version may overwrite existing files. Next to each application there is a column that indicates whether an application of the same name is currently deployed.
2. If you deploy an application that has already been deployed, WebSphere Application Server will stop and restart all servlets. This might cause a disruption in service.
3. WebSphere Application Server for OS/390 does not support the concept of a reloadable classpath. When you deploy an application on Host Publisher for OS/390, you must manually add the Integration Object JAR file names to the java.classpath statement within WebSphere Application Server's jvm.properties file, then restart WebSphere Application Server. This file is located in the /usr/lpp/WebSphere/AppServer/properties/server/servlet/servletservice directory.

Managing licenses

The Host Publisher Server tracks the number of connections established to data sources and automatically logs a message when the value exceeds the number of licenses purchased. One license is equivalent to the right to use one Host Publisher connection at a time.

Host Publisher Server can optionally track license usage history over time. This history maintains the maximum number of licenses used (or connections established at one time) during a one-hour period, logging this information to a file each hour. By default, this option is disabled.

To enable the license tracking option, set the licenseTracking property in the server.properties file from 0 to 1. This file is located in the Server subdirectory under your Host Publisher installation directory.

For detailed information about editing server.properties, see the *IBM WebSphere Host Publisher Programmer's Guide and Reference*.

If you purchase more licenses and want to change your current value, you can do so using Host Publisher Server Administration.

1. Start Host Publisher Server Administration.
2. In the navigation area on the left, click Administration > License Management.
3. Type the new value, then click Set.

Logging and tracing components

The Host Publisher Server produces messages.txt and trace.txt files to help you in problem determination.

The log file (messages.txt)

The Host Publisher log file records information about three kinds of events:

Error events
Problems that prevent an operation from completing. Error events usually require that you take some action to correct the problem.

Warning events
Unexpected occurrences that may require action to correct the problem. Warning events are not as serious as error events.

Information events
Normal occurrences, such as starting and stopping the Host Publisher Server. Information events do not require any action.

The log file is intended for you to read and use as a reference when troubleshooting problems. Most of the messages that appear in the log are documented in this book and contain suggestions for actions you can take to correct problems, when necessary. (See“Appendix A. Server error messages and recovery actions” on page 81.)

The trace file (trace.txt)

The trace file records details of the internal operation of the Host Publisher Server, and is not necessarily intended for you to read. Typically, the trace facility is used when requested by IBM service.

Turning on tracing adversely affects the performance of Host Publisher Server.

Controlling logging and tracing

The Problem Determination panels in Host Publisher Server Administration enable you to control tracing and logging. Using these panels, you can perform the following tasks:

View Log
View the last 200 lines of the log file, download the entire file, or clear the file.

Note: If you click Clear, all the current information is deleted from the log file.

Set Log Options
Control whether information and warning events are written to the log file, and define the file to which the log events are written. If the file already exists, new events are appended to it. Error events are always written to the log file.

View Trace
View the last 200 lines of the trace file, download the entire file, or clear the file.

Note: If you click Clear, all the current information is deleted from the trace file.

Trace File Name
Define the file to which trace events are written. If the file already exists, new events are appended to it.

Host Connection Tracing
Control the tracing levels of None, Minimum, Normal, or Maximum, and select one of three types of trace for host connections:

Macro tracing
Traces the playing of macros

Presentation space tracing
Traces the internal display of the terminal screen

Transport tracing
Traces the network connection

Control whether terminal screens are created on the Server display (except OS/390)

Display screens are created only for connections established while this option is turned on. They are not created for existing connections, or for connections that are idle.

Notes:

1. Display Terminal is not currently supported on OS/390.
2. To use Display Terminal on OS/400, Remote Abstract Windowing Toolkit (RAWT) must be enabled.

CAUTION:
Turning on the Display Terminal option can seriously affect performance or overload the Server. Do not use this on servers with many connections. Display Terminal is intended for use in debugging during application development on a test system; it is not intended for use on a heavily-loaded production server.

Database (JDBC) tracing

Turn on database tracing. This option enables tracing of all Java Database Connectivity (JDBC) activity in WebSphere Application Server and directs the output to the Host Publisher Server trace file.

Note: All JDBC activity, not just Host Publisher's, is traced while this option is enabled. It is possible for another application to also enable JDBC tracing and redirect the output to another location. This would include Host Publisher trace data.

Server tracing

Trace events on the Server. This panel contains two groups of options:

Trace Types

API Tracing

Traces calls in and out of other components, such as Host On-Demand and JDBC.

Entry/exit Tracing

Traces internal calls within the Server. This is where most tracing occurs.

Miscellaneous Tracing

Traces events not covered by the other two trace types.

Trace Sources

Server Tracing
: Controls tracing of the core of the Server. This is the part the manages connections, administration, and so forth.

Integration Object Tracing
: Enables tracing in specific Integration Objects that you have built.

Note: You **must** select one or more options from each group to enable any Server tracing. For example, to enable all trace types in Host Publisher Server, select API, Entry/Exit, and Miscellaneous trace types, then select Server trace source.

Multiple log and trace files

By default, the size of the log and trace files is 512 KB, and two files are saved. If you prefer to work with a different number of log or trace files, or if you want to change the size of the files, you can edit the server.properties file, located in the Server subdirectory where Host Publisher is installed. The keywords and their default values are:

```
maxLogFile=2
maxLogFileSize=512k
maxTraceFile=2
maxTraceFileSize=512k
```

For detailed information about editing server.properties, and for information about enabling multiple log or trace files, see the *IBM WebSphere Host Publisher Programmer’s Guide and Reference*.

Balancing the server’s load

Host Publisher Version 2.2 includes IBM Network Dispatcher, which enables you to balance client request loads across multiple Host Publisher Servers. Network Dispatcher forwards page requests from clients to available Host Publisher Servers in a round-robin fashion, meaning that each Host Publisher Server is dispatched a client request in turn. Clients include the Network Dispatcher’s host name in the URL request for the initial Web application page. Network Dispatcher then replaces its own host name with the appropriate destination Host Publisher Server’s host name in the URL and forwards the request to that server.

Since most Web applications include more than one page and at least one Integration Object, it is important that Network Dispatcher ensure a particular client’s ability to access the same Host Publisher Server for a period of time. This is because Web applications can introduce the concept of a *state*, where a second page assumes specific information was entered on the first page. You would probably not want to be routed to Host Publisher Server A for the first page request, then to server B for the second page request; server B does not know what Server A provided on the page.

Maintaining the relationship between client and server for the duration of the Web application is called *session affinity*. Network Dispatcher achieves session affinity by enabling you to set *port stickytime*. This is a period of time in which a client is always routed to the same server. The timer starts on the first request and restarts on each subsequent request. After the timer expires (no request is received from the client during that time), the next client request will not necessarily be routed to the same server.

When using Network Dispatcher to balance request loads, you probably want to complete the Web application and not allow a long period of time to pass between page requests to avoid disruption of the application, resulting in lost information. Therefore, the administrator should configure port stickytime value large enough to allow for long Web applications or possible pauses or delays between page requests.

IBM Network Dispatcher does not necessarily require an additional server machine; however, we recommend that Network Dispatcher not be installed on the same machine as Host Publisher Server, as it will require resources that might better serve Host Publisher, such as memory and processor time.

For instructions on installing and setting up Network Dispatcher, refer to the User's Guide on the IBM Network Dispatcher CD in the documentation\LANG\\htm\\ directory (where *LANG* is the appropriate language). Refer to the section on **Server Directed Affinity API** for information on how to set the appropriate port stickytime for session affinity.

Application administration

Open Host Publisher Server Administration and click Administration > Application Administration to:

List Applications
View a list of all the applications on the Server, both in the staging and production areas. These applications are identified as Deployed and Not Deployed, and the list is sorted by application name, regardless of its status. An application can appear as both Deployed and Not Deployed if a new version of a deployed application has been transferred to the Server, but not yet deployed.

Deploy Applications
View a sorted list of all applications transferred to the server and select one or more for deployment. The deployed applications move to the Server's production area and become available for use. If any of the applications you want to deploy have user pool data that require strong encryption, you must enter the Host Publisher encryption key

to deploy these applications. This key is defined in the Host Publisher Studio and allows Host Publisher Server to encrypt and decrypt data.

Changes to existing pools or connections will not take place until the Host Publisher Server restarts. When the server is running, you can deploy applications that use newly-defined connection pools, and the application will be activated immediately.

Remove Applications
View a list of all deployed applications, and select one or more of them to delete from the Server. You must stop the server before you can delete applications. If you delete an application accidentally, you cannot undo the deletion; however, if the application is still available in the Host Publisher Studio, you can publish the application again.

Delete Unused Files
View files in the production area of the Server that do not belong to any application, and select one or more of them for deletion. If these files belonged to an older version of an application, you can safely delete them. If, however, they were manually placed on the Server (not using Studio's Transfer to Server), it may not be safe to delete them. Therefore, carefully check that the files are not in use before you attempt to delete them.

You cannot restore files you delete.

Setting up the Display Terminal

Host Publisher Server Administration includes a tracing function that enables you to view the text screen (*green screen*) of any Host Access Integration Object as it runs in real time on the Server. You can enable and disable this function using the Display Terminal check box located in Server Administration, at Problem Determination > Set Trace Options > Host Connection Tracing. Whenever Display Terminal check box is changed, you must restart Host Publisher Server to ensure that the new setting is read. The default is Disabled. This check box is not normally displayed in the AS/400 version of Host Publisher Server Administration.

To display and use the check box, some additional configuration is required on the AS/400 WebSphere Application Server and on your choice of a secondary network system that is capable of supporting Java Abstract Windowing Toolkit (AWT) JDK classes. This is necessary because the AS/400 Server does not support graphical display devices as direct connect devices; it does support them as client and server roles. Java has a graphical solution for this called Remote AWT. In this case, another system, perhaps Windows NT/9x, acting as an AWT server, handles the graphical display of the terminal screen and interaction of Java code running in the primary environment; specifically, Host Publisher Server for AS/400. It works like this:

1. Configure your Windows NT/9x system to run the Java JDK Remote AWT server/listener. To do this:
   a. Copy a Java .jar file to your Windows system.
   b. Run a class to start the AWT server/listener. The IP address of this system displays in a confirmation window while the system is ready to receive and run AWT objects from any client.

      Note: To get the filenames, locations, and specifics, go to the AS/400 Information Center Web site (http://publib.boulder.ibm.com/html/as400/infocenter.html), click Go, then select: Java > AS/400 Developer Kit for Java > Running on a Host without a GUI > Remote AWT.
2. Configure your AS/400 WebSphere Application Server JVM, which also runs Host Publisher Server, to send all AWT requests and functions to the Windows server/listener setup in step b.

   This involves setting two JVM system properties:

   ```
   os400.class.path.rawt=1
   RmtAwtServer=<ip address of server/listener in step 1>
   ```

   To include this in the WebSphere Application Server JVM, put these properties in the SystemDefault.properties file, located in the home directory of the user ID authenticating the JVM. In the AS/400 WebSphere Application Server case, that user ID is QEJBSVR; normally the home directory is /home/QEJBSVR. You may need to create the directory if it does not exist, then create the SystemDefault.properties stream file with the property keys and values named above. The file should be encoded with code page 37, 437, or 819. You can use the CPY OS/400 command to translate the file, if necessary. Refer to the AS/400 Developer Kit for Java Web site (http://publib.boulder.ibm.com/pubs/html/as400/v4r4/ic2924/info/java/rzaha/d for additional system property or Remote AWT detail.
3. When setup and configuration is complete, use OS/400 commands ENDSBS QEJBSBS and STRSBS QEJB/QEJBSBS to stop and start WebSphere Application Server. This will enable Java's Remote AWT, which is required because SystemDefault.properties is read-only at JVM startup.

   Host Publisher Server Administration detects if Remote AWT is enabled and, if it is, displays the Display Terminal check box located in Server Administration at Problem Determination > Set Trace Options > Host Connection Tracing.
4. Select this check box and submit changes.
5. Click Server Status in Host Publisher Server Administration, then restart the Host Publisher Server to enable Display Terminal support. Whenever Display Terminal check box is changed, you must restart Host Publisher Server to ensure that the new setting is read.

Some tips and a word of caution

Tip: First test the Remote AWT setup and configuration with your AS/400 user ID. Put a copy of the SystemDefault.properties file in your AS/400 home directory and run a Java class that uses a simple AWT object. This test is independent of WebSphere Application Server and Host Publisher Server.

Caution: Do not disable Remote AWT without first clearing the Display Terminal check box and submitting changes in HPAdmin. If you do not follow this sequence, a Host Publisher Server 500 Internal Server Error will occur every time a Host Access Integration Object is served. The correct sequence is to clear the Display Terminal check box and submit changes, then disable Remote AWT support in the WebSphere Application Server JVM.

Tip: A simple way to disable Remote AWT on AS/400 WebSphere Application Server is to rename /home/QTMHHTTP/SystemDefault.properties, then stop and start WebSphere Application Server.

Securing access to Host Publisher Administration using WebSphere Application Server

You can control access to your system resources by managing users and groups and associated permissions.

To enable WebSphere Application Server 2.0.3 user ID and password security:

1. Load the WebSphere Application Server in your browser (http://*yourlocalhost*:9527).
2. Click the triangle next to **Security** in the navigation frame.
3. Click Users to add and delete users and control access permissions.
   a. To add a user, click Add, then type the user name and associated password in the Add User dialog box.
   b. To delete a user, highlight the name in the Defined Users list, then click Remove.
   c. To change a password, highlight the associated user name, then click Password. Type the new password in the Change Password dialog box.
4. To efficiently manage a large number of users, create a group. Click Groups in the navigation frame.
   a. To add a group, click Add, then type the name of the group in the Add Group dialog box.
   b. To add users to a group, highlight the user name in the Non-Members list, then click Add.
   c. To remove users from a group, highlight the user name in the Members list, then click Remove.

   d. To remove a group, highlight the group name in the Defined Groups list, then click Remove.

5. To control who can access your Web server resources, click Access Control Lists in the navigation frame
   a. To add an access control list (ACL) to the defaultRealm, click Add, then type the new ACL name in the ACL dialog box and click Add.

      Note: The default realm setting is **defaultRealm**.
   b. Highlight the new ACL in the Defined Access Control Lists, then click Add next to the bottom box.
   c. Make sure **Assign Permission for:** is set on **Files and Folders**.
   d. Highlight your user group.
   e. In the check boxes at the bottom of the box, check Get, Put, and Post.
   f. Click OK.
6. To protect resources by establishing an access control list for each resource, click Resources in the navigation frame.
   a. To add a resource to the defaultRealm, click Add. (The default realm setting is **defaultRealm**.)
   b. In the Protect a Resource dialog box:
      1) Make sure Scheme is set to Basic.
      2) Select the appropriate ACL.
      3) Click the Servlet option, then select HPAdmin from the pull-down list.
      4) Click OK.

After you complete these steps and load http://*yourlocalhost*/HPAdmin/main.jsp in your browser, a logon prompt will appear. To run the servlet, type the user ID and password that you created using the previous procedure.

Using the XML Legacy Gateway

The XML Legacy Gateway allows program access to 3270 and 5250 application data in an XML format. it also provides an HTML emulator for end-user access to 3270 and 5250 applications. The XML Legacy Gateway supplies a portal to the emulator function, which relies on the xmlLegacyGateway servlet for interaction with the host applications.

The XML Legacy Gateway (XLGW) portal consists of:
- **hPubPortalAdmin**, an administrative servlet
- **hPubPortalData.xml**, the XML data file that hPubPortalAdmin creates

- **hPubPortal**, the portal servlet that enables access to host links created through the administrative servlet.
- **xmlLegacyGateway servlet**, Web-based terminal emulator that enables interaction with host terminal applications using XML data formats

hPubPortalAdmin

The Host Publisher XLGW administrative servlet, hPubPortalAdmin, enables you, the Host Publisher administrator, to:

- Access hosts and host applications
- Create a portal page for accessing hosts and host applications

To use hPubPortalAdmin:

1. Install Host Publisher.
2. Access hPubPortalAdmin, as a servlet, with these URLs:

   Running on WebSphere Application Server 2.0.3:
   http://*servername*/servlet/xmlLegacyPortal.hPubPortalAdmin

   Running on WebSphere Application Server 3.0.2:
   http://*servername*/_IBM_HP_WebAdmin_/hPubPortalAdmin

   where *servername* is the name of the Web server where Host Publisher is installed.
3. On the Host Publisher: Legacy Gateway Administration page, do one of the following:
   - Click **Create New Session** to create a new XLGW host connection definition, also called a *session*.
   - Select a session name from the list box, then click **Start**, to start an XLGW host connection and work with the selected session

     To change the appearance of the session list:
     - Click **Move Up** to move the selected session one space up in the list.
     - Click **Move Down** to move the selected session down one space in the list.
     - Click **Delete** to delete the selected session from the list.
     - Click **Edit** to edit the session entry in the list.
4. On the XML Legacy Transformation page, complete the fields as follows. After you define the necessary parameters, you can access a host using the xmlLegacyGateway servlet and xmlAppData JavaBean.

   Session Name
   The name of your choice for this session

   Link Name
   The link name that will appear on the Host Publisher portal page for this session

Link Description
The description of the link that will appear on the Host Publisher portal page

Host Name
The TCP/IP address or the name of the 3270 or 5250 host system

Port The port number of the 3270 or 5250 application

Host Type
The type of host session you want to access; typically, this is 3270 or 5250

Host Code Page
The language that the host system will use to communicate with the Host Publisher XLGW servlets

Display Size
The size of the host screen in rows and columns

- Click **Start** to start the XLGW host connection. Verify that the host session starts properly in a new instance of your Web browser before saving this configuration.
- Click **Save** to save this configuration to the hPubPortalData.xml file, and then return to the Legacy Gateway Administration page.
- Click **Cancel** to return to the Legacy Gateway Administration page without saving your configuration. You can then start a new configuration definition, or choose to work with an existing session definition.

Note: For more information on the xmlLegacyGateway servlet and the xmlAppData JavaBean, see the *IBM WebSphere Host Publisher Programmer's Guide and Reference*.

The XML Legacy Gateway uses specified time delays when interacting with a host application. These time delays control when the screen is read to display the latest host screen to the user.

There are two delays:

Start Delay
Controls when to read the initial screen while accessing the host. The Start delay default is 1 second.

Interval Delay
Recognizes when a screen is no longer changing and is therefore ready to display to the user. The Interval delay default is 1.5 seconds.

To override the defaults for these delays, edit the hPubPortalData.xml configuration file created by the XML Legacy Gateway Portal Administrative servlet. This file is located in the Host Publisher runtime directory under Server\production\documents\xmlLegacyPortal\hPubPortalData.xml.

Edit this file with an ASCII editor. Each record in the file describes one session as defined with the XLGW Portal Administrative servlet. Specify values for DELAY_START and DELAY_INTERVAL to override the defaults. Specify the values in milliseconds; for example, a value of 1000 is 1 second.

Host Publisher XLGW portal page

As an end user, you can use the hPubPortal servlet to access the Host Publisher portal page at the URL appropriate for your version of WebSphere Application Server:

Running on WebSphere Application Server 2.0.3:
http://*servername*/servlet/xmlLegacyPortal.hPubPortal

Running on WebSphere Application Server 3.0.2:
http://*servername*/_IBM_HP_WebAdmin_/hPubPortal

where *servername* is the name of the Web server where Host Publisher is installed.

The portal page consists of administrator-defined links stored in the hPubPortalData.xml file (described in "hPubPortalAdmin" on page 22). These links use defined parameters to access the xmlLegacyGateway servlet, and enable you to use the xmlLegacyGateway servlet to access a desired host session.

Using the xmlLegacyGateway servlet to interact with the host application is similar to using a standard terminal emulator, except data input is performed using a Web browser as follows:

- The Tab key moves among entry fields
- Normal keyboard input is accepted in the entry fields
- Buttons that perform terminal function key commands (PF1, PF2, Clear, Enter, and so forth) are present at the bottom of the page.

Two additional buttons that are unique to the xmlLegacyGateway servlet are Refresh and Disconnect.

Refresh
Updates the browser page to reflect the latest state of the host application. This is necessary because, unlike a traditional emulator, the xmlLegacyGateway servlet does not continuously update the host screen. The servlet updates the screen only when input is sent to the host application. Typically, this occurs when you click a function key button, such as Enter. Refresh enables you to receive an update of the host screen without first having to issue a command or type data.

Disconnect

Signals to the xmlLegacyGateway servlet that you are finished with the host application and the host connection. Resources used by this instance of the servlet are then freed, enabling more efficient access to the resources by other users. If you do not click Disconnect when you are finished, the host session is not freed until the WebSphere Application Server determines that the Web session is no longer accessing the servlet. This time-out value is a configurable WebSphere Application Server parameter that defaults to 30 minutes.

Note: We recommend that you run only one XML Legacy Gateway session from each browser window, and that each browser window be dedicated to the XML Legacy Gateway session.

Enhancing the xmlLegacyGateway servlet

In addition to being a Web-based terminal emulator, the xmlLegacyGateway servlet also enables Host Publisher to interact with host applications using XML data formats. The servlet can:

- Process host screens as XML data.
- Combine the host screen XML data with XML data from other applications.
- Present data to an end user in a traditional Web browser format.
- Receive user input through the browser.
- Use XML techniques to process the browser data.
- Use an XML interface to send data back to the host application.

Refer to the *IBM WebSphere Host Publisher Programmer's Guide and Reference* for more information on how to use the Host Publisher XML Legacy interface to write applications and servlets.

Tracing the xmlLegacyGateway servlet

Host Publisher regards the xmlLegacyGateway servlet as an Integration Object. Tracing and error handling for the servlet is therefore the same as for other Integration Objects, as follows:

1. Open Host Publisher Server Administration.
2. Open Server Tracing Options.
3. Check the box next to Integration Object tracing.
4. Click Submit Changes.
5. Highlight the xmlLegacyGateway object.
6. Click Get Objects to view the object.
7. Select all objects.
8. Click Submit Changes.

To see the trace file and error log, in Host Publisher Server Administration, select Problem Determination > View Trace.

Configuring WebSphere Application Server and your Web server for Host Publisher Server

The following instructions are the WebSphere Application Server and Web server configuration steps required by Host Publisher. If Host Publisher Server installation fails or creates configuration problems, follow these instructions as a solution. If Host Publisher Server installation is successful, these steps are performed automatically during installation, requiring no additional configuration by you.

This information refers to Host Publisher's installation directory, which differs by installation and platform as follows:

- **For Windows NT**, <install_dir> and <install_dir2> is machine-specific. The default for both is C:\HostPub.
- **For AIX and MVS:**
  - <Install_dir> = /usr/lpp/HostPublisher
  - <Install_dir2> = /var/HostPublisher
- **For Solaris**
  - <Install_dir> = /opt/HostPublisher
  - <Install_dir2> = /var/HostPublisher

Web Server Configuration

Modify the Web server's configuration file to contain the following Host Publisher aliases:

Running on WebSphere Application Server 2.0.3 or earlier:

```
Alias  /HostPublisher/doc/ install_dir/Common/doc/
Alias  /HostPublisher/ install_dir/Server/production/documents/
Alias  /_IBM_HP_WebAdmin_/ install_dir/Server/production/documents/
Alias  /_IBM_HP_WebAdmin_/ install_dir/Common/doc/
```

Running on WebSphere Application Server 3.0.2:

```
Alias  /HostPublisher  /install_dir/Server/production/documents/
Alias /_IBM_HP_WebAdmin_  /install_dir/Server/production/documents
Alias /_IBM_HP_doc_  /install_dir/Common/doc/
```

Note: /HostPublisher/ is the default value. If you chose to replace /HostPublisher/ with something different during installation, use that value here.

The Web server's configuration file and location differs based on Web server software and installation choices. For example, the default location for the configuration file for IBM HTTP Server on Windows NT is c:\Program Files\IBM HTTP Server\conf\httpd.conf.

When configuring the http Server for OS/390, be sure to add asterisks after each alias and path name; for example:

```
Alias  /HostPublisher/*  <install_dir>/Server/production/documents/*
```

Note: Depending on the configuration you are modifying, you may have to use a Pass statement instead of an Alias statement. Refer to other alias examples within the configuration to understand which statement to use.

WebSphere Application Server Configuration

Modify three WebSphere Application Server property files to support Host Publisher Server.

1. **bootstrap.properties**

   This file is located under WebSphere Application Server's properties directory.

   On MVS, the file is called jvm.properties and is located in /usr/lpp/WebSphere/AppServer/properties/server/servlet/servletservice. Update the bootstrap properties file's java.classpath entry to include the following:

   ```
   <install_dir>/Common
   ```

   For Windows NT and AIX, the java.libpath and java.path properties should point to the JDK installed by Host Publisher Server, or <install_dir>/Common/jdk/bin.

   For Solaris and OS/390, we rely on the JDK shipped and installed with WebSphere Application Server, so you do not need to update these platforms.
2. **servlets.properties**

   Note: This file does not exist in WebSphere Application Server 3.0.2, because the required information is added to the WebSphere Application Server configuration files or the WebSphere Application Serverdatabase during installation.

   In WebSphere Application Server 2.0.3, this file is located under WebSphere Application Server's properties/server/servlet/servletservice directory. Update this file with the following information:
   a. In the servlets.startup entry, include **HPAdmin** with a space between it and any other entries.
   b. In the servlets.classpath entry, include the following path:

```
<install_dir2>/Server/production/beans
```

c. Append the following configuration information for the HPAdmin servlet to the file:

```
servlet.HPAdmin.description=Servlet used for HostPublisher administration
servlet.HPAdmin.code=com.ibm.HostPublisher.Server.HPAdminServlet
servlet.HPAdmin.allow_delete=false
servlet.HPAdmin.initArgs=install_dir=<install_dir>,log_dir=<install_dir2
>/log,server_dir=<install_dir2>/Server
```

Note: The directory name separator is platform-specific in this file. It is a forward slash (/) for AIX, Solaris, and MVS; it is a backslash (\) for Windows NT.

3. **server.properties**

   This file is located under WebSphere Application Server's properties/server/servlet/servletservice directory. Update this file by setting the enable.urlrewriting property to true.

The configuration steps are now complete.

Stop WebSphere Application Server and the Web server. Restart the Web server, and both WebSphere Application Server and Host Publisher Server will become active.

Chapter 3. Using Host Publisher

You want to extend applications to the Web. Where do you start? This section helps you understand how to get started and how to accomplish some of the most common tasks you will complete.

Getting your information onto the Web using Host Publisher has four main steps. You need to:

1. Create Integration Objects that define the logic for accessing the information you want to publish. See "Creating an Integration Object for host access" on page 31 and "Creating an Integration Object for database access" on page 43.
2. Create Web pages that use the Integration Objects to generate Web content. See "Creating Web pages for Integration Objects" on page 45.
3. Put the Web application onto a Host Publisher Server. See "Transferring applications to a Host Publisher Server" on page 53.
4. Make the pages available over the Web. See "Using the Server" on page 11.

Overview

Host Publisher enables you to create JavaBeans called *Integration Objects*. Integration Objects are JavaBeans that encapsulate interactions with a data source, such as a database or a 3270 application, and return specific data from that source for use as output to Web pages or other Java applications. Integration Objects connect to the data source, navigate to the desired data, extract the data, and store them in Java properties, which can then be accessed.

Host Publisher also helps you generate special Web pages called JavaServer Pages (JSPs), that allow you to include Integration Objects right on the page, invoke them, and return their output for display on the page.

Integration Objects and the JSP pages that reference Integration Objects together form a Web application. Host Publisher Studio publishes the application to Host Publisher Server after you deploy your application to the Server. WebSphere Application Server then parses the JSP pages and converts the special JSP tags on the pages into Java servlets. These servlets are Java programs that run under WebSphere Application Server. They are used to invoke the Integration Objects and display their output to an HTML page (derived from the original JSP). The Web browser displays this HTML page.

When WebSphere Application Server detects that an original JSP has been updated, the Java servlet is recreated; otherwise, JSPs are converted into servlets only once.

To use Host Publisher Studio and Host Publisher Server to build and publish a Web application:

1. Build an Integration Object in Host Publisher Studio to access a data source and return desired data.
2. Build Web pages using one or more Integration Objects to form a Web application.
3. Transfer the Web application to one or more Host Publisher Servers.
4. Using Host Publisher Server's administration facility, start the server and deploy the new application.
5. Use a standard Web browser to access the first page of the application.

Using Host Publisher Studio

Host Publisher Studio has three parts: the Host Publisher Studio, Host Access, and Database Access. You can launch Host Access and Database Access from the Host Publisher Studio application.

Host Access and Database Access provide wizards and other tools that help you build Integration Objects. Host Access is used to access data from a 3270, 5250, or VT application, while Database Access is used to access a JDBC database. Once you create Integration Objects, you can use the Host Publisher Studio to create Java Server Pages (JSPs) that use them to generate dynamic Web content. You can also use the Integration Objects in other Java applications, or you can use Host Publisher Studio to publish other Java objects and beans.

The following sections provide instructions and information about how to complete common Host Publisher Studio tasks.

About macros

The Host Access application is used to build Integration Objects that access data from a 3270, 5250, or VT application. To do this, Host Access records macros that contain information about the way you connect to the host, how you navigate to the information you want to publish, and how you disconnect from the host. These macros become part of the Integration Object you create.

Note: Macros are not used by Integration Objects created using the Database Access application.

In Host Access, the Integration Object includes several types of macros:

Connect
Includes the information Host Publisher needs to connect to the host. The connect macro should contain the steps necessary for logging on to a system from as many initial states as possible. It should take into account different paths that might occur because the previous connection failed or was left in an unknown state. Host Publisher Server uses the connect macro to attempt to connect to a system and to recover from a previously-failed connection. Refer to "Recording conditionals" on page 37 for more information on recording alternate paths.

Data Includes information about how to navigate, extract, and organize the data you want to publish

Disconnect
Includes information about how and when to disconnect from the host. Typically, this means tearing down the network connection. Disconnect macros prepare the host connection to cleanly disconnect.

Macros work by following a sequence of host screens that you define. As you navigate through your application using a terminal screen, you define:

- The screens
- The actions to take on each screen (that is, keystrokes)
- Which screens can appear next after the action has completed for each screen

You can also define loops within the macro and alternate paths (for example, more than one ″next screen″ for a given screen) to follow. Host Access includes a wizard that guides you through recording macros, but you can use the Macro menu and the toolbar to fine-tune them.

When you run your macro on the Host Publisher Server by invoking your Integration Object, your macro will look for the screens you defined to appear on the host sessions and will execute the actions you defined for each screen.

Host Publisher uses IBM Host On-Demand to provide an interface where you can interact with a host and record the actions you take. You might want to edit a macro script after it has been created using Host Access. Manually editing macro scripts should only be performed by advanced users. For detailed macro script syntax information, see the *IBM WebSphere Host Publisher Programmer's Guide and Reference*.

Creating an Integration Object for host access

To create Integration Objects that collect data from applications on the host, use the Host Access application in Host Publisher Studio. To create the Integration Objects, you navigate to the information you want using a 3270, 5250, or VT connection. Host Publisher records the keystrokes you use and lets you define the host application screens that contain information by using macros. (See “About macros” on page 30.)

This section provides information to help you with the following tasks:

- Using the wizard (“Using the wizard”)
- Defining host connections (“Defining host connections”)
- Creating connection pools for host access (“Creating connection pools for host access” on page 33)
- Specifying a host and a user ID (“Defining host connections”)
- Recording interactions with a host (“Recording interactions with a host” on page 35)
- Recording conditionals (“Recording conditionals” on page 37)
- Recording loops (“Recording loops” on page 39)
- Identifying screen data areas and labelling data (“Identifying data to extract” on page 40)
- Replaying the interaction with the host and inspecting the data (“Verifying a macro” on page 41)
- Generating an Integration Object (“Generating an Integration Object” on page 42)
- Creating user lists (“Creating user lists” on page 42)

Using the wizard

When you use Host Access to create an Integration Object, it launches a wizard that can guide you. To start Host Access, open the Host Publisher Studio application, then click **Create > Host Access Integration Object**. The wizard starts automatically. To use the wizard, provide the information it requests and click **Next** to continue until the wizard tells you the Integration Object has been created.

The wizard will ask you for the information it needs to define the connection to the host. It will then connect you to the host server you specify and guide you as you connect to the host, navigate to the data you want to publish, select and organize the data, and disconnect from the host.

If you prefer to create your Integration Object manually, you can bypass the wizard and use the toolbar or menus.

Defining host connections

Before you can record the steps Host Publisher will take to reach the data you want to publish, you need to define how to get to the application itself. Often, this will mean connecting to a host machine. You need to choose the type of connection you want, provide the name of the host, either as a TCP/IP hostname or as an IP address, and provide a specific logical unit (LU) or LU pool name, if any, to use.

Creating connection pools for host access

For each Integration Object, Host Access allows you to create a new connection pool, select to use a default connection pool, or select to use a connection pool that you have already defined.

If you create a new connection pool, by default your connection pool will be set to use connection pooling. You can select to disable connection pooling for each connection pool. If you do not use connection pooling, your Integration Object connects to the host each time you request a connection. Without connection pooling:

1. The Integration Object requests a connection from the pool
2. The connect macro runs to log on to the host session
3. The Integration Object extracts the information
4. The Integration Object returns the connection to the pool
5. The disconnect macro runs to log off the host session
6. The connection is removed from the pool and discarded

Connection pooling keeps one or more connections to the host initialized, thus reducing the response time between when a client with a browser requests information and when the information displays on the page. You specify how many connections you want to remain active in the pool and ready for use, and when to remove connections from the pool. With connection pooling:

1. The Integration Object requests a connection from the pool
2. If an initialized connection is available, the connect macro does not run
3. The Integration Object extracts the information
4. The Integration Object returns the connection to the pool
5. The disconnect macro is not run
6. The connection is returned to the pool in its initialized state

You may specify for each connection pool to use one of several user accounts defined in the associated user list. See "Creating user lists" on page 42.

You can use the Connection Pools tab in Host Access to define pools of connections or follow the Host Access wizard to specify the connection pool. You can also use the Connection Pools tab to modify attributes of the pools, including connection configurations and user lists. You cannot use this tab to select the pool an Integration Object will use; use the Macros tab to select a pool for the Integration Object.

To create a new pool:

1. Select **New Connection Pool...** from the File menu.
2. Specify a pool name and use the checkbox to indicate if you want connection pooling enabled.

3. Define the connection configuration to use for the pool. Connection configurations define a particular host and its connection parameters. To save time, you can define a connection to a particular host once, then share the configuration between pools that connect to the same host.
   - If you want to use a previously-defined connection configuration, select a name from the list.
   - If you want to create a new connection configuration, select New and provide the information to establish a connection to the specified host.

From the Connection Pool tab you can access all connection pools created in Host Publisher Studio. The pool used by the current Integration Object is highlighted.

Use the tabbed panes on the right to configure connection pooling options. You can:
- Configure time-outs and connection limits
- Change connection configuration information
- Add users

When you finish customizing your connection pool, click the Macros tab to continue recording your Integration Object.

Securing passwords and other sensitive data in macros

When you record a macro using Host Access, you may notice that the macro script in the left frame shows the actual characters entered for passwords, even when the password is hidden from view in the host application.

When passwords, or any other sensitive data, are entered during the recording of connect, data, or disconnect macros, the passwords display as they are typed. No attempt is made to hide or scramble the passwords because the underlying macro script cannot process scrambled or encrypted data.

A preferred method for maintaining sensitive data is to save it as part of a user list definition. To define a user list, select the Connection Pools tab in the left pane and the User List Configuration tabs in the right pane. This list of user IDs and password pairs provide values for macros to use at runtime when connecting to the data source. This data can be extended to include other sensitive data, such as PIN codes and other access keys. Refer to the *IBM WebSphere Host Publisher Programmer's Guide and Reference* for details.

To access the user ID using the connect macro of the Host Access Integration Object, navigate to the place where you want to enter the user ID, select Insert Userid from the list, then select the user ID you want to use for this recording session. The macro tree will show the entry as _userid.

To access the password, navigate to the place where you want to enter the password, then select Insert Password from the list. The password associated with the user ID you previously selected will be automatically inserted. The macro tree will show the entry as _password.

Recording interactions with a host

When you use the wizard, it guides you as you interact with the host to navigate to the screen that contains the data you want to publish, specify the specific data, and define the way it should be presented. You use a terminal screen just as you normally would, and Host Publisher records your interactions as a macro. The macro steps are displayed in the left pane in the window in a tree view.

As you record your macro, you can stop and start recording at any time. If you select Connect, Data, or Disconnect macro in the tree diagram and then select **Record**, you begin recording at the end of the selected macro. If you select a screen node with a definition matching the currently-displayed screen and then select **Record**, you begin recording at the selected node. If you select **Record** on any action or keystroke for a screen with a matching definition, you begin recording at the selected node.

If you select a screen node with a definition that does not match the currently-displayed screen on the terminal and then select **Record**, a jump screen is inserted after the currently-selected screen node and you begin recording after the jump screen.

You can use the toolbar and menus to perform many tasks, including:

1. Record interactions without using the wizard
2. Define an entry field
3. Add conditional paths or looping to the macros you create
4. Define a screen

After you stop recording, you can use the shortcut menu to modify a specific step in the macro by selecting a step in the tree and right-clicking on it.

Later, when you execute your Integration on the Host Publisher Server, your recorded macros will run so that the steps you followed to access the data during creation of the Integration Object are the same steps used to return the data to the Web browser.

Defining a screen

Defining a screen provides a way for the macro to identify that it has reached the desired host screen. When the macro plays, it waits until the screen is recognized as defined before playing more keystrokes. (Each step in a macro has at least one screen defined as the possible next screen; Host Publisher waits for the expected screen to appear before playing the actions associated with the next step of the macro.)

When you are prompted to define a screen, you must either define the screen or respond that you do not want to define the screen before you can enter more information on the host session screen. If you choose not to define the screen, the completed macro plays the recorded keystrokes regardless of which terminal screen appears.

Make sure you always specify a detailed screen description to ensure that the last screen segment arrives before the actions in the tag are executed.

If the screen definition is not detailed enough to ensure that your screen does not unexpectedly match a similar screen.

Some tips for defining a screen in a macro are:

1. Never define a screen by text that could change over time, such as a connection identifier, date, time stamp, or user ID.
2. Define a screen from minimal text on the screen, simplifying the recognition process and reducing the possibility for error.
3. Define a unique string that can be located anywhere on the screen without specifying its location; this eliminates the possibility of the string being in a different position and your screen not being recognized.
4. Most screens are drawn in sections, depending on the complexity of the screen. Define the screen using text that is drawn last, so Host Publisher will wait until the screen is complete before continuing with the macro.
5. Define global screens and associated actions for those screens that might appear at any time, such as monthly reminders or messages from colleagues. Global screens allow you to define a generic action to take (such as a clear screen command) if such a screen is encountered, allowing the macro to return to the normal flow. See “About global screens” on page 41.

Note: If you use Host Access to record your macro, you can click Back to navigate back through your actions; however, you can navigate back through only one screen definition. As you are navigating back through the wizard, when the screen display on your terminal screen does not match the screen definition selected on the macro tree (the macro tree selection navigates back along with the wizard), you cannot continue to navigate back.

Using input variables, conditionals, and looping

The macros you record using Host Access can be simple or complex. Simple macros follow a straightforward path; each step leads to only one next step.

Most often, the macros you record will be complex; they will include steps that lead to a choice of several steps, or they will include sequences of steps that repeat. When you have an input variable, for example, the user could enter information that leads to another prompt or to a loop of repeated actions. "Using input variables", "Recording conditionals", and "Recording loops" on page 39, describe how you can use commands on the toolbar to add complexity to a simple macro.

Using input variables: You use input variables for information you want the user (or another Integration Object) to provide. This information is not coded into your macro and is provided when the user makes a page request. For example, if your application enables the user to search for information about a person, you could use an input variable to contain the name to search on.

To insert an input variable:

1. Start recording a macro.
2. When you reach the point where the user will enter information, click the **Enter input variable** button on the toolbar.
3. On the window that appears, type a name for the variable and the default value for the field. You use the name when you design a Web page that uses this Integration Object. For example, you might type `person` for the input variable and `Lewis, Sera` for the value of the input variable to be used during macro recording. The value you provide is used only in the current recording session and does not become part of the macro.

Recording conditionals: Conditionals enable you to handle the situation where a host application could respond to a command (or keystrokes) with more than one screen. Conditionals also enable you to record an alternate path for your macro to follow. When you record a conditional, your screen will have more than one next screen defined in the macro tree.

For example, Figure 2 on page 38 shows an example of a conditional in a connect macro. When the connection is established, the terminal screen can display either a logged-in screen or a welcome screen that must be cleared before the logged-in screen will display. You can use **Insert Conditional** to define how Host Publisher handles either screen.

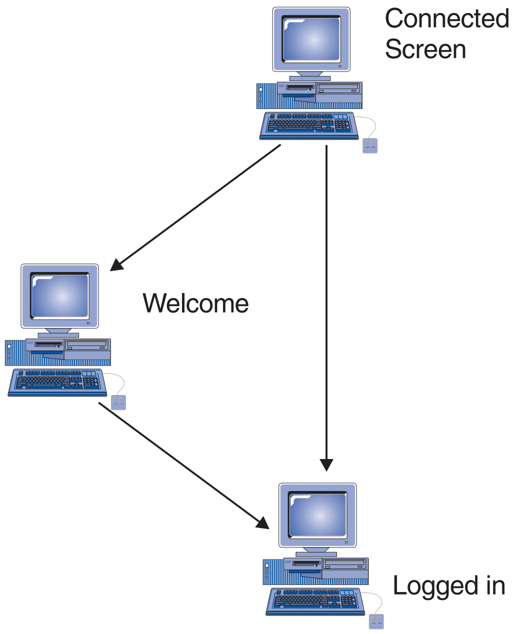

Figure 2. Host Publisher conditionals

To insert a conditional in a macro:

1. Record one path through the macro. For example, record the macro with the connection in a state so that the logged-in screen appears.
2. Put the condition in a different state, then replay the macro and cause the alternate condition to occur. Another way to cause an alternate condition is to specify a different value for an input prompt.
3. Host Access will report that a different-than-expected screen was encountered and display the Macro Play Error window. Click **Record an alternate path** in the Macro Play Error window.
4. Record the keystrokes you take to move from the new screen back to the main path of the macro, then stop recording. You might have several screens to define before you return to the macro, or you might not return to the main path at all. One way to get back to the main path is to highlight the last step in the conditional path and then click **Jump to defined screen** on the toolbar. On the window that appears, select the name of the screen in the main path you want to use as the destination for the jump. You must have already defined the screen before you can select it to be the destination of a jump.

   Note that you can have several conditions for one step of a macro; for example, one screen might have several next screens associated with it in the macro tree. Each screen that you define has actions associated with it.

When you play your macro and Host Access encounters one of the possible screens you specify, it performs the actions associated with that screen.

In the macro tree, the actions and the next screens associated with a screen are indented under the screen's entry.

Recording loops: Looping enables you to define an action that should be repeated until a condition is met.

For example, Figure 3 shows an example of a loop in a data macro. When the user requests data, the connection returns one or several screens of information. You want to display all the information, so you need to define a way for the macro to display all the screens and recognize the last screen of data. You can use **Start loop** to define how Host Publisher handles either condition.

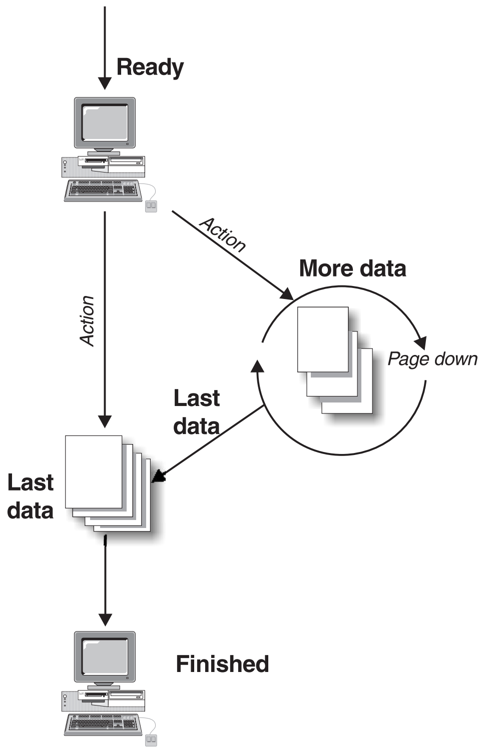

Figure 3. Host Publisher looping

To insert a loop in a macro:

1. Start recording your macro.

2. When you have reached the point where the loop will occur, click **Start loop** on the toolbar.
3. The Recording Loop wizard starts and provides the steps you will perform to record the loop. Click **Next** to begin.
4. Follow the wizard instructions. Navigate the terminal to the screen that contains the data to extract and define that screen.
5. Follow the instructions in the Recording Loop wizard to identify the data to extract and its format. Click **Next** to continue.
6. Type the keystrokes that will cause the terminal to advance to the second screen in the loop. The keystrokes you type will be repeated continuously until the ending loop condition is met. If you performed a data extraction, the data will be accumulated as the loop repeats. Click **Next**.
7. Host Access stops recording your actions. Use the terminal screen to navigate to the last screen in the loop. When the screen appears that indicates that the loop has finished, click **Next**.
8. Host Access starts recording your actions again. Define the current screen displayed in the terminal. In your definition, identify something on the screen that is displayed only when the loop is complete; for example, use words like *Bottom* or *Finished*, or use empty screen lines that do not appear until the last screen of the loop.
9. Press **Next**. The Recording Loop wizard ends and the loop is complete.

Identifying data to extract

When you use the Host Access wizard to create a macro, it helps you navigate to the data you want, identify the data, and structure the data for publication.

The wizard first guides you through defining a host connection, then starts a terminal screen with the host you defined. The wizard will ask you to use the terminal to navigate to the desired data, and will guide you through selecting the data for extraction. You can:

- Select a table of data
- Select a region of text data
- Select multiple regions on a screen by choosing **Capture More Data**

Assign a unique name to each region or box of text you select.

After Host Access guides you through creating an Integration Object, you can use Host Publisher Studio to create a Web page where the data will appear.

For example, if your Integration Object provides a connection to a telephone directory application, your Web user might enter a name to search for. The application displays a table of information about the people whose names match. You can publish all the columns in the table, or you can specify only some columns. You might want to publish only the names and phone numbers and not the office address or job description. You might also want to provide the name of the person's manager, which is displayed on another screen. Using Host Publisher Studio, you can select all the information you want, arrange it, and display it to the user as though it were one piece of data.

Verifying a macro

After you record a macro, verify that it works correctly. To verify a macro in Host Access:

1. Open the Integration Object that contains the macro you want to verify. You will see the steps of your macro displayed in the tree under the Macros tab.
2. On the toolbar, click **Play** to play the connect macro
3. If there are no errors, click **Play** again to play the data macro and inspect the captured data.
4. Click **Play** again to play the disconnect macro.

You will get a message if there are errors that prevent the macros from completing. If a macro does not complete correctly, it might be because there are some screens that appear that you did not define. You might need to define those screens (often these are global screens, such as screens that appear at different places and require you to press CLEAR or another key to continue).

About global screens

Global screens handle application screens that might appear at any time during the execution of your macro. When the same action is required each time such a screen is encountered, you can use global screens. During the execution of a macro, each *next* screen is checked sequentially in the order it is listed. If a next screen does not match, the set of defined global screens are checked for a match. If a match is found, the action associated with the global screen is performed and the macro continues. After the actions associated with the global screen are executed, the next screens are examined again for matches in the order they are listed.

For example, if you are recording a 3270 VM application logon sequence as a macro, you may notice that messages sometimes display after you type the user ID and password. Each time you see one of these messages, you will press CLEAR to remove the message and continue with the macro. When you define these message screens as global screens, you do not need to anticipate when they will occur. The macro player automatically executes a [clear] when the screens are encountered in the connect, data loop or disconnect macro.

Generating an Integration Object

After you record your macro, you create an Integration Object by selecting File > Save. If you have deselected **Automatically Generate Integration Object on Save**, you create an Integration Object by selecting File > Create Integration Object or by clicking Finish.

You are prompted to name your Integration Object. Use a unique name for each Integration Object to avoid overwriting existing objects on the Server. If you publish an application to a Server that contains a file with the same name as an existing fie, a warning message displays. You then have the options to:

- Continue the file transfer and overwrite the existing file.
- Stop the file transfer and keep the existing file.
- Give general permission for overwrites for the current and any future existing files found during this transfer to the server.

Note: These options are not available on an S/390 that does not test for pre-existing files.

Application transfer places the application files in a staging directory, which is cleared when an administrator deploys the application. The situation occurs only when two applications with the same name, or applications with shared parts, are transferred before one of them is deployed. Continue with the transfer and overwrite the existing files *only* if you are sure the application files are consistent with each other and can later be successfully deployed.

Creating user lists

User lists contain user IDs, passwords, and other information that are associated with host accounts.

Imagine that you are creating an application that connects to a host and has various user accounts it can access to connect. The accounts have associated user IDs and passwords.

If you have a user list associated with the connection pool, and you have recorded your connect macro to use this list, Host Publisher uses the list you defined to supply the values that it passes on to the host when you run your Integration Object on the Host Publisher Server. If the first user ID in the list is in use, Host Publisher uses the next user ID (unless you have specified that user IDs can be connected more than once).

Host Publisher tracks the IDs that are being used and updates the list as IDs become available for use; for example, imagine your application has three IDs and passwords as follows:

ID	Password

pam	pampw
sarkar	sarkarpw
villari	villaripw

If **pam** is in use, Host Publisher uses **sarkar**. If **pam** then becomes available, Host Publisher uses **pam** for the next connection. If **pam** and **sarkar** are both in use, Host Publisher uses **villari**.

For information on adding additional properties, see the *IBM WebSphere Host Publisher Programmer's Guide and Reference*.

Creating an Integration Object for database access

You use the Database Access application to create Integration Objects that encapsulate a database statement. To create the Integration Objects, you specify your structured query language (SQL) statement. If you are querying a database, you specify the data you want to retrieve from the database table.

This section provides information to help you with the following tasks:

- Using the wizard ("Using the wizard")
- Defining database connections ("Defining database connections")
- Retrieving information from a database ("Retrieving information from a database" on page 44)
- Generating an Integration Object ("Generating an Integration Object" on page 44)
- Verifying a SQL statement ("Verifying a SQL statement" on page 44)
- Creating connection pools for Database Access ("Creating connection pools for Database Access" on page 45)

Using the wizard

When you use Database Access to create an Integration Object, it launches a wizard to guide you. To start Database Access, open the Host Publisher Studio application, then click **Create** > **Database Access** > **Integration Object**. The wizard starts automatically. Tabs guide you through the process of building and executing a valid SQL statement. You can navigate by clicking **Back** or **Next** at the bottom of the panel. Provide the information it requests for each tab. When you complete the wizard, the Integration Object is created when you save the file or when you select **Finish**.

Defining database connections

Before you can build your SQL statement to access the data you want to publish, you need to connect to your database. Select a database driver, then type the name of your database inside the [ ] and any required user ID and password. Select Connect to connect to your database.

Retrieving information from a database

The Database Access wizard helps you navigate to the tables that contain the data you want, specify conditions to identify the data, and sort the data you want to publish.

You specify the SQL statement type and select the tables in the database to access. You determine which columns of the tables to include, applying conditions to the data in the columns. You can also sort the order in which the data is displayed.

The SQL statement types are:

- Select
- Select Unique
- Insert
- Update
- Delete

After Database Access guides you through creating an Integration Object, you can use the Host Publisher Studio application to create a Web page where the data will appear.

Generating an Integration Object

After you create your SQL statement, click Finish (or select **File > Save**) to create an Integration Object.

After you import a completed Integration Object into an application and publish it, Host Publisher will use the SQL statement you created to access the data that will be returned to the Web browser.

Verifying a SQL statement

To verify your generated SQL statement, click Run SQL... on the SQL tab. If successful, a result window displays your results with a maximum number of records. You can specify the maximum number of records to display by clicking Run SQL Settings... This maximum is only used by Database Access to run a sample of your SQL statement. The maximum will not be used when your Integration Object executes on the Server; all of your records will be returned when executing on the Server. Setting a maximum number of records to display using Database Access will avoid some out of memory conditions because your Server is likely to have more memory than your Studio machine.

If an error occurs when running the generated SQL statement, the Database Access Exception Window appears. When this window appears, do the following

1. Note the cause of the error.

2. Click OK to exit the window and return to the SQL tab.
3. Make corrections to the data on the appropriate tabs.
4. Return to the SQL tab.
5. Click Run SQL again.

Creating connection pools for Database Access

You can use the Connection Pools tab in Database Access to define connection pools. You can also use the Connection Pools tab to modify attributes of the pools, including connection configurations and user lists.

For each Integration Object, Database Access allows you to create a new connection pool, or to select to use an existing connection pool that you have already defined.

If you have followed the tabs of the Database Access wizard in order, you have already connected to a database. The data on the Connection Pools tab will be primed with the database driver, database URL, user ID and password you used when you connected to the database.

Click Pool Properties... if you want to enable connection pooling for this connection pool. If you do not use connection pooling, your Integration Object connects to the database each time you request a connection. Connection pooling keeps one or more connections to the database initialized, which may reduce the response time between when a client with a browser requests information and when it is displayed on the page, especially if you are using a remote database. You specify how many connections you want to remain active in the pool and ready for use, and when to remove connections from the pool.

You may specify only one user ID and password for each connection pool. This is because, unlike Host Access where more than one user ID and password is allowed for each connection pool, user IDs and passwords for databases are always reusable; that is, they can be used by more than one active connection to the database.

Creating Web pages for Integration Objects

You use the Host Publisher Studio to create Web pages that access the Integration Objects you created using Host Access or Database Access. Host Publisher helps you design a page and integrates the Integration Objects for you. You can also use Host Publisher to import other Java beans or objects.

This section provides information to help you with the following tasks:

- Using wizards ("Using the wizards" on page 46)
- Specifying Integration Objects to publish on the Web ("Specifying Integration Objects to publish on the Web" on page 48)

- Choosing properties to render and control appearance ("Choosing properties to control appearance" on page 49)
- Previewing a page ("Previewing a page" on page 49)

Note: Throughout this section, we use the term Integration Object to refer to Integration Objects, as well as other Java objects, that you import into the Studio to use in your application.

Using the wizards

When you use Host Publisher Studio, you can build your application using wizards to guide you. The wizards guide you through naming the new application, importing Integration Objects and other Java object into the application, and creating pages for publishing data.

You can select to create a new application or modify an existing application. To create a new application, specify an application name and follow the wizards using the Next and Back buttons at the bottom of the wizard windows. You can create an entire working application using the New Application wizard. Use the buttons at the bottom of the windows to move from one wizard to another.

If you prefer to create your application manually, you can bypass the wizard and use the menus. To exit the wizard, select Finish at any time. Some of the items on the menus also launch wizards to help you accomplish tasks.

Note that:

- The window with the Define button contains a column that indicates whether an Integration Object has been defined (Yes or No).
- The window with the Render button contains information in the Control and Label columns after the selected elements have been rendered.

The studio wizards are:

New Application wizard
This wizard guides you to name the new application and select the pages and Integration Objects to add to the application. You can use this wizard to create an entire functioning application, or click the Finish button at any time to exit the wizard. If you exit before completing the wizard, use the pull-down menus on the Main Panel to complete your application. This wizard is launched by the New Application item on the File pull-down menu.

New Integration Object Wizard
This wizard guides you to add an Integration Object to a Web page, resolve the Integration Object inputs, and render the outputs. This

wizard is launched by the New Application wizard or by the Insert Integration Object... item on the Insert pull-down menu.

New Page Wizard
This wizard guides you to create a new page, name the page, and specify its function. You identify the resources to add to the page, and whether the resources provide input to an Integration Object or render the output of an Integration Object. This wizard is launched by the New Application wizard or by the HTML Page item on the Create pull-down menu.

Insert Output Control Wizard
This wizard guides you to add an output control to the current page. As part of creating the output control, you specify the Integration Object output to be displayed with the control. This wizard is launched by the New Page wizard, the New Integration Object wizard, and by various items on the Insert pull-down menu.

Insert an Input Wizard
This wizard guides you to add an input control to the current page. When creating the input control, identify the form that contains the new input control and the destination of the submit action for that form (if you are creating a new form). This wizard is launched by the New Page wizard, the New Integration Object wizard, and by various items on the Insert pull-down menu.

New Error Page Wizard
This wizard guides you to create an error page to display to the client when an Integration Object error is encountered. Use the error page to inform the client that an error occurred and how to address the problem, such as calling a service telephone number. When you create an error page, all the Web pages in your application containing Integration Objects are updated with the error page name as an input parameter to the Integration Objects. If you add Integration Objects to new or existing pages, the error page name is added as an input parameter to those Integration Objects, also. If an error page already exists in the application when you create a new error page, the new error page name referenced by the Integration Objects replaces the previous error page name. This wizard is launched by the New Application wizard or Error page item on the Create pull-down menu.

Note: References to Integration Objects in this section do not apply to the Java objects you have imported.

Transfer to Server Wizard
This wizard guides you through the process of transferring application parts to Host Publisher Servers. All Host Publisher Servers to which you want to transfer your application must be configured

using the Preferences Dialog, displayed by the Preferences item on the Options pull-down menu. This wizard is launched by the Transfer to Server item on the File pull-down menu.

Note: During creation of Web pages in Host Publisher Studio, some of the windows contain multiple layers of buttons. You may not be sure how to proceed after Define and Render steps.

Carefully review the instructions presented at the top of each Studio Wizard window. The required steps are listed in chronological order.

Select Next only after you complete all of the actions required on the current window and are ready to move to the next step in the Web page creation process.

Select Back when you want to return to the previous step in the Web page creation process.

Select Finish only when you are ready to leave the Host Publisher Studio wizard.

Specifying Integration Objects to publish on the Web

A Host Publisher application is a collection of Web pages, Integration Objects, and other Java objects that enable the end user to interact with multiple data sources.

The Host Publisher Studio enables you to build a series of Web pages using standard HTML tags in conjunction with special JavaServer Pages (JSP) tags. These standard tags manipulate Java objects (such as an Integration Object) on the Web page. These tags also provide the ability to specify Java object output directly on the page.

JSP tags are designed for any Java object or bean. You can use the Host Publisher Studio to import other Java classes or beans, in addition to Host Publisher Integration Objects, into your application and publish them into Web pages.

The Host Publisher Studio accepts as input any Integration Objects, other Java components, or any prebuilt HTML pages to which you want to add data interactions.

The output is a collection of Web pages (HTML and JSP), which have been generated or modified to interact with Integration Objects.

Choosing properties to control appearance

Integration Objects have properties that return data to the user. The output data can be embedded within graphic controls. Host Publisher will format table, list box, text entry, and text area controls for you.

Previewing a page

When you are creating an HTML or JavaServer page (JSP) in the Host Publisher Studio, you can preview the page with a Web browser to verify the layout. The layout of the page is displayed, but the JSP tags and associated Java objects are not shown.

Satisfying Integration Object input

Before an Integration Object runs, it may have inputs that require data. This data can come from an HTML form on another page, or from other Integration Object output. The wizards guide you through deciding how to satisfy Integration Object inputs.

Showing Integration Object input also as an output

When building Integration Objects, Host Publisher Studio creates an associated output method (*getter*) for any input specified. This enables the page to echo a value from an Integration Object as output, based on a value sent from another page. For example, on one page, a user requests the name of a person to use as input to an Integration Object on another page. The second page searches for a telephone number for the person and uses the output method for the name provided as input. You could construct a sentence on the page like this:

```
The phone number for (name) is (number)
```

where (name) is the value derived from the input and (number) is the phone number found by running the Integration Object.

Creating composite applications

Composite applications combine multiple Integration Objects to produce a single stream of output information to the user. Composite applications enable you to use the output of one Integration Object as input to another or fill a table with data from several different Integration Objects that access different data sources. The four ways to produce a composite application using Host Publisher Studio are described in the following sections.

Combining Integration Object output

This type of composite application contains multiple Integration Objects on a page and displays their combined output on the page. For example, a user could use an application to query his or her own employee information from a central human resources application and database. The application maintains departmental and contact information. The database maintains payroll and insurance information. With two Integration Objects, one that accesses the application and one that accesses the database, a table can be displayed on an output page containing the output from both Integration Objects. The information in the table appears to originate from a single source instead of two different applications.

To create this type of composite application using the Host Publisher Studio wizards:

1. Create a new application.
2. Import the Integration Objects into the project.
3. When defining how the Integration Objects are used, specify that they are to be added to the same execution, or output, page.
4. If any of the Integration Objects require input information, specify that the input controls requesting the input data are all added to the same form on the same input page.
5. When defining how the output is to be rendered, you work with one Integration Object at a time. If you want to create an output table combining data from multiple Integration Objects, close the New Application wizard and select Insert > Output Control > Table from the menu bar. Another wizard guides you through the process of creating a table, including selecting the Integration Object outputs to use to fill the table with data.

You should have two pages in your application. One page requests input in an input form and delivers it to the other page. The second page executes the Integration Objects and displays their data. If the Integration Objects require no input data, you have no input page.

Sequencing Integration Objects on a single page

You can use one Integration Object to gather data from a data source and send the data to another Integration Object as input. For example, one Integration Object takes a user's name as input and provides as output that user's employee department number. This department number is then sent as input to another Integration Object, which takes the department number and locates the members of the department using another application. The department list is displayed to the user on the page.

The easiest way to accomplish this is to have both Integration Objects on the same output page. Be sure that the Integration Objects appear on the page in the correct order.

To create this type of composite application using the Host Publisher Studio wizards:

1. Create a new application.
2. Import the Integration Objects into the project.

3. Define the first Integration Object in the logical sequence first. Specify how to provide input to this Integration Object, if applicable (using another form page, for example). Do not render any output from this Integration Object on the page. The output is used by the next Integration Object.
4. Define the second Integration Object. To satisfy its inputs, specify that they are to be satisfied using other Integration Object output.
5. Select the other Integration Object from that page as providing that output. Note that Integration Objects only accept single-valued inputs, so you can only use single-valued outputs from other Integration Objects as input to another Integration Object.
6. Render the output of the second Integration Object on the page.

You should have two pages in your application. One page requests input in an input form and delivers it to the other page. The second page executes the first Integration Object using the input, executes the second Integration Object using the first Integration Object's output as input, and displays its own output on the page.

More Integration Objects can be added to the page to lengthen the sequence.

Sequencing Integration Objects on multiple pages

You can use multiple Integration Objects to return data to the user in steps, allowing the user to act upon the data and return selected data for the next Integration Object to use. For example, an application contains three pages. The first page uses a phone number-lookup application to request a name. The second page displays all matches found, allowing the user to select one. The third page displays the phone number of the selected individual.

This type of composite application consists of an initial form page followed by a series of pages. Each page contains an Integration Object and an input form filled with the output from that Integration Object. Each Integration Object executes, using the selections from the previous page as input, and fills the new input form on the current page with output. The user can make a selection and continue to the next Integration Object.

To create this type of composite application using the Host Publisher Studio wizards:

1. Create a new application.
2. Import the Integration Objects into the project.
3. For the first Integration Object, specify that it should be placed on execution page output1. Satisfy the first Integration Object's inputs using an input form on a page called inputForm. This inputForm page will submit results to the output1 page.

4. Render the first Integration Object’s output to a list box control. Because a list box also serves as an input control (since you can select an item out of the list), a form is defined for it on the output1 page. Do not specify a page for this form’s destination.
5. For the second Integration Object, specify that it should be placed on execution page output2. Satisfy the second Integration Object’s inputs using the input controls created on page output1. The input form on output1 is modified to point to output2 as the destination for the data.
6. Specify that the second Integration Object’s output should go into a list box, creating a new form without a destination page.
7. Repeat steps 5 and 6 for the third Integration Object. Specify output3 as the execution page and output2 as containing the form providing input.

The result is an interactive composite application that enables a user to interact with data before it is passed to the next Integration Object. The application consists of four pages: inputForm, output1, output2, and output3. Pages inputForm, output1, and output2 each have input forms that provide data to the Integration Object on the next page. Pages output1, output2, and output3 all show the data resulting from executing the Integration Objects on that page.

Sequencing Integration Objects between non-adjacent pages

You can create a composite application similar to the one described in “Sequencing Integration Objects on a single page” on page 50, but the Integration Objects reside on different pages. The output of one Integration Object is stored within the Web session and retrieved by another Integration Object.

You can use this type of composite application to send an output value to an Integration Object as input on another page where an input form is not involved. For example, if there is no form between a page containing an Integration Object that returns an employee’s department number and another page that takes the department number and returns all of the employees in that department, there is no way to send data from the first page to the next. Using Host Publisher Studio, you can cause the first Integration Object to save the department number within the Web session to be retrieved by the second Integration Object. This method requires no interaction with the end user and does not demand that the two pages be adjacent to each other in the logical page hierarchy in your web site.

To create this type of composite application using the Host Publisher Studio wizards:

1. Create a new application.
2. Import the Integration Objects into the project.

3. For the first Integration Object, specify that it should be executed on page execute1. If it has inputs to satisfy, create an input form on page input1. Do not render any of this Integration Object's output.
4. For the second Integration Object, specify that it should be executed on page execute2. Satisfy its inputs from the outputs of the first Integration Object. You can render the output of the Integration Object on this same page (execute2).

When you complete the wizards, a statement will be added to the execute1 page to insert the Integration Object's output into the HTTP connection. On execute2, the value is extracted from the HTTP connection and set as input to the second Integration Object. You must ensure that execute2 occurs after execute1 in your page hierarchy.

Importing Java objects

You can import Java objects into a Host Publisher Studio application. The Java objects can be Java classes, beans, or Host Publisher Integration Objects. You can import Java Class files, ZIP files, or JAR files. ZIP or JAR files can contain many Java objects. If you select a ZIP or JAR file to import, you choose which Java object to import from the file.

If you import an Integration Object generated by one of the Host Publisher Access applications, such as the Database Access application or Host Access application, the Host Publisher Studio knows the inputs, outputs, and execution methods.

If you import a Java class or bean in your Web application that is not a Host Publisher Integration Object, you verify what the inputs, outputs, and execution methods should be.

Transferring applications to a Host Publisher Server

The Host Publisher Studio gives you the ability to transfer your Web application to Host Publisher servers, making the application ready for deployment.

When you transfer your application, all application parts are collected and organized in a directory structure similar to the structure on a Host Publisher server. The directory structure contains the following subdirectory types:

Shared
Contains the parts of the application, such as Integration Objects and connection configuration files required by the Integration Objects, that can be shared with other applications.

Application-specific
Contains the application pages. The name of this subdirectory is the same as the application name.

The wizard that guides you through the transfer process asks you how to secure potentially sensitive user data contained in user lists, such as passwords. Any data in a user list except the user ID can be encrypted. You can choose to:

- Not secure the data, leaving the values readable from the configuration files
- Scramble the data (weak encryption)
- Encrypt the data (strong encryption)

Encryption requires that you specify an encryption key. You are prompted for a password to be used for strong encryption. When a Web application containing a strongly-encrypted user pool is deployed on the Server, the password you specified for strong encryption must be specified when the Server is restarted. Without this password, the Server cannot be restarted. This password is also required by Host Publisher Server Administration when an administrator modifies the strongly-encrypted user pool. If another user pool is created with a property that requires strong encryption, the same password must be specified to encrypt that user pool. The Host Publisher Server can only have one startup password, so the same password must be used by all strongly-encrypted user pools. If weak encryption is used, no password is required.

Once the application is organized into a staging directory on the Host Publisher Studio machine, the file transfer protocol (FTP) is used to transfer the contents of the application to the specified Host Publisher Servers.

During the transfer process, you are asked which servers you want to transfer the application to. Each server must have a server information definition, created from Options > Preferences. When defining the server information, specify the target directory for the transfer and the type of platform. This target directory must correspond to an FTP alias what matches the Host Publisher Server installation directory on that machine. Therefore, before a transfer can take place, the FTP service on that server must be configured to allow access to the Host Publisher Server installation directory.

Here are some examples of target directories by platform:

AIX	/var/HostPublisher

Windows NT	C:\HostPub Note that FTP does not allow the use of a drive letter as a destination. You must configure the FTP service you are using on Windows NT to have an alias (such as /HostPublisher) point to C:\YourInstallDirectory. You then specify /HostPublisher as your target directory for this server.
Solaris	/var/HostPublisher
S/390	/var/HostPublisher
OS/400	/QIBM/UserData/HostPublisher
NetWare	SYS:\HostPublisher

If Host Publisher Studio and Server are installed on the same system, you must still create a server information definition for the local system. Specify **localhost** as the host name so Host Publisher Studio will know to perform a local file copy instead of using FTP.

You can also use this method to transfer applications to remote Windows NT servers by sharing the target drive on the local system and performing a local transfer to that shared drive, again using **localhost** as the host name. In these cases, you must specify the drive letter and destination directory; for example: F:\HostPub.

Enabling tracing

You can enable tracing for the components of the Host Publisher Studio by editing the Studio.ini file located in the Studio directory. Find the entry for Enable Trace and change it to state Enable Trace=true. Tracing begins when the Host Publisher Studio components are restarted.

Trace information is written to separate files for each component:

Host Publisher Studio	webbridge.trc
Host Access	hostaccess.trc
Database Access	dbaccess.trc

The trace files will be located in the Studio directory.

Remote Integration Objects

See the *IBM WebSphere Host Publisher Programmer's Guide and Reference* for detailed information on Remote Integration Objects.

Using WebSphere Studio

You can use the WebSphere Page Designer to view and modify a .jsp file created by Host Publisher Studio. You can also insert an Integration Object's .jar or a .jsp file created by Host Publisher Studio into a WebSphere Studio project.

Using WebSphere Page Designer

WebSphere Page Designer can be launched from Host Publisher Studio to edit Application Pages created by Host Publisher Studio. Select Options > Preferences from the Host Publisher Studio menu bar to specify WebSphere Page Designer as the default page editor. When the Personal Page Editor is launched, WebSphere Page Designer will start.

Inserting a Host Publisher Integration Object into a WebSphere Studio Project

You can insert a Integration Object's .jar file created by Host Publisher into a WebSphere Studio project. To do this, in the WebSphere Application Server Preferences box, specify the Host Publisher .jar files, which contain the classes required by the Integration Object. These .jar files are:

- \Common\HPubCommon.jar
- \Common\HpRte.jar
- \Common\habeansnlv.jar\
- \Studio\jsdk.jar

Inserting a Host Publisher .jsp into a WebSphere Studio project

To insert a .jsp created by Host Publisher into a WebSphere Studio project, the WebSphere Studio project must use JSP version 0.91 and the Code Generation Style must be WebSphereJSPModel. You can then edit the .jsp with Page Designer.

Transferring .jar and .jsp files to the Host Publisher Server

Currently, all the files required to use the .jsp are transferred to the Host Publisher Server by Host Publisher Studio. If you insert a Host Publisher Studio .jsp into a WebSphere project and make modifications to the .jsp, you must copy the .jsp back to the Host Publisher Studio directory. Host Publisher Studio can then transfer the .jsp to the server.

Alternatively, you can use WebSphere Studio to transfer the .jar and .jsp files; however, you must then manually remove the .jar and .jsp files transferred to the server by Host Publisher Studio.

Note: If you do not use Host Publisher Studio to transfer your files to the Server, and you are using Host Publisher's default error page, the default error page must reside in the directory structure one level higher than the other .jsp files. You must place your files on the Server accordingly. Alternatively, you can create your own error page.

Chapter 4. Advanced features

This chapter discusses how to use Host Publisher's advanced features: Integration Object chaining and security.

Using Integration Object chaining

Integration Object chaining enables multiple Integration Objects to execute in sequence, each using the same connection. Integration Object chaining can only be used with Host Access Integration Objects, which have connections to a screen based application, such as a 3270 application. An Integration Object in an object chain leaves the connection in a state (at a particular screen) suitable for the next Integration Object in the chain to use from that browser. Integration Object chaining requires WebSphere Application Server and is not supported when invoking Integration Objects from custom Java applications.

You can use chaining to break up a complex application into multiple tasks, each task represented by an Integration Object. Host Publisher Studio ensures that the order in which Integration Objects are invoked is correct; however, if you edit the jsp files without using the Studio, you must ensure that the order of the Integration Objects is correct.

For example, if you have three Integration Objects in an object chain: A, B, and C, then you must use A first, then B, then C. If Integration Object C is invoked before Integration Object B, then when C requests its connection, the connection is not available in the correct state, and the Integration Object fails. Host Publisher Studio ensures the correct order for you.

To build an application using Integration Object chaining with Host Publisher Studio, you must first build the Integration Objects within the Host Access application, and then import these Integration Objects into Host Publisher Studio and build Web pages around them.

To build the first Integration Object in the object chain:

1. Use the wizard within the Host Access application to define a connection as you normally would (see "Using the wizard" on page 32 for information about defining connections).
2. Record the connect macro. (See "Recording interactions with a host" on page 35 for additional information.)
3. Record your data macro.

4. You will probably not end your data macro at the same point where you began (so that another Integration Object can pick up where you've stopped). Navigate to the desired ending point for the data macro, and click **Stop Recording** on the toolbar.
5. Navigate back to the point where you can disconnect from the host; that is, where the data macro of the last bean in the chain ends.
6. Highlight the Disconnect Macro node in the macro tree, and click **Record**.
7. Record your disconnect macro.
8. Before you create the Integration Object, select Options > Configure Integration Object chaining. On the window that appears, specify a stop state label for this Integration Object and specify that this object should be **first**. Host Publisher uses start and stop state labels to identify the Integration Objects that may precede or follow this one. For example, if this Integration Object has a stop state label of **connected**, only Integration Objects that have **connected** as a start state label can follow this Integration Object in a chain.
9. Save the Integration Object.

Note: The connect and disconnect macros are part of the connection pool.

To record a middle Integration Object in the chain:

1. Use the wizard within the Host Access application to create another Integration Object. When you define the connection, click **Share an existing connection pool** and select the configuration you used for the first Integration Object. All Integration Objects in the same object chain must use the same connection configuration.
2. Optionally, play the connect macro, or connect manually.

   Note: When you create the middle Integration Object, the full connect and disconnect macros are displayed in the macro tree; however they don't necessarily run for that object when it is executed on the Server. The connect macro runs before the first object and disconnect runs after the last object.
3. Use the terminal to navigate to the correct starting point for this Integration Object, and record the data macro by selecting Data Macro in the tree. Then click Record.
4. Select Options > Configure Integration Object chaining. On the window that appears, specify a start state label for this Integration Object that matches the stop state label of the previous Integration Object in the chain, specify a stop state label, and specify that this object should be **middle**.
5. Save the Integration Object.

To record the last Integration Object in the chain:

1. Use the wizard within the Host Access application to create another Integration Object. When you define the connection, click **Share an existing connection pool** and select the configuration you used for the first Integration Object. All Integration Objects in the same object chain must use the same connection configuration.
2. Optionally, play the connect macro, or connect manually.

   Note: When you create the last Integration Object, the full connect and disconnect macros are displayed in the macro tree; however they don't necessarily run for that object when it is executed on the Server. The connect macro runs before the first object and disconnect runs after the last object.
3. Navigate to the correct starting point for this Integration Object, record the data macro, and be sure to end at the same point where the first Integration Object in the chain started.
4. Select Options > Configure Integration Object chaining. On the window that appears, specify a start state label for this Integration Object that matches the stop state label of the previous Integration Object in the chain, and specify that this object should be **last**.

   Note: For the last in the chain, you will supply only the start state label.
5. Save the Integration Object.

The last Integration Object in your object chain should return the host screen to the same state from which the first Integration Object started.

Once you have all of your Integration Objects defined, return to the Host Publisher Studio, and import them into a new application. In Host Publisher Studio, the Available Objects tree, when expanded for an Integration Object, specifies the logical next and previous Integration Objects for the selected object. This makes it easier for you to identify the order in which the Integration Objects can be used.

Within Host Publisher Studio, you can build Web pages to form a sequential chain, where one page leads to the next after an input form is submitted (you can also build a single page that contains an entire chain). As you build a chain of pages, you can add the Integration Objects to the pages only in the logical order in which they were created for the chain.

For example, if you have three Integration Objects in a chain: A, B, and C, then you must use A first, then B, then C. The Host Publisher Studio wizards that guide you through building the application work the same way as for non-chained Integration Objects, but are more restrictive in how object chained Integration Objects can be used within the Web application.

When you have finished building your Web application, transfer it to a server, deploy the application, and test it (see "Transferring applications to a Host Publisher Server" on page 53 for information).

Configuring and using Secure Sockets Layer (SSL) support

Host Publisher uses Host On-Demand to provide connection support to 3270, 5250, and VT applications using Telnet protocols. Host Publisher uses the SSL support provided by Host On-Demand for securing these connections. Using a secure connection over SSL causes data flowing over the connection to be encrypted and therefore protected against observation by a third party.

For a connection to be secured, both Host Publisher and the Telnet server it is connected to must support SSL. To secure the connection, the Telnet server must provide a certificate, which is used to encrypt the data. This certificate uniquely identifies a machine on one end of the connection. No other server in the network should have the same certificate. When the server provides this certificate, it is called *server authentication*.

If, while defining your Host Access Integration Object's connection to the host, you configure SSL support as well as server authentication, you are indicating that you require the telnet server to provide the certificate with which to encrypt the data. If you select the server authentication option when configuring SSL, in addition to verifying the certificate itself, Host Publisher will perform a check to ensure the server's TCP/IP address matches the one specified in the certificate. The server's address must be a part of the certificate for this option to work.

Host Publisher will verify that the certificate is signed by a well-known certificate authority. If it is not, Host Publisher is required to have its own version of the server's certificate for verification purposes. (See information about self-signed certificates in the following paragraphs.)

If the telnet server has a *self-signed certificate*, the administrator of the server generated the certificate based on his or her own information without using a certificate authority. In that case, Host Publisher must have a copy of the self-signed certificate to secure the connection; however, if the server has a certificate signed by a well-known certificate authority, the certificate is guaranteed to be unique and secure, and Host Publisher is not required to have a copy of the certificate.

To use self-signed certificates, you must embed them within a Java class file and insert them into Host Publisher's CLASSPATH. The Host Access application uses this class file to establish a secure connection while the Integration Object is being created.

The gencert.bat tool shipped with Host Publisher is used to generate a certificate file for SSL enablement. The gencert.bat batch file, when used to generate a new certificate file, requests that you enter a password. To generate a proper certificate file, leave the password blank.

To use this utility, type the following command:

c:\HostPublisher\Studio\gencert.bat *certificate_file*

where *c:\HostPublisher* is the directory where you installed Host Publisher and *certificate_file* is the name of the self-signed certificate file from the server.

Press Enter when prompted for a password.

The result will be a .class file called CustomizedCAs.class. You must have a copy of this file on the Studio on the Server. On the Studio, copy this file into the Studio subdirectory (c:\HostPublisher\Studio, for example). On the Server, copy this class file to the \HostPub\Common directory for WebSphere Application Server 2.0.3. For WebSphere Application Server 3.0.2, copy the file to the \HostPub\Server\production\beans directory. You can place it anywhere in the CLASSPATH, so long as WebSphere Application Server picks up the .class file when processing a request for an Integration Object configured for SSL support. Host Publisher Server uses this file to establish the secure connection when the Integration Object is invoked.

Chapter 5. Troubleshooting

This chapter helps you identify problems and determine solutions with Host Publisher. If the solution is not documented, you are directed to gather the necessary information for IBM service.

Host Publisher problem determination procedure

Follow this procedure to resolve a problem found with Host Publisher:

1. Determine whether the error occurred in the Host Publisher Studio or Host Publisher Server, the task being performed when the error occurred, and the symptoms of the error.
2. If the error was caused by a memory shortage for the Java virtual machine, close some applications to free memory and attempt to perform the task again.
3. Refer to “Common problems and limitations” on page 66. If the symptoms of your problem are listed, follow the recommended actions to resolve the problem.
4. Refer to the Host Publisher service page at:

   ```
   http://www.ibm.com/software/network/hostpublisher/service
   ```

   for hints and tips and fixes.
5. Check the Host Publisher logs for error messages that indicate the cause of the problem. Refer to “Logging and tracing components” on page 13 for information about accessing the Host Publisher Server log.

   To route error output for Host Access and Host Publisher Studio to a console window:

   a. Execute from a command line.
   b. Modify the invocation in the .bat file to use java instead of javaw.
   c. Add a showcon option.

   To route error output for Database Access to a console window:

   a. Execute from a command line.
   b. Modify the invocation in the .bat file to use java instead of javaw.

   Refer to “Enabling tracing” on page 55 for information about enabling tracing in the Host Publisher Studio. If output has been scrolled off the console screen, open your own console screen and increase the buffer length using the console properties. Start the application from the console screen using the batch file (.bat) file for the application located in the Studio directory of the Host Publisher installation directory.

6. If the problem still appears when starting WebSphere Application Server or the Host Publisher Server or when accessing a Host Publisher application, check the WebSphere Application Server error logs for information about the problem. For Host Publisher applications, also check the Host Publisher error logs. Refer to "Gathering WebSphere Application Server and Web server problem data" on page 76.
7. If the error log indicates an internal error and that service should be contacted, first try to recreate the problem. Turn on all Host Publisher Server trace options, as well as tracing for the application being run.
8. Contact IBM service to resolve the problem. Refer to "Contacting IBM for service" on page 78.

Common problems and limitations

The following sections document common problems and limitations you may encounter while using Host Publisher. For each symptom, the probable cause and suggested action is provided.

With Host Access and execution of Host Access Integration Objects

Host screen truncated on left at default size

On Korean and Traditional Chinese Windows 95 and NT, some characters may not display in the host session screen, or the wrong characters display and the screen appears corrupted.

This happens because the font used to display characters is not selected correctly, probably because the font properties file is incorrect.

You may solve this problem by installing the latest version of Netscape 4.06; if not, workarounds are available as follows:

Korean: In the \HostPub\Common\JDK\lib directory:

1. Back up the font.properties.ko file.
2. Change the definitions for the Monospaced from:

```
monospaced.0=Gulim,HANGEUL_CHARSET
monospaced.1=Courier New,ANSI_CHARSET
monospaced.2=WingDings,SYMBOL_CHARSET,NEED_CONVERTED
monospaced.3=Symbol,SYMBOL_CHARSET,NEED_CONVERTED
```

   to:

```
monospaced.0=\uad74\ub9bc\uccb4,HANGEUL_CHARSET
monospaced.1=Courier New,ANSI_CHARSET
monospaced.2=WingDings,SYMBOL_CHARSET,NEED_CONVERTED
monospaced.3=Symbol,SYMBOL_CHARSET,NEED_CONVERTED
```

Traditional Chinese: In the \HostPub\Common\JDK\lib directory:

1. Back up the font.properties.zh_TW file.
2. Modify the definitions for the Monospaced as follows:
   a. Change:
      ```
      monospaced.0=\u7d30\u660e\u9ad4,CHINESEBIG5_CHARSET,NEED_CONVERTED
      monospaced.1=Courier New,ANSI_CHARSET
      ```
      to:
      ```
      monospaced.0=Courier New,ANSI_CHARSET
      monospaced.1=\u7d30\u660e\u9ad4,CHINESEBIG5_CHARSET,NEED_CONVERTED
      ```
   b. Change:
      ```
      fontcharset.monospaced.0=sun.io.CharToByteBig5
      ```
      to:
      ```
      fontcharset.monospaced.1=sun.io.CharToByteBig5
      ```
   c. Change:
      ```
      exclusion.monospaced.1=0100-f8ff
      ```
      to:
      ```
      exclusion.monospaced.0=0100-f8ff
      ```

Macros fail in Host Access Integration Objects

Macros can fail for many reasons. Successfully running a macro depends on the application behaving as you expect. The screen flow logic of an application must not change unpredictably over time. Macros must be recorded to be able to process the content of the screens they encounter, and also be precise in identifying the screens processed.

See “Defining a screen” on page 35 for some tips for defining a screen in a macro.

Failures creating Integration Objects

When creating an Integration Object, Host Publisher builds Java source code and compiles it into an executable file. During this process, you may receive a message that the Integration Object could not be created. This message indicates that the compilation of the generated Java source contained errors.

To solve this problem:

- If you are an advanced user, you may find useful information about the cause of the failure in the iofailed.txt file located in the *yourinstalldir*\Studio directory.
- If you have manually modified the host macro files or the .hpi files, use the messages in iofailed.txt to determine if your changes caused the error. If you have not customized any of the Host Publisher files, contact IBM and report the problem.

With Database Access and execution of database Integration Objects

Failures creating Integration Objects

When creating an Integration Object, Host Publisher builds Java source code and compiles it into an executable file. During this process, you may receive a message that the Integration Object could not be created. This message indicates that the compilation of the generated Java source contained errors.

To solve this problem:

- If you are an advanced user, you may find useful information about the cause of the failure in the iofailed.txt file located in the *yourinstalldir*\Studio directory.
- If you have manually modified the host macro files or the .hpi files, use the messages in iofailed.txt to determine if your changes caused the error. If you have not customized any of the Host Publisher files, contact IBM and report the problem.

DBCS limitations with DB2 JDBC driver for OS/390

The DB2 JDBC driver for OS/390 currently supports only database data for the Latin-1 code page (code page 500). DBCS database data will not be properly decoded into unicode by Java applications that use this JDBC interface to gain access to the data.

This is a problem when running the Integration Object from the Database Access application, as well as when running the Integration Object from the Web application on the Server.

Microsoft Access date fields

When you use the JDBC/ODBC bridge to connect to a Microsoft Access database, Date columns do not appear on the Condition panel of the Database Access application. This is because the JDBC/ODBC bridge does not correctly report the column type to the Database Access application.

This is a known JDBC/ODBC bridge driver problem that Host Publisher cannot correct.

Java errors on SQL update

When you connect to an Oracle database using the Oracle8i 8.1.6 JDBC Thin Driver, and you use a variable name for a non-character data type, then click **Run SQL**, you will receive a java.lang.ClassCastException: java.lang.String message. This problem does not occur when the Integration Object is deployed to a server and an application invokes the Integration Object.

Run SQL works correctly for a variable that represents a character data type.

Database interface does not work with Lotus Domino JDBC driver

The Lotus Domino driver for Java Database Connectivity (JDBC) is not supported by Host Publisher. This driver is not JDBC-compliant and is missing some classes required by Host Publisher.

Receive error: No suitable driver found

If you receive a **No suitable driver found** error, make sure you have added runtime.zip and db2java.zip (located in **<drive>\SQLLIB\Java**) to the java.classpath property in the bootstrap.properties file. The bootstrap.properties file is located in WebSphere Application Server's properties directory.

JDBC error: db2jdbc not found error appears in jvm_stderr.txt

This problem has two possible causes:

1. DB2 was not set up correctly.

   Ensure that **SQLLIB\bin** is defined in your system search path.
2. WebSphere Application Server is configured to load the debug Java JVM. The JVM searches for debug libraries owned by the DB2 JDBC driver in the system, but the libraries do not exist in a normal DB2 installation.

   Modify the WebSphere Application Server bootstrap.properties file under the properties subdirectory of the WebSphere Application Server installation directory. Set the java.debug property to **false**.

Connect timeout in a database Integration Object has no effect

The connect timeout value is not implemented by all JDBC drivers.

The JDBC driver will time out while trying to connect to a database, based on the value for the driver. JDBC driver vendors are expected to implement the connect timeout value.

General problems and limitations

General appearance problems

The following list contains general Java appearance or JavaHelp issues.

- **The List and Details buttons that are present in all open object dialogs do not work.**

  This is a known Sun Microsystems problem.
- **The default edit fields and buttons on the Open File dialog appear abnormally large.**

  Correct this problem by resizing the dialog. This will be fixed in a future version of the JRE.
- **You cannot multi-select files in the Open File dialog.**
- **Numerous exceptions are sometimes thrown by the JavaHelp engine when viewing online help.**

These exceptions are benign in nature and can be ignored.

- **Text searches using DBCS character strings do not work.**

  This is a known problem and will be fixed in a future version of JavaHelp.

With the Studio and transferring applications

Problems publishing applications to a Solaris or AIX Host Publisher Server

WebSphere Application Server on Solaris or AIX runs with the user ID and group of *nobody*, and Host Publisher runs with the same permissions. Host Publisher Server staging and production directories must also be owned by the nobody user ID and group. The user ID specified when transferring applications to a server must be the nobody user ID.

When you specify a Sun Solaris or AIX FTP server (Preferences > Options) in Host Publisher Studio, or when configuring a server in the Host Publisher Studio's wizard, specify the nobody user ID to use when transferring application files to that server using FTP.

You will have to use this new user ID when transferring applications to this Host Publisher Server.

With the Server and execution of Integration Objects

Loading of class files

WebSphere Application Server attempts to open all files that exist in directories within its reloadable classpath as if they are JAR files, even if their extension is not .jar. In Host Publisher, the directory placed in the reloadable classpath is the Common directory.

Make sure that the Common directory, and any other directory in WebSphere Application Server's reloadable classpath, are only the original product JAR and class files.

Multiple accesses from the same machine to chained Integration Objects cause problems

Many older browsers use a single HttpSession for Web access when requests are made from the same browser in a different browser window on the same machine. Host Publisher keeps connection instances in the HttpSession object for chained Integration Objects. If you use a browser that uses a single HttpSession for each instance of the browser in a different window, and if you invoke a chained Integration Object, your chained Integration Object may not work correctly.

To avoid this problem, do not use chained Integration Objects in more than one window using the same browser on the same machine.

If you know that your browser uses multiple HttpSessions for each instance of the browser in a different window, you can run chained Integration Objects in each browser window without encountering this problem.

Configuring AS/400 WebSphere Application Server 2.0.3

For improved capacity characteristics, configure WebSphere Application Server 2.0.3 on AS/400 to run (in-process) within the Web server process. You can accomplish this through WebSphere Application Server Administration, or by modifying the WebSphere Application Server bootstrap.properties file with this setting: **ose.mode=in**

Servlet generated by page compilation reports exception: Wrong name

This error occurs when aliases in HTTPD.CONF are created using the same characters but different case; for example, HostPublisher and hostpublisher. Most Web servers do not consider the URL to be case sensitive, so it is possible to specify /HostPublisher/ or /hostpublisher/ in a URL and have it go to the same resource, even if the Web server is configured only with the alias /HostPublisher/. The problem is that the PageCompile depends on a case-sensitive path to the JavaServer Page (JSP). Once a URL is used to access a JSP, you must use the same capitalization within the URL to access that page in the future.

The same problem occurs if you use a different case the second time you reference a file; for example, Tax_Init_Page.jsp and then Tax_init_page.jsp. You must request the page by the *exact* name that was first used, you must delete the information in the pagecompile.

WebSphere Application Server compiles JSPs into Java servlets, then invokes those servlets to render the actual page to a browser. This Java code remembers the exact location of the original page (such as c:\HostPublisher\Server\production\documents\HPAdmin\main.jsp) so that it can reproduce its HTML content. The servlet is rebuilt from the original JSP only if the page is changed (date stamp is updated). If the location of the JSP changes, but its data stamp does not, you receive an internal error. WebSphere Application Server reports the error after trying to process the JSP because it can no longer find the original file. This can happen if you reinstall the same version of Host Publisher Server in a different location.

To correct this problem, remove WebSphere Application Server’s record of the JSP. To do this, remove the corresponding Java and class files from the servlets/pagecompile directory under the WebSphere Application Server installation directory; for example, c:\HostPublisher\WebSphere\AppServer\servlets\pagecompile_HostPublisher_HPA

Out of memory playing JavaServer page

If you receive memory errors while playing a JavaServer Page (JSP), check for REPEAT tags that have non-indexed properties displayed within them; for example:

```
<REPEAT>
<INSERT bean=foo property=bar></INSERT>
</REPEAT>
```

Delete the <REPEAT> and </REPEAT> tags.

If foo.bar is not an indexed property, this JSP will cause an infinite loop that generates output until memory is exhausted. This can happen, for example, if the property was indexed when the page was generated, but the Integration Object was later modified and the property was changed to a single value.

The Web page may not accurately reflect the JavaBean properties if the Integration Object is changed and deployed to the Server without the Web page being recreated using the new Integration Object. Therefore, it seems worthwhile to replace this page with a new Web page created in the Studio, since other changes may also have occurred.

Unable to access Host Publisher directories on AS/400

If you are attempting to access Host Publisher directories from a browser and you receive a message saying that you are not authorized, add the **DirAccess On** statement to the http configuration.

Generic Web browser timeout message is received instead of an expected Host Publisher error message

To avoid this problem, configure the Web server and/or Web browser timeouts (typically 2 minutes) to be greater than the Host On-demand Macro Timeout (default: 60 seconds).

To configure the Host On-demand Macro Timeout, edit the macro. In the first line of the HASCRIPT tag, set the **timeout** field to the desired value (1000 equals 1 second).

If Host Publisher's **Maximum Busy Time Before Disconnection** is set to something other than —1 (never), the sum of this timeout and the Host On-demand Macro timeout should be less than the Web server and Web browser timeouts.

Note: In some cases, the **Maximum Busy Time Before Disconnection** expires and causes the macro to stop; however, a macro timeout error message is generated instead of the busy timeout error message. This will be addressed in a future version of Host Publisher.

Browsers must support cookies when working with Host Publisher on OS/390

When using WebSphere Application Server V1.1 on OS/390, the JavaServer Page (JSP) compiler does not properly process a JSP if the URL for the page is encoded with a session identifier.

URLs are encoded in Host Publisher only when cookies are not supported by browsers connecting to Host Publisher Administration or to applications. Cookies travel between the client and Web server and communicate the session identifier between each request. Without cookie support, the session identifier must be attached to the end of the URL request for the next JSP. This is called URL encoding.

Make sure your browsers and your customers' browsers have cookie support turned on when working with Host Publisher for OS/390.

Recommended Java heap sizes for Solaris Java Virtual Machine (JVM)

WebSphere Application Server recommends that you set the initial Java heap size to between 10% and 25% of the maximum heap size. These values are set in WebSphere Application Server's bootstrap.properties file under properties.mx (maximum heap size) and java.ms (initial heap size).

WebSphere Application Server does ***not*** recommend making these values equal. The ability of the garbage collection algorithm to track allocated memory could result in performance problems if these values are too similar.

Making WebSphere Application Server handle 50 or more requests

The WebSphere Application Server plugin has a hardcoded thread pool size of 50; therefore, even if your Web server has 200 threads and you send it 200 concurrent HTTP requests, WebSphere Application Server will only process 50 of them and queue the other 150.

You can increase the thread pool size by editing a property setting in the WebSphere Application Server configuration file bootstrap.properties. Add the following statement, where **200** is the number of desired requests.

```
ose.outofproc.threadpool.size=200
```

Note: Communication between the Web server and the plugin is handled by a named pipe. On Windows NT, there is a Win32–specific limit of approximately 128 sockets per named pipe. Because of this, even if you raise the thread pool size, you might encounter other operating system-specific limits and WebSphere Application Server might not process the expected number of requests.

PluginTester servlet for debugging WebSphere Application Server problems

WebSphere Application Server 2.03 includes the servlet PluginTester (http://*yourhost*/servlet/PluginTester), which provides useful information about which JAR files have been opened to load classes, as well as other information for debugging problems.

WebSphere Application Server 3.02 includes the servlet showCfg, which is the equivalent of the PluginTester. You can access and configure this servlet at this URL: http://*hostname*/**HostPublisher**/showCfg, where *hostname* is your local server and **HostPublisher** is the Host Publisher alias specified during installation; for example, if you chose **HP** as your alias, you would use http://*hostname*/**HP**/showCfg to access the servlet. The default alias is **HostPublisher**.

Pages are not returned

The error log indicates that Host Publisher ran out of time while trying to set up a connection.

To solve this problem, increase the Connecttimeout parameter in the .connspec file, and the timeout attribute of the <HAScript> tag in all the macros.

Note: The .connspec file is located in the Server/SessionDefs and is in the same location for the Studio. The macros are located in Server/production/poolspecs and Server/production/beans; they are in the same location for the Studio.

.

Shutting down Host Publisher Server

If you shut down Host Publisher Server by shutting down the Web server or WebSphere Application Server, host connections may not be properly cleaned up.

To avoid this problem, stop Host Publisher Server before you stop WebSphere Application Server.

Out of memory error starting 20 sessions

Host Publisher creates one thread per pool, and up to 50 threads during session recovery and shutdown. Host Publisher uses Host On-Demand, which creates a maximum of 100 threads regardless of how many sessions are created.

To avoid this problem, when you configure per process thread limits for a platform, remember that the process is typically used to handle threads of the Web server, as well as threads of WebSphere Application Server.

Problems associated with replicating Host Publisher Servers

You can replicate Host Publisher Server Version 2.2 using one of two methods:

1. Install Host Publisher Server on multiple machines and transfer identical applications to each machine. Then place Network Dispatcher in front of these clones, which are all running the same Host Publisher (Studio-generated) applications. See“Balancing the server’s load” on page 16 for more information on setting up Network Dispatcher. When you are running identical Host Publisher servers and deployment environments on multiple machines with Network Dispatcher in front as a load-balancing mechanism, there is a potential Host Publisher Server replication issue.
2. Install Host Publisher Server on the S/390. Then use the scalable Web server facility on the S/390 (works in conjunction with the S/390 Work Load Manager (WLM)) to create multiple copies of the Web server. Each copy potentially serves different URLs; for example, /HostPublisher/callup/index.html and /HostPublisher/puborder/index.html would go to different Web server instances on the S/390.

You can run multiple Web server clones on the S/390 using WLM; however, only one of them can have WebSphere Application Server V1.1 running. It is therefore not possible to create replicas and clones of Host Publisher server on the S/390 using WLM. WebSphere Application Server V1.2 solves this problem; however, at the time of this writing, WebSphere Application Server V1.2 is not supported by Host Publisher Server. Check Host Publisher’s home page (http://www.ibm.com/software/network/hostpublisher) for up-to-date information on what levels of WebSphere Application Server are supported.

The main points are:

- Everything works fine if the Host Publisher applications deployed.
  - Do not use chained Integration Objects.
  - Do not use user lists for hosts supporting a single logon per user ID.
  - Do not use connection pools where it is important to manage connection pool limits across clones.
- For chained Integration Objects, set the Network Dispatcher sticky port timer to be greater than the HTTP session timer in each WebSphere Application Server configuration to provide HTTP session affinity.
- The administration of each Host Publisher Server must be done from behind Network Dispatcher, not by accessing a common URL through Network Dispatcher. Network Dispatcher cannot guarantee which server you end up managing.
- Global user lists and connection pools are not supported. Each clone creates a pool that can grow as large as the maximum connections configured in the poolspec file. For user lists for hosts supporting a single logon per user

ID, publish a separate userpool file to each clone with different user IDs. If these files have different names, adjust the poolspec file on each clone to point to its respective userpool file.

As an alternative, transfer the same userpool file to each clone; however, make sure there are no duplicate user IDs. Currently, there is no Studio support for ensuring that the user IDs are unique.

Forbidden message received when directory browsing

When a Web server is enabled for directory browsing on a Solaris system running a Java JDK prior to Version 1.2, an attempt to access a Host Publisher application directory using WebSphere Application Server 3.0.2 gets the following message:

```
Forbidden
You don't have permission to access /HostPublisher/dir/ on this server.
```

where *dir* is the Host Publisher application directory.

The problem is that the Solaris JDK prior to 1.2 doesn't create the Host Publisher application directory with the correct permissions. Sun Microsystems is aware of this problem.

The present solution is to perform a **chmod o+r *dir*** operation on the directory. You can then access the Host Publisher application directory.

Gathering WebSphere Application Server and Web server problem data

WebSphere Application Server maintains the following types of error logs:

1. Standard output and standard error output logs from the Java virtual machine
2. Error logs redirected from Web servers
3. Error logs for problems and events discovered by WebSphere Application Server while trying to build or run a servlet

For information on how to obtain problem determination data from WebSphere Application Server, access the WebSphere Application Server Web page at

```
http://www.ibm.com/software/webservers
```

Standard output and standard error output logs

WebSphere Application Server uses a Java virtual machine to run Java servlets, Host Publisher Server, and Integration Objects. WebSphere Application Server creates two files that contain the standard output and standard error output from the Java virtual machine: jvm_stdout.txt and jvm_stderr.txt.

To access these error log files, navigate to the directory path containing the logs for your operating system:

AIX /usr/lpp/IBMWebAS or /usr/WebSphere/AppServer/logs

MVS /usr/lpp/WebSphere/AppServer/logs

NetWare

OS/400

Sun Solaris
/usr/WebSphere/AppServer/logs

Windows NT
(drive):\WebSphere\AppServer\logs

Error logs redirected from Web servers

WebSphere Application Server is configured to use Web servers. WebSphere Application Server redirects error logs from Web servers into the WebSphere Application Server directory structure. The redirected error logs are sequentially numbered. The most recent of these files contain information related to events and errors detected by the Web server. These errors usually indicate unsatisfied page requests or Web server-specific errors. If the WebSphere Application Server directory structure does not contain Web server logs, check your Web server documentation for information on gathering problem data. You might be able to access Web server documentation from the Web server main page by entering the TCP/IP hostname or address as the URL.

To access the redirected Web server error log files (ncf.log), navigate to the directory path containing the logs for your operating system:

AIX /usr/lpp/IBMWebAS or /usr/WebSphere/AppServer/logs

MVS /usr/lpp/WebSphere/AppServer/logs

NetWare
SYS:/WebSphere/AppServer/logs

OS/400
/QIBM/UserData/WebASAdv/default/logs

Sun Solaris
/usr/WebSphere/AppServer/logs

Windows NT
(drive):\WebSphere\AppServer\logs

Error logs for problems and events discovered by WebSphere Application Server

WebSphere Application Server creates three log files to record problems and events discovered while attempting to build or run a servlet. The log files are located in the servlet/servletservice directory. The names of the files are:

- access_log
- error_log
- event_log

Contacting IBM for service

This section lists a number of ways you can reach IBM. Depending on the nature of your problem or concern, we will ask you to be prepared to provide us with information to allow us to serve you better.

If you have a technical problem, please take the time to review and carry out the actions suggested in this chapter. Use your local support personnel before contacting IBM. Only persons with in-depth knowledge of the problem should contact IBM; therefore, support personnel should act as the interface with IBM.

Before you contact IBM, gather the following information:

1. A complete and accurate description of the problem, including whether the problem occurred in the Host Publisher Studio or Host Publisher Server, the task being performed when the error occurred, and the symptoms of the error.
2. If the error occurred in the Host Publisher Studio, a copy of the output in the console window of the application. If the error occurred in the Host Publisher Server, copies of Host Publisher's messages.txt, and copies of jvm_stderr.txt, jvm_stdout.txt, error_log, event_log, and access_log files from WebSphere Application Server.
3. If the error occurred in the Host Publisher Server while attempting to access an application, the files for the application might be needed. You can find these files under the Server\production directory of the Host Publisher installation directory.
4. The level of Host Publisher Studio or Host Publisher Server. To determine the version of the Host Publisher Studio, Select **About...** from the **Help** menu. For Host Publisher Server, the version of the Host Publisher Server is located on the welcome screen for Host Publisher administration.

If you determine that you need to contact IBM, you can do any of the following:

- Consult the **Customer Service and Support Guide**, which is a card contained in the product package.

- Access the Host Publisher Web page at:

  `http://www.ibm.com/software/network/hostpublisher/`
- Access the IBM Personal Software Services Web page, which links to the IBM Software Support Handbook, at:

  `http://ps.software.ibm.com/`

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Appendix A. Server error messages and recovery actions

If you need to contact IBM, see the information in "Contacting IBM for service" on page 78.

(HPS5000) The Admin Servlet initializes the Host Publisher run-time environment.

Explanation
This is an informational message.

User Action
None.

(HPS5001) The value {0} of the attribute {1} in the XML element {2} is not a valid value.

Explanation
In a configuration file, the specified value is not acceptable.

User Action
Publish the application associated with this configuration file again. If the problem continues, review the settings in the Studio for this application.

(HPS5002) The bean filename has a file extension that is not .class, .jar, or .zip.

Explanation
Integration Objects (beans) are expected to be stored in jar or class files, but the specified bean filename does not appear to be a jar or class file.

User Action
If the specified file is a jar or class file, change its name so that it has a .jar or .class file extension, as appropriate. If the file is not a jar or class file, ignore this message.

(HPS5003) There was an unsuccessful attempt to deploy new applications while the server was running.

Explanation
An unexpected error occurred.

User Action
Try again. If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5004) There was an error parsing the application manifest file {0}: {1}

Explanation
A problem occurred while reading the specified application manifest file.

User Action
Look at the error message given, and look up the detailed help for that error message.

(HPS5005) Failed to create a directory named {0}.

Explanation
An error occurred that prevented creating the named directory.

User Action
Ensure the directory path is valid and the user ID that is running the Host Publisher Server has the necessary privileges to create the directory.

(HPS5006) Failed to delete the directory named {0}.

Explanation
An error occurred that prevented deleting the named directory.

User Action
Ensure the directory path is valid and the user ID that is running the Host Publisher Server has the necessary privileges to delete the directory.

(HPS5007) Failed to delete the file named {0}.

Explanation
An error occurred that prevented deleting the named file.

User Action
Ensure the file path is valid and the user ID that is running the Host Publisher Server has the necessary privileges to delete the file.

(HPS5008) There was an error reading the file {0}.

Explanation
Some error occurred while reading the specified file.

User Action
Ensure the specified file exists and the user ID that is running the Host Publisher Server has the necessary privileges to read the file. If the file is part of a published application, publish and deploy it again.

(HPS5009) There was an error writing the file {0}.

Explanation
Some error occurred while writing the specified file.

User Action
Ensure the directory for the file exists, and the user ID that is running the Host Publisher Server has the necessary privileges to create and write files in that directory.

(HPS5010) The installation directory parameter {0} specifies a directory {1} that is not an absolute path.

Explanation
The Host Publisher install directory specified in the WebSphere Application Server servlets.properties file is not a complete unambiguous directory.

User Action
The simplest solution is to install Host Publisher again, taking care to provide a complete installation path. If you prefer, you can use the WebSphere Application Server Administration program to correct the install_dir property of the HPAdmin servlet.

(HPS5011) The installation directory parameter {0} specifies a path {1} that does not exist.

Explanation
The Host Publisher install directory specified in the WebSphere Application Server servlets.properties file does not exist.

User Action
The simplest solution is to install Host Publisher again, taking care to provide a complete installation path. If you prefer, you can use the WebSphere Application Server Administration program to correct the install_dir property of the HPAdmin servlet.

(HPS5012) {0} is not a directory.

Explanation
The named object was expected to be a directory but was not.

User Action
If the message was logged from init, the install_dir property is incorrect for the HPAdmin servlet in WebSphere Application Server. Install Host Publisher again, or correct the servlet property using the WebSphere Application Server Administration program. If the message was logged from

deployApplication, look for other messages that identify the application being deployed, and publish the application again. If the message was logged from start, there is a problem with the Host Publisher installation; install Host Publisher again.

(HPS5013) The XML element {0} is missing the required attribute {1}.

Explanation
Expected item was not found in a configuration file.

User Action
Look for other messages that identify the application. Publish and deploy the application again.

(HPS5014) The initialization parameter {0} is missing. It should specify the installation directory name.

Explanation
WebSphere Application Server is not correctly configured to provide Host Publisher with the name of its installation directory.

User Action
The simplest solution is to install Host Publisher again, taking care to provide a complete installation path. If you prefer, you can use the WebSphere Application Server Administration program to correct the install_dir property of the HPAdmin servlet.

(HPS5015) The XML element {0} is missing either the attribute {1} or the attribute {2}.

Explanation
One of two expected items is missing in a configuration file.

User Action
Look for other messages that identify the application. Publish and deploy the application again.

(HPS5016) The XML element {0} is not a valid element in the file {1}.

Explanation
An unexpected item was found in the specified configuration file.

User Action
Look for other messages that identify the application. Publish the application again.

(HPS5017) Unexpected error {0} occurred. {1}.

Explanation
An unexpected error occurred.

User Action
Try again. If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5018) An exception occurred. {0}.

Explanation
An unexpected error occurred.

User Action
If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5019) There was an XML parse error while reading the file {0} or its DTD file at line {1} in column {2}. The error message is {3}.

Explanation
There was something wrong with a configuration file.

User Action
Look for other messages that identify the application. Publish the application again.

(HPS5020) An exception occurred while parsing XML file {0} : {1}.

Explanation
An unexpected error occurred.

User Action
If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5021) Invalid value of attribute {0} in object {1}. {0} can only be greater than 0.

Explanation
An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action
Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5022) Invalid value of attribute {0} in object {1}. {0} cannot be equal to 0.

Explanation
An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action

Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5023) Invalid value of attribute {0} in object {1}. {0} can only be 0 or higher.

Explanation

An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action

Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5024) Invalid value of attribute {0} in object {1}. {0} can only be -1 or higher.

Explanation

An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action

Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5025) Missing mandatory attribute {0} in object {1}.

Explanation

An expected item was not found in the specified configuration file.

User Action

Look for other messages that identify the application. Publish the application again.

(HPS5026) Missing mandatory attribute {0} in object {1}.

Explanation

An error was found creating the named object. The mandatory named attribute is missing.

User Action

Examine the pool and connection XML files, and ensure you are providing the mandatory attribute for the named object.

(HPS5027) Error creating object {0}. Object {1} not defined.

Explanation

An error was found creating the named object. The second named object did not exist. An error occurred while creating the second named object.

User Action

Examine the log messages before this one and resolve the errors that occurred creating the second named object.

(HPS5028) Object {0} already defined.

Explanation

An attempt was made to create an object already defined.

User Action

Examine the pool and connection XML files, and remove the duplicate occurrence of the named object.

(HPS5029) The server is not running.

Explanation

The Host Publisher Server has not been started, but an operation was attempted that requires the Server to have been started.

User Action

Use the Host Publisher Administration program to start the Host Publisher Server.

(HPS5030) {0} expired. Cannot get a connection from {1}.

Explanation

The maximum waiting time defined for the named connection pool has expired. No connection is available to satisfy the request.

User Action

Based on how often this condition occurs you might want to increase the maximum number of connections defined for the named connection pool.

(HPS5031) Object {0} not defined.

Explanation

The named object is not defined. A Host Publisher Integration Object refers to a connection pool that is not defined on the server.

User Action

Try deploying the Host Publisher application again. This will also deploy the required connection pool definition.

(HPS5032) Cannot load Pool Manager {0}.

Explanation

An unexpected error occurred.

User Action

If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5033) Too little memory. {0} bytes available.

Explanation

There is too little memory left to proceed.

User Action

Free some memory by shutting down applications and freeing disk space. Retry the application request.

(HPS5034) No HttpSession available.

Explanation

An Integration Object tried to acquire an existing connection from the Web session, but there is no web connection. Either the Integration Object that requested the connection or the Integration Object that saved the object are miscoded, or the web connection has timed out, and the web connection (HttpSession) has been deleted.

User Action

Examine the Integration Objects for errors, and determine whether or not the timeout value for the web connection (HttpSession) is too short.

(HPS5035) There is no data source object {0} in HttpSession.

Explanation

An application bean requested a nonexistent data source connection object. The web connection does not contain the requested object. Either the application bean that requested the object or the bean that saved the object are miscoded; or the web connection has timed out, and the object has been removed.

User Action

Examine the application beans for errors, and determine whether or not the timeout values for the connection object and for the web connection (HttpConnection) are too short.

(HPS5036) Attempt to create user pool with invalid type: {0}.

Explanation

An unexpected error occurred.

User Action

If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5037) Attempt to create user pool with invalid name: {0}.

Explanation
An attempt was made to create a user pool with no name.

User Action
Give the user pool a name.

(HPS5038) Error creating object {0}. Object is empty.

Explanation
The named object has no contents. Perhaps the XML file that defines the object has been corrupted. For example, a user pool object has no users.

User Action
Examine the XML files that define the named object. You may need to republish or repair the XML files.

(HPS5039) Cannot get a User from user pool {0}.

Explanation
The named user pool has no free users available.

User Action
Retry the request later, or enlarge the size of the user pool.

(HPS5040) Invalid value of attribute {0} in object {1}. {0} can only be -1 or greater than 0.

Explanation
An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action
Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5043) Internal error, attribute {0} in Conn object is null.

Explanation
This is an internal error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5046) JDBC driver {0} not loadable, received exception: {1}

Explanation
The JDBC driver (Java class) could not be loaded.

User Action
The driver may be missing. The directory where the class resides may not be in WebSphere Application Server or reloadable servlet classpath. Check both. Use the WebSphere Application Server PluginTester servlet, as described in "Contacting IBM for service" on page 78, to check accessibility of the driver.

(HPS5047) JDBC getConnection call failed for URL {0}, received exception: {1}

Explanation
Could not connect to database (URL) specified in the connection specification for this application.

User Action
Verify that the database URL is reachable using DB vendor-supplied tools.

(HPS5048) Call to method {0} not expected for JDBC connections.

Explanation
This is an internal error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5049) The following Session properties are not acceptable to HOD : {0}. Exception received: {1}.

Explanation
This is an unexpected error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5050) Cannot recover connection if HodLogonSpec is null.

Explanation
This connection requires recovery because the user ID and password are characterized as being single use only, and when the connection was returned to the connection pool, the screen was not what was expected. However, this application

has no connect specification, and thus there is no connect or alternate connect macro that can be run to recover the user ID and password.

User Action
: None, if the above scenario is true. If not, you can change the connection specification to define the user IDs to not be single use only, or define connect and alternate connect macros.

(HPS5051) Cannot recover connection if logon or alternate logon macros are null.

Explanation
: This connection requires recovery because the user ID and password are characterized as being single use only, and when the connection was returned to the connection pool, the screen was not what was expected. However, this application has no connect or alternate connect macro that can be run to recover the user ID and password.

User Action
: None, if the above scenario is true. If not, you can change the connection specification to define the user IDs to not be single use only, or define connect and alternate connect macros.

(HPS5052) Cannot set up a connection to the host using the following session properties: {0}

Explanation
: Cannot connect to the host (TN server) specified in the connection specification file for this application.

User Action
: Examine the host name and port number for the TN server that is in the connection specification file for this application. Check that TCP/IP connectivity to the TN server is available using the ping program. If that succeeds, then use an emulator for the right terminal type and check if the TN server is operational.

(HPS5053) Received InterruptedException: {0}

Explanation
: This is an unexpected error.

User Action
: Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5054) Macro state STATE_EMPTY not reached. Current macro state is {0}.

Explanation
This is an unexpected error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5055) Ran out of time while playing HOD macro: Macro file name = {0} Macro = {1}

Explanation
While playing a connect (or disconnect) macro, the connect (or disconnect) timeout specified in the connection specification for this application expired. The timeout value could be too small in the context of how busy the TN server or network traffic is. It is also possible that a screen expected while running the macro did not arrive.

User Action
Increase the timeout value and look at the log to identify the screen.

(HPS5056) Problem encountered while playing macro in file {0} Current state is {0} Initial fragment of the macro: {1}

Explanation
This is an unexpected problem.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5057) PropertyVetoException received from HOD: {0}

Explanation
This is an unexpected error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5058) MacroException received from HOD: {0}

Explanation

The Host On-Demand code threw an exception while executing a connect or disconnect macro. This is an unexpected error.

User Action

Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5059) HOD extract event {0} not expected. Extracts are not permitted in logon or logoff macros.

Explanation

The connect or disconnect macro has extract commands, which are unexpected in such macros and are ignored.

User Action

Examine your macros. Extraction of host screens are not supported while running macros to connect to, or disconnect from, a host.

(HPS5060) Both UserPool and BeanRef are null - cannot provide value for prompt {0} to HOD macro.

Explanation

While trying to supply a parameter to a connect or disconnect macro, the connection manager could not get it from the user pool, nor was there an Integration Object bean that could be interrogated for the value.

User Action

This is an internal error and should be reported to IBM Service.

(HPS5061) Ran out of time while supplying prompt values to HOD macro in file {0}. Exception received: {1}

Explanation

While setting up a connect or disconnect macro for running, the connect (or disconnect) timeout expired. The timeout value could be too small in the context of how busy the TN server or the network traffic is. It is also possible that a screen expected while running the macro did not arrive. The amount of time elapsed in attempting to connect to the host did not allow enough time to run the macro.

User Action

Increase the timeout value and look at the log to identify the screen on which it failed.

(HPS5062) Macro prompt value could not be provided. Exception received: {0}

Explanation
An error was encountered while trying to provide one or more values corresponding to prompt statements in a connect or disconnect macro.

User Action
Examine the prompt statements in the macro (the filename is logged) and check that the value being accessed is in the user pool or is available using a getter method in the Integration Object bean.

(HPS5063) Received the following MacroErrorEvent while playing HOD macro: {0}

Explanation
The execution of a connect or disconnect macro resulted in an error.

User Action
Examine the additional information provided in the log for more information on the cause of the error.

(HPS5064) No get_userid method (with either 0 or 1 argument) in bean.

Explanation
Expected the Integration Object bean to supply the password using a get_userid method since there was no user pool. However, no such method was found.

User Action
Fix the application in the Studio.

(HPS5065) No get_password method in bean.

Explanation
Expected the Integration Object bean to supply the password using a get_password method since there was no user pool. However, no such method was found.

User Action
Fix the application in the Studio.

(HPS5066) No ″getter″ method {0} for parameter {1} in bean {2}.

Explanation
A parameter to be supplied to a prompt statement in a connect or disconnect macro was expected to be found using a getter method in the Integration Object bean, since it was not

found in the user pool entry associated with this connection. However, no getter method was found.

User Action
Correct the application in the Host Publisher Studio.

(HPS5067) Exception received when calling getter method in bean: {0}

Explanation
When attempting to get values from the Integration Object bean to supply macro prompt parameters to a connect or disconnect macro, a bean getter method threw a Java exception. This indicates there was a problem with the Integration Object generated by the Studio.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5068) Exception received: {0}

Explanation
Received an unexpected exception while attempting to supply macro prompt values to a connect or disconnect macro by invoking getter methods of the Integration Object bean.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5069) Attempt to set connection state to {0} is not valid.

Explanation
This is an internal error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5070) Cannot change originating Pool. Old: {0}, New: {1}.

Explanation
This is an internal error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5071) Cannot change connection specification. Old: {0}, New: {1}.

Explanation
This is an internal error.

User Action
Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5072) There is no Pool Manager.

Explanation
An unexpected error occurred.

User Action
If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5073) The Host Publisher encryption key is needed to start the system.

Explanation
The Host Publisher Server was unable to start because a required encryption key was not provided. This case-sensitive encryption key is unique to Host Publisher. The user configures it in the Host Publisher Studio when publishing one or more applications, if at least one of the user ID pools is strongly encrypted.

User Action
Use the Host Publisher Administration program to start the Host Publisher Server and provide the Host Publisher encryption key.

(HPS5074) The Host Publisher encryption key is needed to start the system because user pool {0} is encrypted.

Explanation
The named user pool contains encrypted data that requires the Host Publisher encryption key.

User Action
Supply the required encryption key.

(HPS5075) Received STATE_PLAY_ERROR while playing macro in file {0}.

Explanation
An error was encountered by the Host On-Demand code while executing a connect or disconnect macro.

User Action
The log will contain additional information about the host screen that was seen by the macro engine when the macro

failed, and the macro screen on which the failure occurred. Examine both to determine if this is the screen the macro expected.

(HPS5077) Session is in CONNECTION_ACTIVE state, but not CONNECTION_READY state. A possible reason could be that the TN server port specified does not support the data stream expected. Session properties being used: {0}

Explanation
This can occur if the port specified for the TN server, in the connection specification for this application, does not support the terminal type expected. There could be other reasons too.

User Action
Verify that the port specified does support the expected terminal type (3270, 5250, or VT). If it does, contact your TN server administrator.

(HPS5079) HOD macro {0} has one or more syntax errors.

Explanation
The connect or disconnect macro currently running (identified in the log file) has a syntax error.

User Action
This should not occur unless the macro was manually edited. If you have edited the macro, examine the additional information logged in server.properties to determine the line where an error is being detected, and correct it. If no error can be detected, rerecord the macro using the Host Publisher Studio.

(HPS5080) Invalid value of attribute {0} in object {1}. {0} can only be -1 or greater than {2}.

Explanation
An error was found creating the named object. An invalid value has been specified for the named attribute.

User Action
Examine the pool and connection XML files, and ensure you are providing a valid value for the named attribute.

(HPS5081) The value {0} assigned to the attribute {1} in the file {2} is not valid. It should be a number. A value of {3} will be used.

Explanation
A configuration file contains a value that was supposed to be a number but was not. A default value was used.

User Action
Modify the value in the specified configuration file to be a number.

(HPS5082) Missing property {0} in user pool {1}

Explanation
The password expected by a prompt statement in a connect or disconnect macro is missing. The password was expected to be in the user pool.

User Action
Examine the user pool being used by this application. If you do see entries that define values for the property _password, contact IBM service. Otherwise, the user pool needs to be rebuilt using the Studio.

(HPS5083) Failed to find the file named {0}.

Explanation
The specified file does not exist or could not be read.

User Action
Ensure the specified file exists and the user ID that is running the Host Publisher Server has the necessary privileges to read the file. If the file is part of a published application, publish it again, and deploy it.

(HPS5084) Could not set up a connection.

Explanation
The Host Publisher Server ran out of time while setting up a connection to a backend data source. The backend data source may not be currently available, or your connection timeout in the connection specification file may be too low.

User Action
Check the availability of the backend data source. If it is available, then it is possibly too busy, and you should consider increasing the timeout value in your connection specification and redeploying that application.

(HPS5085) Licenses used ({0}) exceeding licenses purchased ({1}).

Explanation
The number of licenses in use has exceeded the number of licenses purchased.

User Action
Contact your support organization to purchase more licenses.

(HPS5086) Invalid value of attribute {0} in object {2}. {0} cannot be greater than attribute {1}. The value of attribute {1} will be used.

Explanation
An inconsistency was found creating the named object. An invalid value has been specified for the first named attribute. The value of the first named attribute cannot be greater than the value of the second named attribute. The value of the second attribute will be used for the first attribute.

User Action
This message is informational only; no action is required.

(HPS5087) There was an error deploying the application {0}. See previous log messages for details.

Explanation
An error occurred that prevented successfully deploying the application.

User Action
Examine the log messages before this one, and resolve the errors that occurred.

(HPS5088) There was an error copying directory {0} to directory {1} while deploying application {2}.

Explanation
Due to an error, it was not possible to copy the files from one directory to another in order to deploy the application.

User Action
Ensure the source directory exists and is readable and the destination is writable by the user ID that is running the Host Publisher Server. Also ensure that the application has been successfully published.

(HPS5089) There was an error copying file {0} to file {1} while deploying application {2}.

Explanation
An error occurred that prevented copying a file in order to deploy an application.

User Action
Ensure the source file exists and is readable and the destination directory is writable by the user ID that is running the Host Publisher Server. Also ensure that the application has been successfully published.

(HPS5090) System start.

Explanation
This is an informational message.

User Action
The Host Publisher Server has been started. This message is informational only; no action is required.

(HPS5091) System shutdown.

Explanation
This is an informational message.

User Action
The Host Publisher Server has been stopped. This message is informational only; no action is required.

(HPS5092) {0} Security attribute {1} appears to be corrupted, or the Host Publisher encryption key is incorrect. {2}

Explanation
An attempt to decrypt user pool data failed. The supplied encryption key is incorrect or the user pool data has been corrupted.

User Action
Examine the user pool's XML file, and ensure you are supplying the correct encryption key. You may have to republish the user pool with the correct encryption key.

(HPS5093) {0} Invalid schema value for {1} property. {2}

Explanation
An invalid encryption level is specified for the named property.

User Action
Specify a valid encryption level in the schema for the user pool.

(HPS5094) {0} User property {1} conflicts with schema. {2}

Explanation
An unexpected error occurred.

User Action
If the problem persists, turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5095) User {0} not found in user pool {1}.

Explanation
The named user was not found in the named user pool.

User Action

Respecify the name of the user.

(HPS5096) Pool {0} exceeded its capacity. Overflow connection created.

Explanation

The maximum number of connections defined for the named pool has been reached. A non-pooled connection has been created to satisfy the request.

User Action

Based on how often this condition occurs you might want to increase the maximum number of connections defined for the named pool. This message is informational only.

(HPS5097) Pool {0} exceeded its capacity. Request failed.

Explanation

The maximum number of connections defined for the named pool has been reached. A new connection could not be created to satisfy the request.

User Action

Based on how often this condition occurs you might want to increase the maximum number of connections defined for the named pool.

(HPS5098) Ran out of time while playing HOD macro: Macro file name = {0} Macro = {1} Exception received: {2}

Explanation

While supplying parameters corresponding to prompt statements in a connect or disconnect macro, the connecttimeout or disconnecttimeout timer expired.

User Action

The timeout value could be too small in the context of how busy the TN server or the network traffic is. It is also possible that a screen expected while running the macro did not arrive, and the connect (or disconnect) timeout expired.

(HPS5099) Internal error, the following attribute or result is unexpected {0}

Explanation

This is an internal error.

User Action

Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS5100) User ID {0} from UserPool {1} was used to set up a session to host {2}, and must be recovered.

Explanation

During shutdown, a disconnect macro associated with a host connection could not be run because the connection was in use by the Integration Object bean. This message logs which user ID is potentially lost as a result.

User Action

You can manually log on to the host with the above user ID and perform the alternate connect steps to recover the user ID. Otherwise, Host Publisher will perform that recovery on a per connection basis, on restart. The manual recovery will result in more efficient operation after restart.

(HPS5101) The WebSphere servlet engine cannot create an HttpSession object to store the Host Publisher connection with the key {0}.

Explanation

This is a Servlet engine error.

User Action

Turn on all Host Publisher Server traces and try to recreate the condition. Contact IBM service and provide the trace information.

(HPS6002) Conn exception while releasing session to the Session Manager

Explanation

A connection resource could not be appropriately freed. This may cause resources to accumulate and therefore degrade performance.

User Action

Use a resource-usage monitoring tool to check memory, threads, and handle usage. Increasing use of these resources indicates the server will soon lock up. Check preceding error messages for more information.

(HPS6003) Unable to acquire connection from the Session Manager

Explanation

A connection to the data resource could not be acquired.

User Action

Examine the message log for other messages that should give more details about this problem.

(HPS6004) Exception while invoking a property's read method

Explanation

An exception occurred when trying to access a routine that returns the value of a program variable.

User Action

Contact application vendor if problem persists.

(HPS6005) Internal processing error

Explanation

An unexpected error occurred during program execution.

User Action

Contact application vendor if problem persists.

(HPS6006) Introspection exception

Explanation

An Exception occurred while the Integration Object was performing internal introspection.

User Action

Contact application vendor if problem persists.

(HPS6100) Unable to acquire session from the Session Manager

Explanation

A connection to the data source, using a configured pool or connection, could not be established.

User Action

Examine the message log for other messages that should give more details about this problem.

(HPS6101) Internal HOD macro processing error

Explanation

An error occurred during execution of the host macro. The host macro is a step-by-step, predefined interaction between the terminal connection and the host.

User Action

Examine the message log for other messages that should give more details about this problem. Contact application vendor if problem persists.

(HPS6102) Internal macro processing error, extraction coordinates out of bounds

Explanation

An error occurred during execution of the host macro. An attempt was made to extract data from the terminal connection, but the terminal connection coordinates of this data are not defined within this terminal connection.

User Action

Contact application vendor if problem persists.

(HPS6103) Internal macro processing error, macro empty

Explanation

An error occurred during execution of the host macro. The host macro that was to be executed was either not found, or was empty.

User Action

1. Check whether the macro file is empty. If it is, go back to Host Publisher Studio and attempt to recover or record the macro again.
2. If it is not empty, then there might have been a problem during transfer. Use Host Publisher Studio to transfer the application to the Host Publisher Server again. Then redeploy it.
3. If the problem persists, contact IBM service.

(HPS6105) Received unexpected interrupted exception while waiting

Explanation

The synchronization logic of the application received an unexpected error.

User Action

Contact application vendor if problem persists.

(HPS6106) Unable to communicate with the Session Manager

Explanation

The application was not able to establish communications with the program that is responsible for maintaining connections to the data sources.

User Action

Contact application vendor if problem persists.

(HPS6107) Illegal access exception while invoking a property's read method

Explanation

An exception occurred when trying to access a routine that returns the value of a program variable.

User Action

Contact application vendor if problem persists.

(HPS6108) Invocation target exception while invoking a property's read method

Explanation
An exception occurred when trying to access a routine that returns the value of a program variable.

User Action
Contact application vendor if problem persists.

(HPS6109) Exception occurred, cannot retrieve current screen

Explanation
An exception occurred while attempting to gather information about a prior error condition.

User Action
This message can be ignored. Examine the message log for messages about the original error condition.

(HPS6200) Internal error: Unable to build SQL query statement

Explanation
Unable to build a valid SQL query statement from the program parameters.

User Action
Check for prior error messages. Contact application vendor if this error persists.

(HPS6201) Internal error: Unable to build SQL update/insert/delete statement

Explanation
Unable to build a valid SQL update/insert/delete statement from the program parameters.

User Action
Check for prior error messages. Contact application vendor if this error persists.

(HPS6202) Internal error: Unable to locate read method for property

Explanation
An attempt was made to access a method that returns a value for a user or application program defined variable. This method does not exist.

User Action
The application is in error or a problem is present in the Java Virtual Machine environment. If the problem persists, contact the application vendor.

(HPS6203) Internal error: Unable to locate marker for end-of-input-variable in SQL statement

Explanation
Unable to parse the internally generated SQL statement.

User Action
The application is in error or a problem is present in the Java Virtual Machine environment. If the problem persists, contact the application vendor.

(HPS6204) Internal error: Unable to locate ForSQL read method for property

Explanation
An attempt was made to access a method that returns a value for a user or application program defined variable. This method does not exist.

User Action
The application is in error or a problem is present in the Java Virtual Machine environment. If the problem persists, contact the application vendor.

(HPS6205) SQL error while opening statement

Explanation
An error was received for the SQL call: createStatement.

User Action
Investigate the cause of the SQL error. A more detailed description of the SQL exception should be in message log file.

(HPS6206) SQL error while executing query with SQL statement: {0}

Explanation
An error was received for the SQL call: executeQuery.

User Action
Investigate the cause of the SQL error. A more detailed description of the SQL exception should be in message log file.

(HPS6207) SQL error while getting result set

Explanation
An error was received for the SQL call: getString.

User Action
Investigate the cause of the SQL error. A more detailed description of the SQL exception should be in message log file.

(HPS6208) SQL error while closing statement

Explanation

An error was received for the SQL call: closeStatement.

User Action

Investigate the cause of the SQL error. A more detailed description of the SQL exception should be in message log file.

(HPS6209) SQL error while executing SQL statement: {0}

Explanation

An error was received for the SQL call: executeUpdate.

User Action

Investigate the cause of the SQL error. A more detailed description of the SQL exception should be in message log file.

Appendix B. Examples for designing custom Web pages

Use the following examples to design custom Web pages. To cut and paste an example, use the online version of this book, located at Start > Programs > Host Publisher [Server or Studio] > Administrator's and User's Guide.

Note: You can find more code examples on the Host Publisher Web page (http://www-4.ibm.com/software/network/hostpublisher/library/) under Product documentation.

Java access to page parameters

How can I get access to parameters passed to my Web page?

Write a Java scriptlet to call the request.getParameter method and specify the name of the input parameter.

This example gets the values of input parameters tu_in and dupno_in and places them in String variables. They are then displayed as part of HTML h2 headers.

```
<%
   String tu_num = request.getParameter("tu_in");
   String dup_num = request.getParameter("dup_in");
%>

<h2>Val1 is <%=tu_num %></h2>
<h2>Val2 is <%=dup_num %></h2>
```

Redirecting based on Integration Object results

How can I test the output of an Integration Object and call subsequent JavaServer Pages (JSPs) based on the results of my test?

Write a Java scriptlet to get the desired property from the executed Integration Object. Test the property and issue a response.sendRedirect to the appropriate JSP based on the test.

This example calls getTaxmenuid and tests the value to determine the page to redirect to.

```
<%
   String status=Tax_Menu_Init.getTaxmenuid();
   if (status.indexOf("Sign") != -1)
      response.sendRedirect("TaxSignon_Error.jsp");
   if (status.indexOf("Display") != -1)
```

```
      response.sendRedirect("TaxSignon_Inuse.jsp");

   response.sendRedirect("TaxDetails_search.jsp");
%>
```

Invoking Integration Objects based on previous Integration Object results

How can I test the output of an Integration Object and not invoke a subsequent Integration Object if the first Integration Object failed?

Write a Java scriptlet to test a property from the first executed Integration Object. Use an IF statement for the test and wrap the execution of the subsequent bean within the IF.

In this example, the Tax_Details_F12 Integration Object will execute ***only*** if the word DUPLICATE is ***not*** found in variable status.

```
<%
   String status = Tax_Addr.getTaxstatus();
   if (status.indexOf("DUPLICATE") == -1) { %>

      <BEAN NAME="Tax_Details_F12" TYPE="IntegrationObject.Tax_Details_F12"
      <INTROSPECT="yes" CREATE="yes" SCOPE="request">
      </BEAN>
      <% Tax_Details_F12.doHPTransaction(request, response); %>

<%}%>
```

Building dynamic HTML based on Integration Object properties

How can I build dynamic output based on the properties of an Integration Object?

Write a Java scriptlet that will use out.printLn to write output into the HTML stream. Properties of the Integration Object can be embedded in the out.printLn statements.

```
<%
String empno;
try
{
empno = Dbfirst.getDANAPEMPLOYEEEMPNO_(0);
out.printLn("<P>Employee number: <b>" + empno + "</b><p>");
out.printLn(
   Dbfirst.getDANAPEMPLOYEELASTNAME_(0) + "," +
   Dbfirst.getDANAPEMPLOYEEFIRSTNME_(0) + "" +
   Dbfirst.getDANAPEMPLOYEEMIDINIT_(0) + "." +
);

out.printLn("<p>Show <A HREF=\"showfirst.jsp?empno=" + empno +
 "\">Next</A> Employee");
out.printLn("<p>Show <A HREF=\"Deleted.jsp?empno=" + empno +
 "\">Delete</A> Employee");

}
catch(Exception e)
{
```

Validating user input

How can I validate user input from an HTML form?

Use JavaScript to validate user input on an HTML page.

In this example, the function is called by *onSubmit="return padzero(this);"* on the input HTML FORM statement. Function padzero ensures that the input value is numeric and is between 1 and 999999 inclusive.

```
<SCRIPT LANGUAGE="JavaScript1.1">
function padzero(f)
{
   var num = parseInt(f.elements[0].value,10);
   if (num < 1 || num > 999999 || isNaN(num) ||
     num != f.elements[0].value )
    {
     alert("Employee Number not valid ... value must be between" + " " 1 and 999999");
     return false;
    }
   var len = f.elements[0].value.length;
   var zeros = "";
   for (var i = len; i < 6; i++)
    zeros = zeros.concat("0");
   f.elements[0].value = zeros.concat(f.elements[0].value);
   return true;
}
</SCRIPT>
```

Testing for successful database record deletion

How can I test to ensure a database access record delete is successful?

Use a Java Scriptlet to query the Integration Object.

This example calls getHPubNumberOfRowsChanged and tests to see if a row was changed.

```
<%
   DbDelete.setHPubStartPoolName("Db");
   DbDelete.doHPTransaction(request, response);
   int changed;
   changed = DbDelete.getHPubNumberOfRowsChanged();

   if (changed == 0)
{    out.printLn("Error deleting employee number");   }
   else
   if (changed == -1)
   {   out.printLn("No entry found for employee number ");   }
   lse
   {   out.printLn("Successfully deleted employee number");   }
%>
```

Testing for successful database record addition

How can I test to ensure a database access record addition is successful?

Use a Java Scriptlet to query the Integration Object.

This example calls getHPubNumberOfRowsChanged and tests to see if a row was changed. Note that a write of the empno and name into the HTML stream is also in the scriptlet.

```
<%
   DbAddMin.setHPubStartPoolName("Db");
   DbAddMin.doHPTransaction(request, response);
   int changed;
   changed = DbAddMin.getHPubNumberOfRowsChanged();

   if (changed == 0)
   {   out.printLn("Error adding employee number");   }
   else
   {   out.printLn("Successfully added employee number");   }

   out.printLn("<b>" + DbAddMin.getEmpno());
out.printLn("</b> (" + DbAddMin.getLast() + "," + DbAddMin.getFirst()    + " " +
 DbAddMin.getMiddle() + ".).");

   if (changed == 0)
   {   out.printLn("<p>Employee number may alredy be assigned.");   }
%>
```

Passing Java variables to JavaScript function

How can I pass Java variables to JavaScript?

In this example, maxfields is passed to JavaScript function CheckData.

```
<% int maxfields = 99; %>

<SCRIPT Language="Javascript">
function CheckData(formitem, max)
{
   if (formitem.fgn_number..value > max)
      alert("Value is too large. Please re-enter")
}
</SCRIPT>


Sample HTML form to call function:
```

```
<FORM Name="testform" Method="pose"
   onSubmit="return CheckData(this,<%= maxfields %>)" >
  <input type="text" name="fgn_number" size="12" maxlength="12"></p>
</FORM>
```

Using Java to display variables passed into a page

How do I display the variables that are passed into a page?

Use a Java scriptlet and call request.getParameter("<name>"), then use out.printLn. Parameters are sent to pages as Name/Value pairs.

```
<%
   out.printLn("Test Bad");
   String payee_name = "";
   String fgn_input = request.getParameter("fgn_number");
   out.printLn(fgn_input + " ");
   out.printLn(fgn_input.length());
%>
```

Using Java to pad an input value and passing it to an Integration Object

How can I pad input data to the correct length without using JavaScript on the client side?

You can use Java to pad an input variable to the proper length for the host application.

In this example, the dcn_number is padded to be 9 characters.

```
<%
   String dcn_input = request.getParameter("dcn_number");
   String final_dcn_value = "";
   dcn_input = dcn_input.trim();
   for( int i = dcn_input.length(); i < 9; i==)
     final_dcn_value = final_dcn_value.concat("0");
   dcn_input = final_dcn_value + dcn_input;
%>


<BEAN NAME+"Okdhs_dbupdate" TYPE="IntegrationObject.Okdhs_dbupdate"
INTROSPECT="yes" CREATE="yes" SCOPE="reqyest">
</BEAN>
<% Okdhs_dbupdate.setHPubStartPoolName("OKLADHSDB"); %>
<% Okdhs_dbupdate.setDcn_number(dcn_input); %>
<% Okdhs_dbupdate.doHPTransaction(request, response); %>
```

Using a function type passed from a hidden HTML form variable to determine page to execute

How can I use one input form to redirect to different host functions?

Use an onClick method to set a function type in a hidden HTML form variable. This variable can then be tested by Java to determine correct JSP to handle the function. A sample form follows the example.

```
<%
   String function_selected = request.getParameter("funtype");
  try {
     db_pin_value = Okdhs_dbselect.getDMCPEAKEOKLADHSDCN_(0);
     if (db_pin_value.indexOf(dcn_input( != -1)
     {
      if(function_selected.indexOf("CHANGE") == -1)
        newurl = "cfrr_page.jsp?fgn_number=" + fgn_input;
      else
        newurl = "change_pin.jsp?fgn_number=" + fgn_input;
     } else {
      newurl = "cfrr_error.jsp?fgn_number="+
        request.getParameter("fgn_number");
      response.sendRedirect(newurl);
     }
    }
  catch(exception e)
  {
   newurl = "cfrr_error.jsp?fgn_number=" +
      request.getParameter("fgn_number");
   response.sendRedirect(newurl);
  }
%>
```

Sample HTML form:

```
<form name="subform" method="POST"
   ACTION="<%=response.encodeUrl("csml_page.jsp") %>">
   <p align="center"><font face="Comic Sans MS">Please type your identification
   number</font>
   <input type="text" name="fgn_number" size="12" maxlength="12></p>
   <p align="center"><font face="Comic Sans MS">Please enter your PIN</font><input
type="password" name="pin_number" size="9" maxlength="9">>/p>

   <input type="hidden" name="funtype" size="6" maxlength="6">>

   <p align="center"><input type="submit" value="Inquire" name="B1"
onClick="subform.funtype.value='INQUIRE'">&nbsp;&nbsp;
   <input type="reset" value="Clear Form" name="B2">&nbsp;&nbsp;
   <input type="submit" value="Change PIN" onClick="subform.funtype.value='CHANGE'">
   </p></p>
</form>
```

Using Java to prevent blank lines in an HTML table

If I extract more data than is available, how do I prevent blank lines from being added to the HTML table?

Use a Java scriptlet to test the property of the JavaBean. If it is blank, break out of the repeat loop.

```
<TABLE>
...
<REPEAT INDEX=idx1><tr>
<%
  if (Okdsh_cfrr.getCfrr_tableentry_date(idx1).indexOf(" ")!=-1)) break;
%>
<td><INSERT BEAN =Okdsh_cfrr PROPERTY =CFRR_TABLEENTRY_DATE></insert></td>
<d><INSERT BEAN =Okdsh_cfrr PROPERTY=cfrr_tablereceipt_amt></INSERT></td>
</tr>
</REPEAT>
</TABLE
```

Using Java to control display of HTML table based on host results

If there is no data for the table extracted from the host, how can I prevent the display of an empty table?

One way is to also extract a status line from the terminal screen, assuming a status line exists. Then use an IF statement to test the status line to determine if the table should be displayed.

Note: This pertains only to Integration Objects created by the Host Access application.

```
<% String status_value;
   status_value = Spurs_add1.getStatus_line();
   if (status_value.indexOf("VENDOR NOT FOUND") == -1 { // IF "Vendor not found"
   was in status line, do not display the table and instead display a
   "Vendor not found" message.
%>
<h3 align=center>Vendor Listing</h3><TABLE BORDER align=center>
  <TBODY>
    <TR>
      <th>Sequence Number</th>
      <th>Vendor Name</th>
      <th>City/State or Country</th>
    <TR>
    <% String seqnum; %>
    <REPEAT INDEX=idx1>
    <tr>
     <% seqnum = Spurs_addl.getVendor_tableseqnum(idx1);
         if (seqnum.indexOf(" ") <3) break;
    %<
    <td><INSERT BEAN =Spurs_add1 PROPERTY =vendor_tableseqnum></INSERT></td>
    </REPEAT>
   </TBODY>
</TABLE>
< } else { %>
    <h3 align="center">Vendor not found in vendor file.</h3>
<% } %>
```

Determining number of page downs and tabs for making a selection

How does a user select one item when the application contains items on multiple terminal screens?

There are two ways to accomplish this.

1. Use Integration Object chaining and return one screen of items at a time. When the user clicks an item in the list box, use the index of the selected item to select the entry on the screen.
2. Return all the items and display them in a list box. Position the host screen on the first item. Use the following example to calculate the number of page downs and tabs required to get to the proper entry. The value passed in from the previous page is the index of the item in the list box.

Note: This pertains only to Integration Objects created by the Host Access application.

This example assumes there are 9 items per host screen. If your application has a different number per screen, change this value.

```
<HTML>
<BODY>
<!-- Retrieve the parameter passed into this page -->
<!-- This parameter indicates which entry was selected from the inspection list -->
<% int selNum = new Integer(request.getParameter("sel")).intValue(); %>

<!-- Calculate the number of pages downs and tabs -->
<% int numPgDowns =0;
   numPgDowns = selNum / 9;
   int numTabs =0;
   numTabs = selNum%9;
%>

<!-- Build the strings to cause the macro to return to the proper screen -->
<%
   String pgDownString = new String();
   String tabsString = new String();
   //
   //out.printLn("adding "+numPgDowns+" page downs");
   //out.printLn("adding "+numTabs+" tabs");

   for(int i =0; i < numPgDowns; i++)
      pgDownString=pgDownString+"[pagedn]";

   //out.printLn("page down string is "+pgDownString);
   for(int j =0; j < numTabs; j++)
      tabsString += "[tab]";

   //out.printLn("tab string is "+tabsString);

<!-- Now run the macro and retrieve the detail data -->
<BEAN NAME="inspinq2" TYPE="IntegrationObject.Inspinq2" INTROSPECT="yes"
```

```
CREATE="yes"SCOPE="request">
</BEAN>
<% Inspinq2.setOpt(pgDownString+tabsString); %>
<% Inspinq2.doHPTransaction(request, response); %>
```

Changing the action value of a form based on the clicked button

How can I use one form to go to a different page based on the user's selection?

The action value for a form can be changed dynamically.

```
<%
   String invoice_url = response.encodeUrl("WCB_clinic.jsp");
   String cheque_url = response.encodeUrl("WCB_chk_logon.jsp");
%>


<FORM>
   <INPUT type="submit" value="Enter Invoice"
      onClick = "document.forms[0].action = '<
%=invoice_url %>'">

   <INPUT type="submit" value="View Check Information"
      onClick = "document.forms[0].action = '<
%=cheque_url %>'">
```

Using HTTP Session object to pass values

How can I pass values from one page to a subsequent page?

Store the values in an HTTP session object.

```
** First page stores the values.**
   HttpSession ssess = request.getSession(false);
   if (sess!=null) sess.putValue("caregiver_type", WCB_caregiver.getCaregiver_type());
   if (sess!=null) sess.putValue("caregiver_idn", WCB_caregiver.getCaregiver_idn());
%>

** Subsequent page retrieves the values.**
<%
   HttpSession sess = request.getSession(false);
   if (sess != null) WCB_fee_code.setCaregiver-type((String)
sess.getValue("caregiver_type");
   if (sess != null) WCB_fee_code.setCaregiver_idn((String)
sess.getValue("caregiver_idn"));
%>
```

Disabling the browser back button

How can I disable the browser back button?

Place the following JavaScript on every page, including the logon page.

This example uses the browser’s history object to tell the browser to go forward after the page is loaded.

```
<HTML>
<SCRIPT Language="Javescript">
function goHist(a)
{
   history.go(a);   //Go forward one.
}
</SCRIPT>
</head>

<<body onLoad="goHist(1);">
   some HTML ...
</body>
</HTML>
```

Appendix C. 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 that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will

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. 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 IBM's application programming interfaces.

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

Programming interface information

This Administrator's and User's Guide contains information on intended programming interfaces that allow the customer to write programs to obtain the services of Host Publisher.

Appendix D. Trademarks

NetWare, Novell Installation Services (NIS), and NetWare Enterprise Web Server are registered trademarks of Novell in the United States and other countries.

Oracle is a registered trademark of Oracle Corporation.

PC Direct is a trademark of Ziff Communications Company in the United States, other countries, or both and is used by IBM Corporation under license.

Sybase and its corresponding logo are property of Sybase, Inc.

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

SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC. For further information see http://www.setco.org/aboutmark.html.

Index

A

access control list 20
ACL 20
additional information 6
administration, application 17
advantages 4
AIX
  publishing applications to 70
application administration 17
applications
  deleting 17
  deleting unused files in 17
  deploying 11, 17
  listing 17
  transferring to Host Publisher Server 53
applications, composite 49
AS/400 WebSphere, configuring 71

B

balancing the server's load 16

C

certificate, self-signed 62
chain, object 59
chaining
  Integration Object 59
checking a macro 41
class files, loading 70
common problems 66
  appearance 69
  buttons abnormally large 69
  cannot access Host Publisher directories on AS/400 72
  configuring AS/400 WebSphere 71
  Database Access connect timeout 69
  database interface does not work with Lotus Domino JDBC driver 69
  DBCS limitations with DB2 JDBC driver for OS/390 68
  failures creating integration objects 67, 68
  forbidden message received when directory browsing 76
  generic browser timeout message received 72
  Host Access macros fail 67
common problems 66 *(continued)*
  host screen truncated on left at default size 66
  Java errors on SQL update 68
  JavaHelp 69
  JavaHelp engine throws numerous exceptions 69
  JDBC error 69
  List and Details buttons 69
  Microsoft Access date fields 68
  multiple accesses from the same machine to chained Integration Objects cause problems 70
  no suitable driver found error 69
  out of memory error starting 20 sessions 74
  out of memory playing JavaServer page 72
  pages not returned 74
  PluginTester servlet for debugging WebSphere problems 74
  publishing to AIX 70
  publishing to Solaris 70
  replicating Host Publisher servers 75
  securing passwords and other sensitive data in macros 34
  servlet generated by page compilation reports exception: Wrong name 71
  shutting down Host Publisher Server 74
  WebSphere handling 50 or more requests 73
comparing Host Publisher to
  Host On-Demand 4
  WebSphere 3
components 5
composite applications 49
conditionals 36
conditionals, using 37
configuration
  Web server for Host Publisher Server 26
  WebSphere for Host Publisher Server 26
configuring AS/400 WebSphere 71
connect macro
  about 30
connection pools
  creating for Database Access 45
  definition 3
connection requests, multiple 32
connections, defining database 43
connections, defining host 32
contacting IBM 78
cookies, with Host Publisher on OS/390 73
creating a Database Access Integration Object 44

D

data macro
  about 30
data to extract 40
Database Access
  connect timeout 69
  creating an Integration Object 44
  creating connection pools for 45
  retrieving information 44
  wizard 43
database connections, Defining 43
database information, retrieving 44
defining a screen 35
defining host connections 32
deleting applications 17
deleting unused files 17
deploying applications 11, 17
designing custom Web pages 109
disconnect macro
  about 30
Display Terminal, for AS/400 18
documentation 6

E

error events 13
error logs
  problems discovered by WebSphere 78
  redirected from Web server 77
  standard output 76
event
  error 13
  information 13
  warning 13
events, logging 13

F
FAQs 9
frequently asked questions 9

G
gathering problem data 76
getting started 29
global screens 41
green screen
  Display Terminal setup for AS/400 18
groups, managing 20

H
heap sizes, Java 73
Host Access
  checking a macro 41
  defining host connections 32
  generating an Integration Object 42
  identifying data to extract 40
  pooling 32
  recording interactions with a host 35
  using conditionals in 37
  wizard 32
host connections, defining 32
host interactions, recording 35
Host On-Demand
  documentation 6
  Host Publisher and 4
Host Publisher
  advantages of 4
  components of 5
  Host On-Demand and 4
  new in V2.2 6
  overview of 1
  Web site address 1
  WebSphere and 3
Host Publisher Server
  transferring applications to 53
Host Publisher Server Administration
  using 11
Host Publisher Studio
  using 30
  using the wizards 46
how do I...? 29

I
IBM, contacting 78
importing Java objects 53
information, additional 6
information, identifying 40
input variables 36
Integration Object
  chaining 59
  combining output 49
  controlling appearance 49
  creating a Database Access 44
  creating for Database Access 43
  creating for host access 31
  creating Web pages for 45
  generating a Host Access 42
  input also shown as an output 49
  satisfying input 49
  sequencing between non-adjacent pages 52
  sequencing on multiple pages 51
  sequencing on single page 50
  Web publishing 48
Integration Objects
  definition 1

J
Java heap sizes 73
Java objects, Importing 53
JavaHelp
  DBCS character strings, text searches using 69
  exceptions 69
JDBC error 69

L
label
  start 59
  stop 59
Legacy Gateway portal, XML 21
license
  enabling tracking option 12
  managing usage 12
  requests per minute 12
list, access control 20
listing applications 17
lists, user 42
loading class files 70
log file 13
log files, multiple 16
logging
  controlling 14
  log file 13
logs, error 76
looping 36
  graphical representation 39
  inserting in a macro 39

M
macro
  verifying 41
macros
  connect 30
  data 30
  disconnect 30
  Host Access, fail 67
macros, recording 35
managing licenses 12
managing users and groups 20
multiple connection requests 32

N
naming objects 42
Network Dispatcher
  documentation 6
new V2.2 function 6

O
object chain 59
  first 59
  last 60
  middle 60
object chaining
  definition 2
objects, importing Java 53
output logs 76
overview
  Host Publisher 1
Overview
  using Host Publisher 29

P
page, previewing 49
page design example
  building dynamic HTML based on Integration Object properties 110
  changing the action vlaue of a form based on the clicked button 117
  determining number of page downs and tabs for making a selection 116
  disabling the browser back button 117
  invoking Integration Objects based on previous results 110
  Java access to page parameters 109
  passing Java variables to JavaScript function 112
  redirecting based on Integration Object results 109
  testing for successful database record addition 112
  testing for successful database record deletion 111

page design example *(continued)*
- using a function type passed from a hidden HTML form variable 113
- using HTTP session object to pass values 117
- using Java to control display of HTML table based on best results 115
- using Java to display variables passed into a pages 113
- using Java to pad an input value and passing to an Integration Object 113
- using Java to prevent blank lines in an HTML table 114
- validating user input 111

planning and installation guide 7

previewing a page 49

problem determination procedure 65

problems, common 66
- appearance 69
- buttons abnormally large 69
- cannot access Host Publisher directories on AS/400 72
- configuring AS/400 WebSphere 71
- Database Access connect timeout 69
- database interface does not work with Lotus Domino JDBC driver 69
- DBCS limitations with DB2 JDBC driver for OS/390 68
- failures creating integration objects 67, 68
- forbidden message received when directory browsing 76
- generic browser timeout message received 72
- Host Access macros fail 67
- host screen truncated on left at default size 66
- Java errors on SQL update 68
- JavaHelp 69
- JavaHelp engine throws numerous exceptions 69
- JDBC error 69
- List and Details buttons 69
- Microsoft Access date fields 68
- multiple accesses from the same machine to chained IOs cause problems 70

problems, common 66 *(continued)*
- no suitable driver found error 69
- out of memory error starting 20
- sessions 74
- out of memory playing JavaServer page 72
- pages not returned 74
- PluginTester servlet for debugging WebSphere problems 74
- publishing to AIX 70
- publishing to Solaris 70
- replicating Host Publisher servers 75
- securing passwords and other sensitive data in macros 34
- servlet generated by page compilation reports exception: Wrong name 71
- shutting down Host Publisher Server 74
- WebSphere handling 50 or more requests 73

programmer's guide and reference 7

R
readme 8
realm 20
recording a macro 35
requests, multiple connection 32
requests per minute 12
retrieving database information 44

S
screen, defining 35
screens, global 41
Secure Sockets Layer (SSL) 62
security 20
self-signed certificate 62
sequencing
- Integration Objects 50, 51, 52

Server
- balancing load 16
- overview of 6
- using 11

server administration
- using 11

servlets
- hPubPortal 21
- hPubPortalAdmin 21
- hPubPortalData.xml 21
- PluginTester 74
- showCfg 74

showCfg 74

Solaris
- publishing applications to 70

Solaris JVM
- Java heap sizes for 73

SSL
- configuring 62
- self-signed certificate 62

start state label 59
state label, start 59
state label, stop 59
stop state label 59
Studio
- overview of 5

T
technical assistance 78
testing a macro 41
trace file 14
trace files, multiple 16
tracing 13
- controlling 14
- enabling 55
- trace file 14
- xmlLegacyGateway servlet 25

transferring applications to Host Publisher Server 53
troubleshooting 65

U
unused files, deleting 17
URLs 9
user lists 42
users and groups, managing 20
using
- Host Publisher 29
- Host Publisher Studio 30

using conditionals 37

V
variables, input 36
verifying a macro 41

W
Web page 9
Web pages
- design examples 109

Web server
- gathering problem data 76

Web Server
- configuration 26

WebSphere
- application server 20
- configuration 27
- configuring for AS/400 71
- cookies support 73
- documentation 6
- error logs 76

WebSphere *(continued)*
  gathering problem data 76
  handling 50 or more requests 73
  Host Publisher and 3
  Java heap size 73
  loading class files 70
white papers 9
wizard
  Database Access 43
  Host Access 32

X

XML Legacy Gateway portal 21
xmlLegacyGateway servlet 21
  enhancing 25
  tracing 25

Readers' Comments — We'd Like to Hear from You

IBM® WebSphere™ Host Publisher
Administrator's and User's Guide
Version 2.2

Publication No. GC31-8728-01

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name	Address
Company or Organization	
Phone No.	

Readers' Comments — We'd Like to Hear from You
GC31-8728-01

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Information Development
Department CGMD / Bldg 502
P.O. Box 12195
Research Triangle Park, NC
27709-9990

Fold and Tape **Please do not staple** Fold and Tape

GC31-8728-01

Cut or Fold
Along Line

Part Number: CT71MIE

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GC31-8728-01

CT71MIE