Copyright

© 2017 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Business Making Progress, Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, Deliver More Than Expected, Icenium, Kendo UI, Making Software Work Together, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, SequeLink, Sitefinity (and Design), SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. AccelEvent, Analytics360, AppsAlive, AppServer, Arcade, BravePoint, BusinessEdge, DataDirect Spy, DataDirect SupportLink, DevCraft, DigitalFactory, Fiddler, Future Proof, High Performance Integration, JustCode, JustDecompile, JustMock, JustTrace, OpenAccess, ProDataSet, Progress Arcade, Progress Profiles, Progress Results, Progress RFID, Progress Software, ProVision, PSE Pro, SectorAlliance, Sitefinity, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, WebClient, and Who Makes Progress are registered trademarks or service marks of Progress Software Corporation and/or its affiliates or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

Please refer to the Release Notes applicable to the particular Progress product release for any third-party acknowledgements required to be provided in the documentation associated with the Progress product.

The Release Notes can be found in the OpenEdge installation directory and online at: https://community.progress.com/technicalusers/w/openedgegeneral/1329.openedge-product-documentation-overview.aspx.

For the latest documentation updates see OpenEdge Product Documentation on Progress Communities: https://community.progress.com/technicalusers/w/openedgegeneral/1329.openedge-product-documentation-overview.aspx.

March 2017

Last updated with new content: Release 11.7.0

Updated: 2017/03/24
# Table of Contents

Preface ............................................................................................................................................13
  Purpose.........................................................................................................................................13
  Audience.......................................................................................................................................13
  Organization.................................................................................................................................14
  Using ABL documentation............................................................................................................14
    References to ABL compiler and run-time features.................................................................15
    References to ABL data types.................................................................................................15
  Typographical conventions..........................................................................................................15
  Examples of syntax descriptions.................................................................................................16
    Long syntax descriptions split across lines............................................................................18
    Complex syntax descriptions with both required and optional elements............................18
  OpenEdge messages....................................................................................................................19
    Obtaining more information about OpenEdge messages......................................................19

Chapter 1: Introduction ..................................................................................................................21
  OpenEdge Management and OpenEdge Explorer.......................................................................21
    OpenEdge Management............................................................................................................22
    OpenEdge Explorer................................................................................................................22
  Getting started............................................................................................................................23
    Comparing OpenEdge Management and OpenEdge Explorer..............................................23

Chapter 2: Configuring OpenEdge Databases ............................................................................27
  Database configuration and administration..............................................................................28
    Viewing the database log file.................................................................................................29
    Database configuration awareness.........................................................................................29
  Managing databases..................................................................................................................29
  Viewing or modifying database properties............................................................................30
    Database properties..............................................................................................................30
  Viewing the various database configuration properties.........................................................31
    Database configuration properties.........................................................................................31
    Editing the database default configuration...........................................................................42
    Creating a database configuration..........................................................................................43
    Assigning a different default database configuration..........................................................43
  Viewing the server group..........................................................................................................44
    Database server group properties..........................................................................................44
    Editing a database server group.............................................................................................47
    Creating a database server group..........................................................................................47
  Starting or stopping the database.............................................................................................48
Chapter 3: Configuring DataServers..........................................................61
  ODBC DataServer configuration and administration.................................61
    Working with the ODBC DataServer.....................................................62
    Configuring an ODBC DataServer broker.............................................62
    Editing an ODBC DataServer broker configuration............................69
    Creating an ODBC DataServer broker..................................................69
    Deleting an ODBC DataServer broker..................................................69
    Starting or Stopping an ODBC DataServer broker................................70
    Starting an ODBC DataServer broker automatically..........................70
    Viewing the status of an ODBC DataServer broker.............................70
    Viewing the ODBC DataServer log files..............................................72
  Oracle DataServer configuration and administration..................................73
    Working with the Oracle DataServer broker........................................73
    Configuring an Oracle DataServer broker..........................................74
    Editing an Oracle DataServer broker configuration...........................81
    Creating an Oracle DataServer broker...............................................81
    Deleting an Oracle DataServer broker................................................81
    Starting or Stopping an Oracle DataServer broker...............................82
    Starting an Oracle DataServer broker automatically..........................82
    Viewing the status of an Oracle DataServer broker.............................82
    Oracle DataServer broker property Status summary and details............83
    Viewing the Oracle DataServer broker log files..................................84
  MS SQL Server DataServer configuration and administration.....................85
    Working with the MS SQL Server DataServer broker...........................85
    Configuring an MS SQL Server DataServer broker...............................86
    Editing an MS SQL Server DataServer broker configuration..................92
Chapter 4: Configuring NameServers

NameServer configuration and administration
NameServer features
Working with the NameServer
NameServer as an optional client connection request handler
Local and remote NameServers
Configuring a NameServer
Configuring local instances
NameServer properties
Configuring remote instances
Editing a NameServer configuration
Setting NameServer environment variables
Deleting NameServer environment variables
Creating a NameServer
Deleting a NameServer
Configuring fault tolerance and load balancing
Connection-level fault tolerance
Server-level fault tolerance
Load balancing
Location transparency
Starting or Stopping a NameServer
Starting a NameServer automatically
Viewing the status of a NameServer
NameServer status summary properties
NameServer status detail properties
Handling UDP and firewall issues
Viewing the NameServer log file

Chapter 5: Configuring AppServers and AppServer Internet Adapters
Creating an AppServer.............................................................................................................135
Starting and stopping an AppServer........................................................................................135
Adding or trimming AppServer agents.....................................................................................136
Viewing the status of an AppServer.........................................................................................137
Viewing AppServer process details..........................................................................................140
Stopping an AppServer agent..................................................................................................141
Killing an AppServer agent.......................................................................................................141
Deleting an AppServer.............................................................................................................141
Viewing the AppServer log files...............................................................................................142
For additional AppServer information.................................................................................................142
AppServer Internet Adapter configuration and administration............................................................142
Working with the AppServer Internet Adapter..........................................................................143
Configuring an AppServer Internet Adapter.............................................................................143
AppServer Internet Adapter (AIA) properties...........................................................................144
Editing an AppServer Internet Adapter configuration...............................................................149
Creating an AppServer Internet Adapter..................................................................................149
Viewing the AppServer Internet Adapter log file......................................................................150
Enabling or disabling an AppServer Internet Adapter broker...................................................151
Deleting an AppServer Internet Adapter broker.......................................................................151

Chapter 6: Configuring OpenEdge SOAP Web Services.......................153
Configuring and managing SOAP Web services............................................................................153
SOAP-AVM gateway....................................................................................................................154
Web Service Application Server................................................................................................154
Managing SOAP Web services with the WSA..........................................................................154
SOAP Web service session models and the WSA....................................................................155
Managing Web Services Adapter (for OpenEdge SOAP Web Services) instances......................156
Creating a new Web Services Adapter (for OpenEdge SOAP Web Services) instance..............156
Configuring a Web Services Adapter (for OpenEdge SOAP Web Services) instance..............157
Editing a Web Services Adapter (for OpenEdge SOAP Web Services) instance configuration.166
Starting Web Services Adapter (for OpenEdge SOAP Web Services) instances.......................167
Setting SOAP Web service defaults.........................................................................................167
Deploying a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance..........................................................................................................................168
Exporting a SOAP Web service.................................................................................................169
Importing a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance..............................................................................................................................................170
Listing the deployed SOAP Web services for a Web Services Adapter (for OpenEdge SOAP Web Services) instance..................................................................................................................................................170
Viewing the status of a Web Services Adapter (for OpenEdge SOAP Web Services) instance.171
Viewing Web Services Adapter (for OpenEdge SOAP Web Services) instance statistics......171
Viewing statistics for deployed SOAP Web services..................................................................172
Changing Web Services Adapter (for OpenEdge SOAP Web Services) instance run-time properties.............................................................................................................................................................172
Chapter 7: Configuring WebSpeed Messengers and WebSpeed Transaction Servers .................................................................179

WebSpeed Messenger administration .................................................................................................................................179
Configuring a WebSpeed Messenger...............................................................................................................................180
Editing a WebSpeed Messenger configuration ...............................................................................................................184
Viewing the WebSpeed Messenger log file ......................................................................................................................184

WebSpeed Transaction Server administration ..................................................................................................................185
Working with the WebSpeed Transaction Server ...........................................................................................................186
Configuring a WebSpeed Transaction Server ..................................................................................................................186
Editing a WebSpeed Transaction Server configuration ................................................................................................205
Creating a WebSpeed broker ..................................................................................................................................................205
Starting and stopping a WebSpeed broker ........................................................................................................................205
Adding or trimming WebSpeed agents ............................................................................................................................206
Deleting a WebSpeed broker .................................................................................................................................................207

Additional WebSpeed information ..................................................................................................................................208

Chapter 8: Configuring SonicMQ Adapters ........................................................................................................................209

SonicMQ Adapter Broker configuration and administration .................................................................................................209
Working with the SonicMQ Adapter .......................................................................................................................................210
Configuring a SonicMQ Adapter ............................................................................................................................................210
Viewing or modifying SonicMQ Adapter broker properties ................................................................................................211
SonicMQ Adapter broker properties .......................................................................................................................................211
Viewing or modifying SonicMQ Adapter server properties ................................................................................................215
SonicMQ Adapter server properties .......................................................................................................................................216
Viewing or modifying SonicMQ Adapter SSL properties ..................................................................................................217
SonicMQ Adapter SSL properties ..........................................................................................................................................217
Setting or deleting SonicMQ Adapter environment variables ............................................................................................218
Enabling or disabling a SonicMQ Adapter broker ................................................................................................................219
Editing a SonicMQ Adapter configuration ........................................................................................................................219
Creating a SonicMQ Adapter broker ....................................................................................................................................219
Deleting a SonicMQ Adapter broker ....................................................................................................................................220
Starting or Stopping a SonicMQ Adapter ............................................................................................................................220
Starting a SonicMQ Adapter automatically ........................................................................................................................220
Viewing the status of a SonicMQ Adapter broker ................................................................................................................221
Chapter 9: Configuring OE Web Servers

Configuring and managing OE Web Servers

REST Management Agent
OE Web Server
REST Web Applications
Apache Tomcat
Installing a REST Management Agent into a Java AdminServer
Managing REST Web Applications with the OE Web Server

Managing OE Web Servers

Starting and Stopping Tomcat server
Creating an OE Web Server instance
Configuring an OE Web Server instance
Editing an OE Web Server instance configuration
Setting OE Web Server application defaults
Deploying a REST Web application (.war file)
Viewing the status of an OE Web Server instance
Viewing OE Web Server instance statistics
Changing OE Web Server run-time properties
Deleting an OE Web Server instance
Enabling or Disabling an OE Web Server instance
Viewing the OE Web Server log file

Working with REST Web Applications

Enabling or disabling a REST Web application for client access
Viewing REST Web application statistics
Using the REST Web application log file viewer
Updating a deployed REST Web application
SSL support for REST Web Applications
Undeploying a REST Web application

Chapter 10: Configuring Batch Programs

Overview
Creating a Batch Program
Modifying Batch Program configuration
Starting a Batch Program
Stopping a Batch Program
Deleting a Batch Program
Viewing and managing Batch Program processes
Viewing Batch Program log files
Batch Program properties
Batch Program General properties
Batch Program Startup properties
Chapter 11: Configuring third-party Web applications

Overview.............................................................................................................................................269

Extension mechanism in OpenEdge Management..............................................................................270
OpenEdge Management security........................................................................................................270

Requirements for hosting a third-party Web application......................................................................271

Hosting a third-party Web application..................................................................................................271

Implementing single sign-on (SSO)......................................................................................................271
Accessing JSP Standard Tag Libraries................................................................................................272
Loading Java classes............................................................................................................................272

Page contributions for a third-party Web application...........................................................................273

Behavior of OpenEdge Management..................................................................................................273
Registering a contributor....................................................................................................................274

OpenEdge Management Extension framework.....................................................................................274

Accessing OpenEdge Management internals....................................................................................274

Third-party Web application layout....................................................................................................275

OpenEdge Management REST API framework....................................................................................276

Configuring a third-party Web application...........................................................................................277

REST API code example.......................................................................................................................279
URL mapping.......................................................................................................................................280

Exception handling..............................................................................................................................280
REST handler method response............................................................................................................281
Method annotations..............................................................................................................................282

Parameter annotations..........................................................................................................................283

Sample Web application in OpenEdge Management............................................................................286

Using the sample Web application.....................................................................................................286
Accessing the REST API extension.......................................................................................................287
REST API for the sample Web application............................................................................................287
Accessing resources.............................................................................................................................288

Compiling sample Web application using Ant.....................................................................................288
Index.............................................................................................................291
Preface

For details, see the following topics:

- Purpose
- Audience
- Organization
- Using ABL documentation
- Typographical conventions
- Examples of syntax descriptions
- OpenEdge messages

Purpose

This manual describes how to establish property and configuration settings for OpenEdge databases, DataServers (for ODBC, Oracle, and MS SQL Server), NameServers, AppServers, AppServer Internet Adapters, Web Services Adapters (for OpenEdge SOAP Web Services), REST Web Services, WebSpeed Transaction Servers and Messengers, and SonicMQ Adapters in OpenEdge Management and OpenEdge Explorer. In addition, the manual also provides detail about viewing status and log files.

Audience

This manual is intended for OpenEdge Management and OpenEdge Explorer users as well as OpenEdge Management database and system administrators.
Organization

Introduction on page 21
Introduces OpenEdge Management and OpenEdge Explorer and describes their features.

Configuring OpenEdge Databases on page 27
Describes how to establish and modify property and configuration settings for a database, database configuration, or database server group; add a database instance; start and stop a database; use the background writers and watchdog process; view status; view the log file; and delete a database.

Configuring DataServers on page 61
Describes how to establish and modify property and configuration settings for an Oracle or MS SQL Server DataServer; add a DataServer instance; start and stop a DataServer; view status; view the log files; and delete a DataServer.

Configuring NameServers on page 97
Describes how to establish and modify property and configuration settings for a NameServer; add a NameServer instance; start and stop a NameServer; work with fault tolerance and load balancing; view status; view the log files; and delete a NameServer.

Configuring AppServers and AppServer Internet Adapters on page 113
Describes how to establish and modify property and configuration settings for an AppServer or AppServer Internet Adapter; add an AppServer or AppServer Internet Adapter instance; add or trim AppServer agents; start and stop an AppServer; enable or disable an AppServer Internet Adapter; view status; view the log files; and delete an AppServer or AppServer Internet Adapter.

Configuring OpenEdge SOAP Web Services on page 153
Describes how to configure and manage SOAP Web services; manage Web Services Adapter instances (for OpenEdge SOAP Web Services); and work with SOAP Web services.

Configuring WebSpeed Messengers and WebSpeed Transaction Servers on page 179
Describes how to establish and modify property and configuration settings for a WebSpeed Transaction Server or WebSpeed Messenger; create or delete a WebSpeed Transaction Server instance; start and stop a WebSpeed Transaction Server; add or trim WebSpeed Transaction Server agents; view status; and view the log files.

Configuring SonicMQ Adapters on page 209
Describes how to establish and modify property and configuration settings for a SonicMQ Adapter; create or delete a SonicMQ Adapter instance; start and stop a SonicMQ Adapter; view status; and view the log file.

Configuring OE Web Servers on page 225
Describes how to configure and manage OE Web Servers, REST Web services; manage REST Web Services instances; and work with OE WebServers.

Using ABL documentation
OpenEdge provides a special purpose programming language for building business applications. In the documentation, the formal name for this language is ABL (Advanced Business Language). With few exceptions, all keywords of the language appear in all UPPERCASE, using a font that is appropriate to the context. All other alphabetic language content appears in mixed case.
References to ABL compiler and run-time features

ABL is both a compiled and an interpreted language that executes in a run-time engine. The documentation refers to this run-time engine as the **ABL Virtual Machine (AVM)**. When the documentation refers to ABL source code compilation, it specifies **ABL or the compiler** as the actor that manages compile-time features of the language. When the documentation refers to run-time behavior in an executing ABL program, it specifies the **AVM** as the actor that manages the specified run-time behavior in the program.

For example, these sentences refer to the ABL compiler's allowance for parameter passing and the AVM's possible response to that parameter passing at run time: "ABL allows you to pass a dynamic temp-table handle as a static temp-table parameter of a method. However, if at run time the passed dynamic temp-table schema does not match the schema of the static temp-table parameter, the AVM raises an error." The following sentence refers to run-time actions that the AVM can perform using a particular ABL feature: "The ABL socket object handle allows the AVM to connect with other ABL and non-ABL sessions using TCP/IP sockets."

References to ABL data types

ABL provides built-in data types, built-in class data types, and user-defined class data types. References to built-in data types follow these rules:

- Like most other keywords, references to specific built-in data types appear in **all UPPERCASE**, using a font that is appropriate to the context. No uppercase reference ever includes or implies any data type other than itself.
- Wherever **integer** appears, this is a reference to the **INTEGER** or **INT64** data type.
- Wherever **character** appears, this is a reference to the **CHARACTER**, **LONGCHAR**, or **CLOB** data type.
- Wherever **decimal** appears, this is a reference to the **DECIMAL** data type.
- Wherever **numeric** appears, this is a reference to the **INTEGER**, **INT64**, or **DECIMAL** data type.

References to built-in class data types appear in mixed case with initial caps, for example, **Progress.Lang.Object**. References to user-defined class data types appear in mixed case, as specified for a given application example.

Typographical conventions

This documentation uses the following typographical and syntax conventions:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bold</strong></td>
<td>Bold typeface indicates commands or characters the user types, provides emphasis, or the names of user interface elements.</td>
</tr>
<tr>
<td><strong>Italic</strong></td>
<td>Italic typeface indicates the title of a document, or signifies new terms.</td>
</tr>
<tr>
<td>Convention</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SMALL, BOLD CAPITAL LETTERS</td>
<td>Small, bold capital letters indicate OpenEdge key functions and generic keyboard keys; for example, GET and CTRL.</td>
</tr>
<tr>
<td>KEY1+KEY2</td>
<td>A plus sign between key names indicates a simultaneous key sequence: you press and hold down the first key while pressing the second key. For example, CTRL+X.</td>
</tr>
<tr>
<td>KEY1 KEY2</td>
<td>A space between key names indicates a sequential key sequence: you press and release the first key, then press another key. For example, ESCAPE H.</td>
</tr>
</tbody>
</table>

**Syntax:**

<table>
<thead>
<tr>
<th>Fixed width</th>
<th>A fixed-width font is used in syntax, code examples, system output, and file names.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed-width italics</td>
<td>Fixed-width italics indicate variables in syntax.</td>
</tr>
<tr>
<td>Fixed-width bold</td>
<td>Fixed-width bold italic indicates variables in syntax with special emphasis.</td>
</tr>
<tr>
<td>UPPERCASE fixed width</td>
<td>ABL keywords in syntax and code examples are almost always shown in upper case. Although shown in uppercase, you can type ABL keywords in either uppercase or lowercase in a procedure or class.</td>
</tr>
<tr>
<td>Period (.) or colon (:)</td>
<td>All statements except DO, FOR, FUNCTION, PROCEDURE, and REPEAT end with a period. DO, FOR, FUNCTION, PROCEDURE, and REPEAT statements can end with either a period or a colon.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Large brackets indicate the items within them are optional.</td>
</tr>
<tr>
<td>[]</td>
<td>Small brackets are part of ABL.</td>
</tr>
<tr>
<td>{ }</td>
<td>Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.</td>
</tr>
<tr>
<td>{}</td>
<td>Small braces are part of ABL. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.</td>
</tr>
<tr>
<td></td>
<td>A vertical bar indicates a choice.</td>
</tr>
<tr>
<td>...</td>
<td>Ellipses indicate repetition: you can choose one or more of the preceding items.</td>
</tr>
</tbody>
</table>

**Examples of syntax descriptions**

In this example, ACCUM is a keyword, and aggregate and expression are variables:
Syntax

ACCUM aggregate expression

FOR is one of the statements that can end with either a period or a colon, as in this example:

FOR EACH Customer NO-LOCK:
  DISPLAY Customer.Name.
END.

In this example, STREAM stream, UNLESS-HIDDEN, and NO-ERROR are optional:

Syntax

DISPLAY [ STREAM stream ] [ UNLESS-HIDDEN ] [ NO-ERROR ]

In this example, the outer (small) brackets are part of the language, and the inner (large) brackets denote an optional item:

Syntax

INITIAL [ constant [, constant ] ]

A called external procedure must use braces when referencing compile-time arguments passed by a calling procedure, as shown in this example:

Syntax

{ &argument-name }

In this example, EACH, FIRST, and LAST are optional, but you can choose only one of them:

Syntax

PRESELECT [ EACH | FIRST | LAST ] record-phrase

In this example, you must include two expressions, and optionally you can include more. Multiple expressions are separated by commas:
Syntax

MAXIMUM ( expression , expression [ , expression ] ... )

In this example, you must specify MESSAGE and at least one expression or SKIP [ ( n ) ], and any number of additional expression or SKIP [ ( n ) ] is allowed:

Syntax

MESSAGE { expression | SKIP [ ( n ) ] } ... 

In this example, you must specify {include-file, then optionally any number of argument or &argument-name = "argument-value", and then terminate with }:

Syntax

{ include-file
   [ argument | &argument-name = "argument-value" ] ... }

Long syntax descriptions split across lines

Some syntax descriptions are too long to fit on one line. When syntax descriptions are split across multiple lines, groups of optional and groups of required items are kept together in the required order.

In this example, WITH is followed by six optional items:

Syntax

WITH [ ACCUM max-length ] [ expression DOWN ]
[ CENTERED ] [ n COLUMNS ] [ SIDE-LABELS ]
[ STREAM-IO ]

Complex syntax descriptions with both required and optional elements

Some syntax descriptions are too complex to distinguish required and optional elements by bracketing only the optional elements. For such syntax, the descriptions include both braces (for required elements) and brackets (for optional elements).

In this example, ASSIGN requires either one or more field entries or one record. Options available with field or record are grouped with braces and brackets:
**Syntax**

\[
\text{ASSIGN} \quad \left\{ \begin{array}{l}
\quad \left[ \text{FRAME } \text{frame} \right] \quad \left\{ \begin{array}{l}
\quad \text{field} \quad [ = \text{expression} ] \\
\quad \left[ \text{WHEN } \text{expression} \right] \end{array} \right\} \ldots \\
\quad \left\{ \begin{array}{l}
\text{record} \quad [ \text{EXCEPT field} \ldots ] \\
\end{array} \right\}
\end{array} \right\}
\]

**OpenEdge messages**

OpenEdge displays several types of messages to inform you of routine and unusual occurrences:

- **Execution messages** inform you of errors encountered while OpenEdge is running a procedure; for example, if OpenEdge cannot find a record with a specified index field value.

- **Compile messages** inform you of errors found while OpenEdge is reading and analyzing a procedure before running it; for example, if a procedure references a table name that is not defined in the database.

- **Startup messages** inform you of unusual conditions detected while OpenEdge is getting ready to execute; for example, if you entered an invalid startup parameter.

After displaying a message, OpenEdge proceeds in one of several ways:

- Continues execution, subject to the error-processing actions that you specify or that are assumed as part of the procedure. This is the most common action taken after execution messages.

- Returns to the Procedure Editor, so you can correct an error in a procedure. This is the usual action taken after compiler messages.

- Halts processing of a procedure and returns immediately to the Procedure Editor. This does not happen often.

- Terminates the current session.

OpenEdge messages end with a message number in parentheses. In this example, the message number is 200:

```
** Unknown table name table. (200)
```

If you encounter an error that terminates OpenEdge, note the message number before restarting.

**Obtaining more information about OpenEdge messages**

In Windows platforms, use OpenEdge online help to obtain more information about OpenEdge messages. Many OpenEdge tools include the following Help menu options to provide information about messages:

- Choose **Help > Recent Messages** to display detailed descriptions of the most recent OpenEdge message and all other messages returned in the current session.

- Choose **Help > Messages** and then type the message number to display a description of a specific OpenEdge message.

- In the Procedure Editor, press the HELP key or F1.
On UNIX platforms, use the OpenEdge pro command to start a single-user mode character OpenEdge client session and view a brief description of a message by providing its number.

To use the pro command to obtain a message description by message number:

1. Start the Procedure Editor:

```
OpenEdge-install-dir/bin/pro
```

2. Press F3 to access the menu bar, then choose Help > Messages.
3. Type the message number and press ENTER. Details about that message number appear.
4. Press F4 to close the message, press F3 to access the Procedure Editor menu, and choose File > Exit.
Introduction

OpenEdge Management is a system management center that provides visibility, analysis, and proactive monitoring of critical information assets. OpenEdge Management optimizes the availability and performance of OpenEdge-based applications through system monitoring, alerting, and automatic handling of corrective actions. OpenEdge Management empowers Progress Software customers to become more efficient, decrease the cost of managing the OpenEdge environment, and ensure high availability and performance.

OpenEdge Explorer is a subset of OpenEdge Management and runs within the OpenEdge Management console in a Web browser. OpenEdge Explorer enables you to set configuration properties, to start or stop, and to view the status or log files for various OpenEdge resources. OpenEdge Explorer is supported on all the UNIX platforms that support OpenEdge.

For details, see the following topics:

• OpenEdge Management and OpenEdge Explorer
• Getting started

OpenEdge Management and OpenEdge Explorer

You can use OpenEdge Management or OpenEdge Explorer to configure your licensed OpenEdge resources.

Note: The AdminServer referenced in the following sections is a background process that provides a common point of management for the resources managed by OpenEdge Management or OpenEdge Explorer. For more information on the AdminServer, see OpenEdge Getting Started: Installation and Configuration.
OpenEdge Management

OpenEdge Management provides database administrators and systems operations managers with the performance tools and processes required to configure, monitor, diagnose, and manage the OpenEdge environment. This is all done from an easy-to-use graphical interface known as the management console, which runs in a Web browser.

Using OpenEdge Management, you can monitor the following resources running under a local AdminServer or under an AdminServer on a remote machine:

- OpenEdge databases (including those that are enabled for OpenEdge® Replication)
- System resources (CPU, disk, memory, file system)
- File resources
- OpenEdge servers, such as the AppServer, NameServer, WebSpeed® Transaction Server, and supported DataServers (Oracle and MS SQL Server)
- WebSpeed Messengers
- AppServer Internet Adapters, Web Services Adapters (for OpenEdge SOAP Web Services), OE Web Servers, and SonicMQ® Adapters
- TCP-based network services

OpenEdge Management's deep level of monitoring provides information and details about your environment, enabling you to proactively manage operations.

In addition, OpenEdge Management includes the Database Administration Console, which allows you to manage and work with databases.

OpenEdge Explorer

OpenEdge Explorer allows you to set and modify properties from within the OpenEdge Management console. The functionality that you can use for the various resources in OpenEdge Explorer is provided in the following table.

Table 1: OpenEdge Explorer functionality by resource

<table>
<thead>
<tr>
<th>OpenEdge Resource instance</th>
<th>Set config. props</th>
<th>Edit config. props</th>
<th>Start</th>
<th>Stop</th>
<th>Delete</th>
<th>Create</th>
<th>View generic status</th>
<th>View log file</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppServer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>AppServer Internet Adapter</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Database</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Messengers</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>MSS DataServer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NameServer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
OpenEdge Explorer also includes the Database Administration Console, which allows you to manage and work with databases.

### Getting started

This guide provides details about how to use OpenEdge Management and OpenEdge Explorer to configure OpenEdge resources.

For introductory information about using the OpenEdge Management and OpenEdge Explorer console, see *OpenEdge Management and OpenEdge Explorer: Getting Started*.

### Comparing OpenEdge Management and OpenEdge Explorer

OpenEdge Management and OpenEdge Explorer are packaged with OpenEdge and no longer require a separate installation; however, OpenEdge Management still requires a separate license. When OpenEdge Management is licensed, the OpenEdge Explorer functionality is included. If you did not purchase OpenEdge Management, however, its options are grayed out and unavailable in the management console when you are using OpenEdge Explorer.

In general, the settings you can establish and the options you can access when using OpenEdge Management and OpenEdge Explorer are determined both by the platform you are using and the products that you are licensed to use. If you notice that a particular link is dimmed in the OpenEdge Explorer console and not available for modification, it is because that link has no meaning within your operating environment or your licensed configuration.

<table>
<thead>
<tr>
<th>OpenEdge Resource instance</th>
<th>Set config. props</th>
<th>Edit config. props</th>
<th>Start</th>
<th>Stop</th>
<th>Delete</th>
<th>Create</th>
<th>View generic status</th>
<th>View log file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle DataServer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SonicMQ Adapter</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WebSpeed Transaction Server</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Web Services Adapter (for OpenEdge SOAP Web Services)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OE Web Server</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Note: The settings (properties) for most of the resources managed by OpenEdge Management and OpenEdge Explorer (except for OpenEdge databases) are stored in the ubroker.properties file installed with OpenEdge. These properties are stored as name-value pairs. The names of these properties shown in OpenEdge Management and OpenEdge Explorer are long forms of the short names for the same properties stored in the ubroker.properties file. For more information about this file, see OpenEdge Getting Started: Installation and Configuration. For information on property storage for OpenEdge databases, see Database configuration and administration on page 28.

Feature comparison

The following table provides a comparison between the functionality available to you if you have purchased and installed OpenEdge Management (with OpenEdge Explorer), and the functionality available to you if you are working solely with OpenEdge Explorer.

Table 2: OEM and OEE feature comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Available in OpenEdge Explorer</th>
<th>Available in OpenEdge Management (with OpenEdge Explorer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring or modifying default property settings</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Starting/stopping resource instances (if applicable)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Viewing status (if applicable)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Viewing log files</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Adding and configuring new resource instances</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supporting resource and log file monitoring for:</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>• Databases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• DataServers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• AppServers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• NameServers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• WebSpeed Transaction Servers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Web Speed messengers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Network components (TCP, UDP, PING, HTTP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• System components (CPU, disk, file system, memory)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• File components</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SonicMQ Adapters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Web Services Adapters (for OpenEdge SOAP Web Services)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• OE Web Servers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Available in OpenEdge Explorer</td>
<td>Available in OpenEdge Management (with OpenEdge Explorer)</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>Supporting log file monitoring for:</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>• AppServer Internet Adapters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• WebSpeed Messengers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creating jobs for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Creating reports for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Using Library functions, such as rule sets, for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Alerts for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Scheduling and polling for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Graphing for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Trending for supported resources</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SNMP support</td>
<td>No</td>
<td>Yes (with SNMP license)</td>
</tr>
<tr>
<td>Dashboard</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Collections</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Configuring OpenEdge Databases

You can use OpenEdge Management and OpenEdge Explorer to configure and administer the database configuration of an existing OpenEdge RDBMS.

For information about managing and working with databases enabled for multi-tenancy, see *OpenEdge Management and OpenEdge Explorer: Configuring Multi-tenancy*.

For details, see the following topics:

- Database configuration and administration
- Managing databases
- Viewing or modifying database properties
- Viewing the various database configuration properties
- Viewing the server group
- Starting or stopping the database
- Using the background writers and watchdog processes
- Viewing database status
- Deleting a database, database configuration, or database server group
- Configuring an OpenEdge Replication-enabled database
Database configuration and administration

Using OpenEdge Management or OpenEdge Explorer, you can set properties for a database, its various configurations, and its associated server groups. Any configuration settings that you establish for a managed database, database configuration, or database server group in OpenEdge Management or OpenEdge Explorer are automatically reflected in the OpenEdge database Configuration Manager property file (conmgr.properties), and vice versa.

Note: You cannot create a new physical database from within OpenEdge Management or OpenEdge Explorer. For more information about creating a database, see OpenEdge Data Management: Database Administration.

A database recognized and managed by the AdminServer is called a managed database. A database not managed by an AdminServer is called a scripted database. Scripted databases are administered outside the AdminServer using parameter files (.pf) and operating system-dependent scripts.

A managed OpenEdge database starts up by using the default configuration information stored in the conmgr.properties file. The file contains the property settings for all managed databases on the host where OpenEdge is installed.

The conmgr.properties file resides in the properties subdirectory of the OpenEdge install directory. Each database is represented in the file by the following groups:

- **Database** — The group that describes the physical database. There is one database group for each physical database.

- **Configuration** — One or more groups that describe the database arguments and server groups for an active instance of the database. Only one database configuration can be active at one time. The active configuration is the default configuration identified in the server group.

- **ServerGroup** — One or more groups that describe brokering behavior. The first server group within a configuration defines the primary broker and each subsequent group defines a secondary broker. The properties in a server group define the arguments that are specific to that primary or secondary broker.

You can edit the conmgr.properties file manually; any changes that you make to it are automatically reflected in OpenEdge Management or OpenEdge Explorer.

Note: Although you can make manual edits to the conmgr.properties file, Progress Software recommends that you use the mergeprop utility, OpenEdge Management, or OpenEdge Explorer to make property changes. For more information about the mergeprop utility, see OpenEdge Getting Started: Installation and Configuration.

From OpenEdge Management or OpenEdge Explorer, you can perform the following database-related actions:

- Add an existing database
- Create a new database configuration
- Edit an existing or new database configuration
- Use the default server group
- Create a new database server group
- Edit a new or existing database server group
- Start a database
- Start a database automatically
Managing databases

- Stop a database
- Use the background writers and watchdog processes
- View the status of a database
- Delete a database
- Delete a database configuration (provided there is more than one)
- Delete a database server group (provided there is more than one)
- Configure the properties of an OpenEdge Replication-enabled database

You can also manage and work with databases enabled for multi-tenancy. For details, see *OpenEdge Management and OpenEdge Explorer: Configuring Multi-tenancy.*

Viewing the database log file

You can use the log file viewer to review the database log file. For more information, see the section about using the log file viewer in *OpenEdge Management: Database Management.*

Database configuration awareness

You can use OpenEdge Management or OpenEdge Explorer to establish database settings and options. These settings and options are determined both by the platform you are using and the products that you are licensed to use. If a property is not applicable to the database, its setting is either ignored or, when appropriate, an error or warning is issued.

Managing databases

You can use OpenEdge Management or OpenEdge Explorer to manage your database. You cannot physically create a database in the management console; however, you can create the definition of a database so that you can configure, start, stop, or view status for the primary broker and all auxiliary processes associated with that broker, and view status for the log file for the database.

Database configuration information is stored in the `conmgr.properties` file. For a complete list of how the database properties map to the command-line `.pf` arguments, see the `conmgr.properties.README`. You can locate the `README` file in the `OpenEdge-install\properties` directory.

A database recognized and managed by the AdminServer is called a *managed* database. A database not managed by an AdminServer is called a *scripted* database. Scripted databases are administered outside the AdminServer using parameter files (.pf) and operating system-dependent scripts. You can migrate a scripted database to a managed database.

For details about adding a managed or a scripted database, see *OpenEdge Management: Database Management.*

For details about managing and working with databases enabled for multi-tenancy, see *OpenEdge Management and OpenEdge Explorer: Configuring Multi-tenancy.*
Viewing or modifying database properties

Each database definition has a set of properties. You can view and modify the properties if you want to establish new values.

To view or modify the database properties:

1. Click Resources from the OpenEdge Management console menu. The Resources page appears with all the resources managed by your console, in a grid list.
2. From the resources list, select a database whose default configuration properties you want to modify.

   **Note:** You can also search for a specific database by entering the database name in the Search field or selecting Database from the resources type drop-down.

3. Click the Edit icon. The Database details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The Database Configuration page appears. From the Database Configuration page, you can do the following:
   - View a read-only display of the database properties as described in **Database properties** on page 30.
   - Click **Edit** to modify the database properties. When you finish, click **Save**.
   - Click the relevant **Configuration** and **Server Group** link to access the database default configuration properties or the database server group properties. For details about these properties, see **Viewing the various database configuration properties** on page 31 and **Viewing the server group** on page 44.
   - Set OpenEdge Replication Server or Agent properties, if the database is replication-enabled or if an associated `<db-name>.repl.properties` file exists in the physical database directory. For details, see **Configuring an OpenEdge Replication-enabled database** on page 52.

Database properties

Table 3: Database properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database name</td>
<td>Identifies the full path to the database.</td>
</tr>
<tr>
<td>Autostart</td>
<td>Indicates whether the database starts automatically when the AdminServer starts.</td>
</tr>
<tr>
<td>Cluster enabled</td>
<td>Indicates whether you have already configured the database as cluster-enabled (by using the PROUTIL DESCRIBE command). This allows the database to register with the cluster manager at start time.</td>
</tr>
<tr>
<td>Default configuration</td>
<td>Provides the name of the default configuration that is used when the database starts.</td>
</tr>
</tbody>
</table>
Viewing the various database configuration properties

OpenEdge Management and OpenEdge Explorer provide a predefined default database configuration. Once you add a database instance, you can use the property settings established in the default configuration, you can modify the default settings, or you can create and modify additional configurations.

Note that only one database default configuration can be selected for a database at a time.

To view the database default configuration properties:

1. Click **Resources** from the OpenEdge Management console menu. The **Resources** page appears with all the resources managed by your console, in a grid list.

2. From the resources list, select a database whose default configuration properties you want to view.

   **Note:** You can also search for a specific database by entering the database name in the **Search** field or selecting Database from the resources type drop-down.

3. Click the **Edit** icon. The **Database** details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.

5. Click **configuration.<databaseName>.defaultconfiguration**. The **Database Configuration** page appears, displaying the default configuration properties in the following categories: General, Background Writers, After Image Management, International, Advanced, Statistics Collection Configuration, SQL-92 Configuration, SSL Configuration, and Agent. These properties are described in Database configuration properties on page 31.

From the **Database Configuration** page, you can do the following:

- View a read-only display of the database default configuration properties as described in the tables that follow this procedure.

- Click **Edit** to modify the default database properties. For details, see Editing the database default configuration on page 42.

- Click the relevant link to access the associated database server group properties. For details about these properties, see Viewing the server group on page 44.

Database configuration properties

The following sections describe the database configuration properties.

**General database configuration properties**

**Table 4: General database configuration properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocks in database buffers (primary)</td>
<td>The number of blocks in the database buffers. The optimum value for this property depends on your application. The default value is 0.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Alternate buffer pool** | The number of blocks in the alternate buffer pool. The optimum value depends on your application. If you specify **Alternate buffer pool** blocks for a Replication target database, the property setting is ignored.  
**Note:** Alternate buffer pool functionality requires an Enterprise database license. For more information about alternate buffer pools, see *OpenEdge Data Management: Database Administration*. |
| **Hash table entries**    | The number of hash table entries to use for the buffer pool. The default value is 0.  
**Note:** Do not edit this property unless directed to do so by Progress Software Technical Support.                                                                 |
| **Lock table entries**    | The number of entries in the record locking table. If you specify a value that is not a multiple of 32, the value is rounded to the next highest multiple of 32. Each record that is accessed by any user takes one entry.  
The default value is 8192.                                                                 |
| **Tenant Lock Table Governors(%)** | Specifies the percentage of the lock table that any tenant can occupy. The valid values are 1 through 100. The default value is 100.  
**Note:** This parameter applies to only multi-tenant databases. In addition, this applies to only the default tenant and regular tenant. The super tenants do not use the governors parameters.  
You can also monitor this parameter using PROMON utility. |
| **Max users**             | The maximum number of users on the system for the database. After the maximum number of users have connected to the database, additional user connect attempts are rejected.  
The default value is 20.                                                                 |
| **Tenant Max. User Governor** | Specifies the maximum number of users that may login for any tenant. The valid values are 1 through the maximum number of users (32000).  
The default value is the maximum number of users (-n).  
**Note:** This parameter applies to only multi-tenant databases. In addition, this applies to only the default tenant and regular tenant. The super tenants do not use the governors parameters.  
You can also monitor this parameter using the PROMON utility. |
The maximum number of remote client servers that can be started by the broker process.

Use the **Max servers** property to limit the number of remote user servers that can be started on the system. The performance trade-off to consider is swapping overhead for many servers versus overloading (slowing down) a server with too many clients.

**Note:** This parameter pertains only to the OpenEdge® Workgroup RDBMS and the OpenEdge® Enterprise RDBMS. In addition, this parameter applies only to databases that are accessed from remote network nodes.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max servers</td>
<td>The maximum number of remote client servers that can be started by the broker process. Use the <strong>Max servers</strong> property to limit the number of remote user servers that can be started on the system. The performance trade-off to consider is swapping overhead for many servers versus overloading (slowing down) a server with too many clients.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum private database buffers per user</td>
<td>Controls the number of private buffers a connection can request through the Private Read-only Buffers (-Bp) parameter.</td>
</tr>
<tr>
<td>Direct I/O</td>
<td>Enables the Direct I/O property, which opens all files in unbuffered mode. Direct I/O enables the database to use an I/O technique that bypasses the operating system buffer pool and transfers data directly from a buffer to disk. This technique has some advantages over buffered reads and writes, such as avoiding the overhead of maintaining the operating system buffer pool, and eliminating competition for operating system buffers between the database and other programs. The operating system buffer-pool algorithms are designed for efficient sequential file access; the OpenEdge buffer-pool algorithms are more efficient for access to an OpenEdge RDBMS. The <strong>Direct I/O</strong> check box is cleared by default.</td>
</tr>
<tr>
<td>Other arguments</td>
<td>Any other arguments for which there is no defined property available.</td>
</tr>
</tbody>
</table>

### Background writers database configuration properties

**Table 5: Background writers database configuration properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watchdog process</td>
<td>Start the Watchdog process automatically when the database starts. The check box is selected by default.</td>
</tr>
</tbody>
</table>
## Asynchronous page writers

The number of Asynchronous Page Writer (APW) processes to start.

A database can have between zero and nine APWs running simultaneously. The optimal number is dependent on your application and environment.

To start, use one page writer for each disk where the database resides. If this is insufficient, add more.

For an application that performs many updates, start one APW for each disk containing your database, and add one additional APW. Applications that perform fewer changes to a database require fewer APWs. If an application performs no updates, no page writers are required.

The default value is 1.

## Before image process

Start a Before-imaging process automatically when the database starts.

The check box is selected by default.

## Before image stall

Send a message to the log file when the recovery log threshold is reached, preventing emergency shutdown. (For details about the `-bistall` startup parameter, see *OpenEdge Data Management: Database Administration*.)

The check box is cleared by default.

## Before image buffers

The number of before-image buffers to use.

This property is useful only when running the before-image writer (BIW). The BIW continually writes the filled before-image buffers to disk, making the buffers available to other client and server processes. Without a BIW writing the buffers, any gain from increasing the number of buffers is negligible.

The default value is 5.

## Before image threshold

The maximum size to which BI files can grow as a percentage of the largest possible recovery log file. (For details about the `-bithold` startup parameter, see *OpenEdge Data Management: Database Administration.*)

The default value is 3.

## Before image delay writes

A positive value that you specify to delay OpenEdge from synchronously writing out to disk the last before-image (BI) file records at the end of each transaction.

On UNIX systems using shared memory, the value also specifies the interval at which the broker process wakes up to make sure all BI file changes have been written to disk.

The default value is 3.
### Property | Description
--- | ---
**Before image cluster age** | The number of seconds before OpenEdge reuses a before-image cluster.  
The default value is 60.
**Group delay** | The number of milliseconds a transaction waits before committing.
**After image process** | Start an After-imaging process automatically when the database starts.  
The check box is cleared by default.
**After image stall** | Suspend database activity if all the after-image (AI) files are filled.  
By default, if all the AI files are filled and the database cannot switch to an empty one, the database shuts down. With AI stall, you can suspend database activity, back up the full after-image file, and mark it as empty.  
The check box is cleared by default.
**After image buffers** | The number of after-image buffers to use.  
This parameter is useful only when running the after-image writer (AIW) process because the AIW writes the filled after-image buffers to disk, making the buffers available to other client and server processes.  
The default value is 5.

### After Image Management database configuration properties

Table 6: After Image Management database configuration properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>After-image archival directory list</strong></td>
<td>The directories where the AI File Management utility writes the archived after-image files.</td>
</tr>
<tr>
<td><strong>Create after-image archival directories</strong></td>
<td>Directs the AI File Management utility to create the specified directories.</td>
</tr>
</tbody>
</table>
| **After-image archival interval** | Specifies on-demand mode archiving or the extent switch interval for timed-mode archiving.  
The minimum value is 120, and the maximum value is 86400. The default value is 120. |
## International database configuration properties

Table 7: International database configuration properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Collation table (-cpcoll)** | A collation table in the `convmap.cp` file that you use for collation rules.  
The collation table you specify is used with the code page OpenEdge uses in memory, which is specified by the Internal Code Page (`-cpinternal`) parameter.  
The default value is BASIC. |
| **Case table name (-cpcase)** | A case table in the `convmap.cp` file that you use for case rules.  
Case rules are used by the 4GL CAPS and LC functions and by the exclamation point (!) formatting character, which you use to tell OpenEdge to convert all characters to uppercase during input.  
The case table you specify is used to establish case rules for the code page used in OpenEdge's memory. The memory code page is set with the Internal Code Page (`-cpinternal`) parameter.  
The default value is BASIC. |
| **Conversion map (-convmap)** | The conversion map file that OpenEdge uses for code page conversions, collation orders, and case conversions.  
By default, OpenEdge uses `OpenEdge-install-dir\convmap.cp` in Windows and `OpenEdge-install-dir/convmap.cp` on UNIX. |
| **Server code page (-cpinternal)** | The code page that OpenEdge uses in memory.  
For GUI clients, the `-cpinternal` code page should be the same code page the operating system uses.  
If you do not specify a value, OpenEdge uses the iso8859-1 code page by default. |
| **Log code page (-cplog)** | The code page that the database uses when it writes to a log file.  
By default, the database uses the internal code page.  
Choose a code page that agrees with your operating system and its devices. If you have a Unicode database, you might want to specify the log character set as Unicode so information is properly logged.  
If you do not specify a value, OpenEdge uses the iso8859-1 code page by default. |
Advanced database configuration properties

Table 8: Advanced database configuration properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database service communication area size</td>
<td>The value (which corresponds to the -pica command-line option) that sets the size of the database service communications area. This area is used for storage of after-image block write notifications intended for OpenEdge Replication. The minimum value is 4 and the maximum value is 8192; the value is entered in Kb units.</td>
</tr>
<tr>
<td>Windows Event Level</td>
<td>The level of information written to the Windows Application Event Log. Choose from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — No events are written to the Event Log.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Brief</strong> — Error and Warning messages are written to the Event Log.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Normal</strong> — Error and Warning messages are written to the Event Log along with any message that is normally written to the log file (.lg). This is the default value.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Full</strong> — Error, Warning, and Informational messages are written to the Event Log along with any messages generated by the Message Statement.</td>
</tr>
<tr>
<td>Minimum heap size</td>
<td>The minimum value is 0, and the maximum value is 65535. The minimum size for temporary storage.</td>
</tr>
<tr>
<td>Lock table hash table size</td>
<td>The size of the hash table that controls access to the lock table.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: Progress Software Corporation recommends changing the value of this property only after contacting Technical Support.</td>
</tr>
<tr>
<td>Maximum number of database areas</td>
<td>The highest area number available for use during the time the database is online.</td>
</tr>
<tr>
<td>Nap interval</td>
<td>The minimum nap time in the spin lock algorithm. The default value is 1.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Nap time increment</strong></td>
<td>The nap increment value. (Enterprise database only)</td>
</tr>
<tr>
<td><strong>Note:</strong> Do not edit this property</td>
<td>unless directed to do so by Progress Software Technical Support.</td>
</tr>
<tr>
<td><strong>Nap max interval</strong></td>
<td>The maximum nap time in the spin lock algorithm before the nap time resets to the minimum. The default value is 1.</td>
</tr>
<tr>
<td><strong>Nap time steps between nap time</strong></td>
<td>The steps between the nap increment. (Enterprise database only)</td>
</tr>
<tr>
<td><strong>Note:</strong> Do not edit this property</td>
<td>unless directed to do so by Progress Software Technical Support.</td>
</tr>
<tr>
<td><strong>Page writer max buffers</strong></td>
<td>The maximum number of modified database buffers to write to disk during a buffer scan cycle. The default value is 25.</td>
</tr>
<tr>
<td><strong>Page writer queue delay</strong></td>
<td>The number of milliseconds between scans of the page writer queue. The default value is 100.</td>
</tr>
<tr>
<td><strong>Page writer queue minimum</strong></td>
<td>The minimum number of buffers required on the page writer queue before an APW writes them to disk. The default value is 1.</td>
</tr>
<tr>
<td><strong>Page writer scan</strong></td>
<td>The number of buffers that an APW should scan during a buffer scan cycle. The default value is 1.</td>
</tr>
<tr>
<td><strong>Page writer scan delay</strong></td>
<td>The number of seconds of delay between each scan of the database buffers by the APW. The default value is 1.</td>
</tr>
<tr>
<td><strong>Pending connection timeout</strong></td>
<td>The amount of time a client has to connect to a server before the broker clears the client's reservation.</td>
</tr>
</tbody>
</table>
### Viewing the various database configuration properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Semaphore sets</strong></td>
<td>The number of semaphore sets available to the database broker.</td>
</tr>
<tr>
<td></td>
<td>On single-processor systems, OpenEdge uses semaphores to synchronize the activities of server and self-service client processes that are connected to a database. By default, each database has an array of semaphores, one for each user or server. Each process uses its semaphore when it must wait for a shared resource. Semaphores are not used for single-user sessions or for client sessions connecting to a remote database on a server system. By default, this value is set to 1. When more than 1,000 users connect to a single database, there might be high contention for the semaphore set. If there is a lot of semaphore contention on a system, using multiple semaphore sets helps alleviate this contention and improves performance on high user counts.</td>
</tr>
<tr>
<td><strong>Shared memory overflow size</strong></td>
<td>In kilobytes, the size of the shared-memory overflow area. This parameter replaces the default value of the shared-memory overflow area; it does not increase it. The overflow area is appended to the shared-memory area. The default value is 0.</td>
</tr>
<tr>
<td><strong>Pin shared memory segments</strong></td>
<td>Prevents the database engine from swapping shared memory contents to disk.</td>
</tr>
<tr>
<td><strong>Maximum shared-memory segment size</strong></td>
<td>The maximum number of bytes, specified in MB or GB, in a shared memory segment.</td>
</tr>
</tbody>
</table>

[OpenEdge Management and Explorer: Configuration](39)
Spin lock retries

The number of times a process tries to acquire a latch before pausing.
If the process cannot acquire the resource’s latch, it continues the attempt. This iterative process is called spinning.
By default, this value is set to 0.
If the value of this property is greater than zero, a spin lock algorithm is used for shared-memory data structures. When a process has to lock a shared-memory structure, the process attempts to acquire the latch for that structure up to the specified number of times.
If the process has not acquired the latch in the specified number of attempts, then the process pauses, or naps. The length of the pause increases gradually if the process repeatedly fails to acquire a latch. After the allotted nap time, the process wakes up and attempts to acquire the latch again. If it fails to acquire the latch, it retries up to the number of attempts specified.

Storage object cache size

The size of the object cache for all database objects.

TXE commit lock skip limit

Sets the maximum SHARE/UPDATE locks that can skip the queued COMMIT locks.

Note: Do not edit this property unless directed to do so by Progress Software Technical Support.

User MUX latches

A value that controls the granularity of access to large database resources in shared memory.

Note: Do not edit this property unless directed to do so by Progress Software Technical Support.

Statistics Collection Configuration database properties

Table 9: Statistics Collection Configuration database properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base index number</td>
<td>The starting index number in the range of indexes for which you want to track access statistics</td>
</tr>
</tbody>
</table>
The number of indexes for which you want to track access statistics

The starting table number in the range of tables for which you want to track access statistics

The number of tables for which you want to track access statistics

SQL-92 Configuration database properties

Table 10: SQL-92 Configuration database properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java classpath</td>
<td>The pathname of the classpath</td>
</tr>
<tr>
<td>Number of open SQL cursors</td>
<td>The number of open cursors per connection</td>
</tr>
<tr>
<td>SQL stack size</td>
<td>The size, in MB, of the SQL Stack</td>
</tr>
<tr>
<td>SQL statement cache size</td>
<td>The number of statements allowed in the SQL statement cache</td>
</tr>
<tr>
<td>SQL sorting memory</td>
<td>The size of the temporary table buffer in memory</td>
</tr>
<tr>
<td>SQL sorting on disk</td>
<td>The size of the temporary table for backup storage</td>
</tr>
<tr>
<td>SQL temporary data page size</td>
<td>The size of the temporary table data page</td>
</tr>
<tr>
<td>Maximum number of JTA transactions</td>
<td>The number of JTA transactions simultaneously allowed</td>
</tr>
</tbody>
</table>

SSL Configuration database properties

Table 11: SSL Configuration database properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL for remote connections</td>
<td>Specifies that all connections to this server socket must use SSL.</td>
</tr>
<tr>
<td>Key alias name</td>
<td>Sets the alias name within the keystore of the private key and digital certificate entry to use to authenticate all connections to this server socket. If the key alias name is not specified, the server socket uses the default_server server certificate alias.</td>
</tr>
</tbody>
</table>
**Key alias password**
Sets the password to use for accessing the private key and digital certificate.
You must specify a password when you specify the `-keyalias` option, and the password must be encrypted. You must specify the password as an encrypted value that you can obtain using the `genpassword` utility located in the `bin` directory of your OpenEdge installation.
If you use the default_server server certificate, it also has a default password that you do not need to specify.

**Disable SSL session cache**
Disables session caching. (Session caching allows a client to reuse a previously established session if it reconnects prior to the session cache timeout expiring.)

---

**Agent database configuration properties**

Table 12: Agent database configuration property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitored</td>
<td>Starts the database monitoring agent automatically. This option is selected by default.</td>
</tr>
</tbody>
</table>

---

**Editing the database default configuration**

OpenEdge Management and OpenEdge Explorer provide a default database configuration. You can modify the existing configuration and then either save it as a replacement for the default configuration or create a new configuration.

**Note:** When you create a new configuration, the configuration automatically uses the pre-defined OpenEdge defaults.

**To edit a database default configuration:**

1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose default configuration you want to edit.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and Control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. Click the `configuration.<databasename>.defaultconfiguration`. The **Database Configuration** page appears, with the default configuration properties organized into the categories described in Viewing the various database configuration properties on page 31.
6. Click **Edit** and make the modifications.
7. Click **Save**.

**Creating a database configuration**

OpenEdge Management and OpenEdge Explorer provide a predefined default database configuration; however, you can create a new configuration by modifying the default settings.

**To create a database configuration:**

1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose default settings you want to modify to create a new configuration.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. Click **configuration.<database_name>.defaultconfiguration**. The **Database Configuration** page appears and displays the default configuration properties.
6. Click **Create**.
7. Type a new configuration group name in the field provided, and click **Save**. The **Database Configuration** page appears.
8. Click **Edit** to set the properties of the new database configuration. (Refer to for a description of each property.)
9. When you finish, click **Save**.

To make this new configuration the default, you must select it as the default on the **Database Configuration** page.

To assign the new configuration to a database, see **Assigning a different default database configuration** on page 43.

**Assigning a different default database configuration**

OpenEdge Management and OpenEdge Explorer provide a default database configuration. You can modify that default configuration, or you can create one or more additional configurations and assign one of them as the default.

**To assign a different default database configuration:**

1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database to which you want to assign a different default configuration.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. Click **Edit**.
6. From the **Default configuration** dropdown, select the default configuration you want to use.
7. Click **Save**.
Viewing the server group

The server group specifies the logical collection of server processes to start. You can apply properties to a database server group by either using the default configuration or creating and then specifying a configuration of your own choosing.

You can set or edit any of the database server group properties from OpenEdge Management or OpenEdge Explorer. The properties are organized into the General and Advanced categories.

To view the server group:

1. Click Resources from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose server group you want to view.
3. Click the Edit icon. The Database details page appears.
4. In the Command and control section of the page, click Configuration. The Database Configuration page appears.
5. Click servergroup.<databaseName>.defaultconfiguration.defaultservergroup. The Database Configuration page for the server group appears.

From the Database Configuration page, you can do the following:

- View a read-only display of the General and Advanced database server group properties as described in Database server group properties on page 44.
- Click Edit to modify the database server group properties. For details, see Editing a database server group on page 47.
- Click Create to create or modify a server group. For details, see Creating a database server group on page 47.
- Click Delete to delete a server group, provided it is not running. For details, see Deleting a database server group on page 52.

Database server group properties

The following sections describe the database server group properties.
# General database server group properties

## Table 13: General database server group properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Service name or port</strong></td>
<td>The service name or port number through which clients may connect to the database. The default value is 0.</td>
</tr>
<tr>
<td></td>
<td>Note: The service name or port number must represent a non-zero value in order to start a networked database broker. When you define multiple server groups for the same configuration, you must assign a unique service name or port number to each server group.</td>
</tr>
<tr>
<td><strong>Number of servers</strong></td>
<td>The maximum number of server processes that can be started to service clients. The default value is 0.</td>
</tr>
<tr>
<td><strong>Message buffer size</strong></td>
<td>The standard message buffer size in bytes. OpenEdge uses message buffers to move records (messages) between servers and remote clients. Records (plus 40-byte headers) larger than the message buffer size are fragmented into multiple messages. If your database records are large, increase this parameter to avoid record fragmentation. However, if the network works more efficiently with small messages, reduce the <strong>Message Buffer size</strong> value and fragment larger records. The default value is set dynamically based on your platform (UNIX or Windows) and the type of database (Personal, Workgroup, or Enterprise).</td>
</tr>
<tr>
<td><strong>Reporting interval</strong></td>
<td>The interval, in minutes, at which the server polls for license usage statistics that are reported in the usage report file. The default value is 1.</td>
</tr>
</tbody>
</table>
### TCP/IP Version

The TCP/IP version—IPv4 or IPv6—that the database broker uses to open the database port. The default value is IPv4.

### Client Type

One of the following client types:

- **4GL only** — The server group can service requests only from ABL clients. The settings are the same as those for the Both 4GL and SQL option.

- **SQL only** — The server group can service requests only from SQL clients. The settings are the same as those for the Both 4GL and SQL option, except that the Message buffer size and Reporting interval settings are unavailable.

- **Both 4GL and SQL** — The server group can service requests from both ABL and SQL clients.

**Note:** Remember that the client type choice you make here is honored only when you have installed the proper licenses for that client. For example, if you selected the 4GL only option during the OpenEdge installation, choosing the SQL only option or the Both 4GL and SQL option in the management console has no effect. However, if you chose the Both 4GL and SQL option during the OpenEdge installation, choosing SQL only in the management console is valid.

---

### Advanced database server group properties

**Table 14: Advanced database server group properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum clients per server</td>
<td>The maximum number of remote users per database server. The default value is 0.</td>
</tr>
<tr>
<td>Minimum clients per server</td>
<td>The minimum number of remote users on a server before the broker starts another server (up to the maximum number of servers). The default value is 1.</td>
</tr>
<tr>
<td>Maximum dynamic port</td>
<td>The port number that is the highest in a specified range of server port numbers accessible to a client. You specify the lowest port number with the Minimum dynamic port property. The default value is set dynamically based on your platform (UNIX or Windows) and the type of database (Personal, Workgroup, or Enterprise).</td>
</tr>
</tbody>
</table>
### Viewing the server group

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum dynamic port</td>
<td>The port number that is the lowest in a specified range of server port numbers accessible to a client. You specify the higher port number with the Maximum dynamic port property. Ports below 1025 are usually reserved for system services. The default value is set dynamically based on your platform (Unix or Windows) and the type of database (Personal, Workgroup, or Enterprise).</td>
</tr>
<tr>
<td>Host</td>
<td>The database's host machine.</td>
</tr>
</tbody>
</table>

#### Editing a database server group

For a database instance you added to OpenEdge Management or OpenEdge Explorer, you can edit any of the database server group properties.

To edit a database server group:

1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose server group you want to edit.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. Click the `servergroup.<databasename>.defaultconfiguration.defaultservergroup` you want to edit. The **Database Configuration** page for the server group appears.
6. Click **Edit** and make the modifications. (For details about the properties in the **General** and **Advanced** categories, refer to .)
7. Click **Save**.

#### Creating a database server group

You can create a new database server group from OpenEdge Management or OpenEdge Explorer.

To create a database server group:

1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database where you want to create a new server group.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. Click the `servergroup.<databasename>.defaultconfiguration.defaultservergroup` you want to edit. The **Database Configuration** page for the server group appears.
6. Click Create.

7. Type a new server group name in the field provided, and click Save. The Database Configuration page appears.

   You can click Edit to set the properties of the new database server group. (For details about the properties in each category, refer to .) When you finish, click Save.

### Starting or stopping the database

OpenEdge Management and OpenEdge Explorer support starting and stopping both local databases and databases running under a remote AdminServer on another machine. A local database resides on the same machine as OpenEdge Management or OpenEdge Explorer.

Before you can start or stop a database, you must first connect to the AdminServer that contains a definition of the database.

To start or stop a database, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters. The database agent works only if the database is running. Therefore, when you stop a database, the agent status automatically changes to Not Running. However, when you start a local database whose AgentMonitored property is disabled, the agent does not automatically start. You must specifically click Start Agent in the Database control page.

**Note:** You can run only one configuration for a database at a time. You can set the database configuration to start automatically when the AdminServer starts. For details, see Starting a database automatically on page 48.

### Starting a database automatically

You can set a database to start automatically when the AdminServer starts.

To start a database automatically:

1. Click Resources from the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database you want to start automatically.
3. Click the Edit icon. The Database details page appears.
4. In the Command and control section of the page, click Configuration. The Database Configuration page appears.
5. Click Edit.
6. Select the Autostart option by clicking in the check box.
7. Click Save.
Using the background writers and watchdog processes

On shared-memory systems, setting database background writer and watchdog processes can improve database performance. Background writers continuously perform overhead functions in the background.

Background writers

There are three types of background writers:

• **Asynchronous Page Writer (APW)** — The APW improves performance by performing overhead operations in the background. These operations provide available buffers, reduce the number of buffers that OpenEdge reads before writing to disk, and reduce the overhead associated with before-image checkpoints (the process of synchronizing the buffer pool of modified blocks to the database). You can start and stop an APW at any time without shutting down the database.

• **Before-image Writer (BIW)** — The BIW improves performance by continually writing before-image buffers to disk. These writes occur in the background. BIWs are optional; however, they are highly recommended for improving I/O performance. You can run only one BIW per database. You can start and stop a BIW at any time without shutting down the database.

• **After-image Writer (AIW)** — The AIW improves performance by continually writing after-image (AI) buffers to disk soon after OpenEdge fills the buffers. The after-imaging feature lets you recover a database that was damaged when a failure caused the loss of the database or primary recovery (BI) area. When you enable after-imaging, OpenEdge writes notes to the after-image files that contain a description of all changes to the database. You can run only one AIW process per database. You can start and stop an AIW at any time without shutting down the database.

The watchdog process

The watchdog process improves performance by cleaning up after improperly terminated processes. It releases locks, backs out any live transactions, and releases shared-memory locks. It also disconnects and cleans up the server’s remote clients.

Viewing database status

You can view the status of a database’s configurations, server groups, and auxiliary processes through OpenEdge Management or OpenEdge Explorer. You can also view the status of the database monitoring agent.

**Note:** Most of the broker instances provide a status view, enabling you to monitor the status of multiple brokers simultaneously. You can monitor instances located on the same or different servers.

To view the status of a database:
1. Click **Resources** from the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the database whose status you want to view.

3. Click the Edit icon. The **Database** details page appears.

4. In the **Command and control** section of the page, click **Control**. The **Database Control** page appears.

5. Review the following status information for the database:
   - **Database information** — The database name, the active port or ports, and the name of the configuration being used by the database.
   - **Database status** — The status for the database and monitoring agent.
   - **Database auxiliary status** — The status of the After image page writer, the Before image page writer, the Auxiliary page writers, and the Watchdog process for this server group.
   - **Properties** — The status of the Monitor and **Enabled** settings: enabled (checked) or disabled.

### Viewing database process details

You can access real-time details and statistics that provide you with snapshot information about a running database at the point you access this information from the **Control** page. You can review this information to help you assess the database's performance.

To access processing details:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the database whose status you want to view.

3. Click the Edit icon. The **Database** details page appears.

4. In the **Command and control** section of the page, click **Control**. The **Database Control** page appears.

5. Click a **PID** link. The following sections appear:
   - The **Process summary** section identifies the **Process name** and **Process start time**. **User id** and **Group id** values appear when UNIX-based data is shown. The **Parent pid** provides the identifier number associated with the process that spawned this current process.
   - The **Process statistics** section presents details about the database's real-time operational status. Values presented without parentheses identify that the processing time determined since the last scheduled polling interval, as noted, has occurred. Values presented within parentheses have been calculated based on information obtained since the start of the process.

### Process statistics data

The following table identifies and describes the fields of information presented in the **Process statistics** section.

**Table 15: Process statistics data**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resident size</td>
<td>The physical size of the process as defined by the host system.</td>
</tr>
<tr>
<td>Virtual size</td>
<td>The virtual size of the process as defined by the host system.</td>
</tr>
</tbody>
</table>
### The percentage of time spent using the CPU in either the user or kernel mode since the last scheduled poll.

**CPU**

The percentage of time spent using the CPU in either the user or kernel mode since the last scheduled poll divided by the number of CPU processors on the system. This value appears only when there is more than one CPU process on the system where the process is running.

**Weighted CPU**

The percentage of time spent using the CPU in either the user or kernel mode since the last scheduled poll divided by the number of CPU processors on the system. This value appears only when there is more than one CPU process on the system where the process is running.

**User time**

The amount of CPU time spent in the user mode since the last scheduled poll.

**Kernel time**

The amount of CPU time spent in the kernel mode since the last scheduled poll.

**Process time**

The sum of the values that appear in the User time and Kernel time fields.

---

## Deleting a database, database configuration, or database server group

You can delete a database, provided it is not running, and you can delete a database configuration or database server group, provided it is not currently in use by a database.

### Deleting a database

When you no longer need a particular database, you can delete the database and all associated configuration information from the management console. When you delete a database from the console, you are not deleting the actual, physical database files; you are only deleting the database definition from the `conmgr.properties` file. After you delete the database, it no longer appears in the console.

You cannot delete a database that is running. You must first stop the database and then delete it.

To delete a database from the management console:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose status you want to delete.
3. Click the Edit icon. The **Database** details page appears.
4. Click **Delete** and then click **OK** to confirm the deletion.

   The **OpenEdge Management Resources** page appears. A set of next-step options that are related to resources are available for your use.
Deleting a database configuration

You can delete a database configuration from the management console.

If a configuration is running, you cannot delete it. You must first stop the configuration, and then delete it.

To delete a database configuration from the management console:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database you want to delete.
3. Click the Edit icon. The Database details page appears.
4. In the Command and control section of the page, click Configuration. The Database Configuration page appears.
5. In the Configuration and Server Group Links section of the page, click the database configuration you want to delete. The configuration information appears.
6. Click Delete, and then click OK to confirm the deletion.

Deleting a database server group

You can delete a database server group from the management console.

If the configuration associated with the server group is in use, you cannot delete it. You must first stop the configuration, and then delete the server group.

To delete a database server group:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose server group you want to delete.
3. Click the Edit icon. The Database details page appears.
4. In the Command and control section of the page, click Configuration. The Database Configuration page appears.
5. In the Configuration and Server Group Links section of the page, click the server group you want to delete. The server group information appears.
6. Click Delete, and then click OK to confirm the deletion.

Configuring an OpenEdge Replication-enabled database

To configure your source database and target database for OpenEdge Replication, you must configure the replication property files for both the OpenEdge Replication source database and the OpenEdge Replication target database.
In the `OpenEdge-install-dir/properties` folder, OpenEdge Replication provides two sample property files: `source.repl.properties` and `target.repl.properties`. You can save these files and modify them, using either OpenEdge Management or OpenEdge Explorer, or a text editor, so that you can use them with your source and target OpenEdge Replication databases. Because `source.repl.properties` and `target.repl.properties` are samples, some of the properties and values are generic; it is up to you to customize the properties to reflect the settings you want to use with your own source and target databases.

When you use OpenEdge Management or OpenEdge Explorer to make configuration changes to the Replication server or Replication target properties, the changes are reflected in the applicable property file.

**Rules for setting properties for a Replication-enabled database**

From the management console, you can configure various properties for a database enabled for Replication. The options you have in the console for configuring these Replication-related properties depend upon the following rules:

- For a database **not enabled** for Replication:
  - If the Replication property file **does not** exist, no Replication links are available.
  - If the Replication property file **does** exist but is empty, a **Server (create new)** link and an **Agent (create new)** link appear.

  **Note:** You can configure transition only when one of these groups exists.

- If the Replication property file **does** exist and **is not** empty, then all groups defined in the property file appear.

- For a database **enabled** for Replication:
  - If the Replication property file **does not** exist, Replication-related links still appear. However, you can create only the group for which the database is enabled. In other words, you can create the Server group for source-enabled databases and the Agent group for target-enabled databases.
  - If the Replication property file **does** exist, only the groups that are applicable to the database enablement appear. For source enablement, **Server**, **ctrl-agents**, and **transition** appear. For target enablement, **Agent** and **transition** appear.
  - If the Transition group database-role is set to **reverse**, you either see the existing group for the reverse role or are allowed to create a new group for the reverse role.
  - If you specify **Alternate buffer pool** blocks for a Replication target database, the property setting is ignored.

**Setting Replication-related properties**

If you are familiar with OpenEdge Replication and its configuration, see the following topics for details about the properties you can configure through the management console:

- **Setting OpenEdge Replication Agent properties** on page 54
- **Setting OpenEdge Replication Control Agent properties** on page 56
- **Setting OpenEdge Replication Server properties** on page 55
Setting OpenEdge Replication Agent properties

You set OpenEdge Replication Agent properties to define the configuration for the local agent running for the target database.

Note: If you are not familiar with OpenEdge Replication and its configuration, see OpenEdge Replication: User Guide.

To set the OpenEdge Replication Agent properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose Replication Agent properties you want to set.
3. Click the Edit icon. The Database details page appears.
4. In the Command and control section of the page, click Configuration. The Database Configuration page appears.
5. In the OpenEdge Replication Group Links section, click Agent. The Database Replication Configuration page opens.
6. Set the following properties:

   • **Agent name** — Specifies the OpenEdge Replication agent name. By convention, the agent name should match the name specified in the source database properties file. ALL is not allowed as an agent name. Each agent must have a unique name.

   • **Database** — Specifies the source database name.

   • **Connection Timeout** — Specifies how many seconds the OpenEdge Replication agent will wait for connection from the OpenEdge Replication server before the Replication agent shuts itself down.

     The seconds value must be greater than or equal to 120, and less than or equal to 86,400.

     Using this property means you do not have to do a forced shutdown on your target database. If the OpenEdge Replication agent does not receive a connection attempt from the OpenEdge Replication server before the number of seconds specified have elapsed, the OpenEdge Replication agent will terminate and allow some limited system-level target database connections.

   • **Polling delay minimum** — Specifies the minimum value, in milliseconds, for a polling delay.

     By default, the polling delay starts at 5 milliseconds and automatically increases during periods of inactivity to a maximum of 500 milliseconds.

     The milliseconds value must be greater than or equal to one, and less than or equal to ten.

   • **Polling delay maximum** — Specifies the maximum value, in milliseconds, for a polling delay.

     By default, the polling delay starts at 5 milliseconds and automatically increases during periods of inactivity to a maximum of 500 milliseconds.

     The milliseconds value must be greater than or equal to 500 and less than or equal to 1000.

   • **Agent minimum port** — Specifies the minimum TCP port number. The agent selects a port in a range between the values specified by the minimum port and the maximum port.
The minimum port number value must be greater than 1024 and less than the value of the maximum port.

- **Agent maximum port** — Specifies the maximum TCP port number for the agent.
  
The maximum port number value must be greater than the minimum port + 1, and less than the maximum allowable port number on the system.
  
  For UNIX the maximum port number is 65534. For Windows the maximum port number is 32765.

- **Other arguments** — Any other arguments for which there is no defined property available.

7. Click **Save**.

## Setting OpenEdge Replication Server properties

You set the OpenEdge Replication Server properties to specify server and agent property values.

---

**Note:** If you are not familiar with OpenEdge Replication and its configuration, see *OpenEdge Replication: User Guide* before setting the properties.

---

The server properties are divided into the following categories:

- General
- Advanced

To set the OpenEdge Replication Server properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose Replication Server properties you want to set.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. In the **OpenEdge Replication Group Links** section, click **Server**. The **Database Replication Configuration** page opens.
6. Set the following **General** properties:

   - **Database** — Specifies the source database name.
   - **Defer agent startup** — Specifies for how long, in minutes, the server attempts to connect to an agent if the first connection attempt is unsuccessful. The minutes value must be greater than or equal to zero and less than or equal to 10080.
   - **Keep alive** — Specifies a timeout period for communications between a server and its agents. If a connection between the server and the agent is not verified before the timeout expires, failure recovery begins.
      
      By default, this property is enabled and has a value of 300 seconds. The minimum value is 60 seconds; there is no maximum value.
   - **Server transition timeout** — Specifies the number of seconds the target database waits before it performs auto-transition. This property is ignored when transition is set to manual.
      
      The value is incremented by the sum of the connect-timeout for all configured agents.
   - **Server transition** — Specifies how to transition the target database to a normal database.
If a synchronous agent or an asynchronous critical agent cannot reconnect with the source database within the time specified in the transition-timeout property, the following occurs:

- **auto** — The agent automatically transitions the target database.
- **manual** — The agent listens for the server until it reconnects with the source database or the DSRUTIL command is executed. The DSRUTIL db-name -C transition command transitions the target database.

7. Set the following **Advanced** properties:

- **Other arguments** — Any other arguments for which there is no defined property available.
- **Schema lock action** — Specifies the action an agent takes if an exclusive schema lock is not granted. Possible actions are:
  - **wait** — The agent waits until the exclusive schema lock is granted. The server blocks until the exclusive schema lock is granted.
  - **force** — The agent attempts to acquire the exclusive schema lock five times. If the fifth attempt fails, the agent disconnects all users from the target and makes another attempt. If the last attempt fails, the server and all agents terminate. When schema update activity completes, the server and target can be restarted.
- **Polling delay minimum** — Specifies the minimum value, in milliseconds, for a polling delay. By default, the polling delay starts at 5 milliseconds and automatically increases during periods of inactivity to a maximum of 500 milliseconds.
  The value must be greater than or equal to one and less than or equal to ten.
- **Polling delay maximum** — Specifies the maximum value, in milliseconds, for a polling delay. By default, the polling delay starts at 5 milliseconds and automatically increases during periods of inactivity to a maximum of 500 milliseconds.
  The value must be greater than 500 and less than 1000.
- **Server port minimum** — The minimum port available.
- **Server port maximum** — The maximum port available.
- **Agent shutdown action** — Specifies the action an agent takes during a shutdown when the Replication server ends. Possible actions are:
  - **recovery** — The agent remains active but in a standby state waiting for the replication server to reconnect.
  - **normal** — The agent terminates; the target database stays up.

8. Click **Save**.

**Setting OpenEdge Replication Control Agent properties**

You set the OpenEdge Replication control agent properties to define for the server which agents it will contact, where it will contact them, and how the agent should perform. Each server group supports up to two control agents.

**Note:** If you are not familiar with OpenEdge Replication and its configuration, see *OpenEdge Replication: User Guide* before setting the properties.
To set the OpenEdge Replication Control Agent properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the database whose Replication Control Agent properties you want to set.

3. Click the Edit icon. The **Database** details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.

5. In the **OpenEdge Replication Group Links** section, click **Control-Agent**. The **Database Replication Configuration** page opens.

6. Click **Edit**.

7. Set the properties as follows:
   - **Database** — Specifies the source database name.
   - **Host** — Specifies to the server which host the agent will start on.
   - **Port** — Specifies which port the server uses to connect to the target database. The port number specified must be the same port specified with the `-S` parameter when the target database broker was started.
   - **Connection Timeout** — Specifies how many seconds the OpenEdge Replication agent will wait for a connection from the OpenEdge Replication server before the Replication Agent shuts itself down. The seconds value must be greater than or equal to 120, and less than or equal to 86,400.

   Using this property means you do not have to do a forced shutdown on your target database. If the OpenEdge Replication agent does not receive a connection attempt from the OpenEdge Replication server before the number of seconds specified elapses, the OpenEdge Replication agent will terminate and allow some limited system-level target database connections.

   This property is also used by the server when reconnecting to the agent after communication has been lost.
   - **Maximum message** — The maximum number of bytes used for the TCP/IP communication messages.
   - **Replication method** — Asynchronous or synchronous.
   - **Critical agent** — Specifies whether the agent is critical. (Select the check box for a critical agent.)

     A critical agent is an asynchronous agent for the target database that can become the source database if the source database becomes unavailable.
   - **Other arguments** — Any other arguments for which there is no defined property available.

8. Click **Save**.

### Setting OpenEdge Replication Transition properties

You set the Transition properties to specify the transition process values, which are used when the target database becomes a normal database after a failure.

**Note:** If you are not familiar with OpenEdge Replication and its configuration, see *OpenEdge Replication: User Guide* before setting the properties.

The transition properties are divided into the following categories:

- **General**
To set the OpenEdge Replication Transition properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the database whose Replication Transition properties you want to set.
3. Click the Edit icon. The **Database** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Database Configuration** page appears.
5. In the **OpenEdge Replication Group Links** section, click **Transition**. The **Database Replication Configuration** page opens.
6. Set the following **General** properties:
   - **Database role**:
     - **reverse** — The role of the database is reversed: A source database becomes a target database, and a target database becomes a source database.
     - **normal** — The role of the database becomes that of a normal database; the database is no longer enabled for replication once the transition is performed. This is the default value.
   - **Responsibility**:
     - **primary** — This is the primary database.
     - **secondary** — This is the secondary database.
   - **Transition to agents** — Transitions to the first agent in the list when a failure occurs. If the first agent is not available, transition to the second agent in the list.
     A valid value for this property is any configured agent name; separate the names by a comma if you are listing more than one. For example, agent1,agent2.
7. Set the following **Start Up** properties:
   - **Restart after transition** — The database can be automatically restarted after transition is performed. When you select this option, the ***-startup-arguments** properties must be supplied, or the database startup will fail:
     - If the database role is normal, you must specify the **normal-startup-arguments**.
     - If the database role is reversed, you must specify the **source-startup-arguments** and the **target-startup-arguments**.
   - **Source startup arguments** — If the database is transitioned to a source database, these arguments are used when the database is started. The arguments are appended to the PROSERVE command (used to start the database).
     In most cases, the only argument specified here should be **-pf** followed by a parameter file name; for example, **-pf db-name.source.pf**.
     Because the database is a source, you must also specify the **-DBService replserv** argument, as an indication to the broker to start the Replication server.
• **Target startup arguments** — If the database is transitioned to a target database, these arguments are used when the database is started. The arguments will be appended to the PROSERVE command. In most cases, the only argument specified here should be `-pf` followed by a parameter file name; for example, `-pf db-name.target.pf`.

Because the database is a target, you must also specify the `-DBService replagent -S port-number or service-name` arguments, as an indication to the broker to start the Replication agent and to listen on the TCP/IP port specified with `-S port-number or service-name`.

• **Normal startup arguments** — If the database is transitioned to a normal database, these arguments are used when the database is started. The arguments will be appended to the PROSERVE command. In most cases, the only argument specified here should be `-pf` followed by a parameter file name; for example, `-pf db-name.normal.pf`.

8. Set the following **After image** properties:

   • **Begin after imaging after transition** — Select the option to automatically begin AI after a target-to-source database transition. After-imaging can be started for a database that has AI areas.

   • **Automatically add after imaging extents** — This property instructs the transition process to automatically add AI areas to the database if:
      
      • It is transitioned to a source database.
      
      • There are currently no AI areas for the database.

   If you do not select the option, AI areas are not added to the database. If you do select the option, AI areas are automatically added to the database by using the structure file specified in the `ai-structure-file` property.

   • **Structure file containing after image area definitions** — The name of the structure file that contains the list of AI areas to add.

9. Set the following **Back Up** properties:

   • **Backup method** — Specifies the backup method performed before AI is enabled. The following values are valid:
      
      • **mark** — Marks the database backed up by using the `rfutil db-name -C mark backedup` command. Marking the database as backed up does not allow future AI extents to be used when recovering from a disaster.
      
      • **full off-line** — Backs up the database offline by using the OpenEdge PROBKUP utility. The backup is performed in two steps. The first backup is a full backup, which is performed before AI is enabled for the database. The second backup is an incremental backup, which is performed after AI is enabled, and after the role of the database is changed.
      
      • **full on-line** — This backup is performed after the database is restarted, which occurs after the database has been transitioned.
      
      • **Full on-line and off-line backup arguments** — The arguments required for the full online and offline backups that are performed for the database. At a minimum, the target file or device must be specified in these arguments for both online and offline backups.

   To avoid overwriting a backup, do not use the same target file or device for both the backup and the incremental backup.

   Do not use backup validation parameters (such as `-vp` and `-vf`).

   The arguments should begin with `device-name`. 
• **Incremental backup arguments** — The arguments required for the offline incremental backup performed after AI is enabled and the database's role is reversed.

The arguments should begin with `device-name`. 
Configuring DataServers

You can configure DataServer property settings and add new DataServer instances in OpenEdge Management and OpenEdge Explorer. You can also start, stop, and delete DataServer instances, as well as view their status and log files.

For details, see the following topics:

• ODBC DataServer configuration and administration
• Oracle DataServer configuration and administration
• MS SQL Server DataServer configuration and administration

ODBC DataServer configuration and administration

Note: The following sections are applicable only for existing ODBC DataServer instances running on remote AdminServers prior to OpenEdge Release 11.7.

The OpenEdge DataServer for ODBC allows the OpenEdge Application Development Environment (ADE) and applications created with OpenEdge in Windows to access the ODBC-compliant databases DB2 and Sybase. You can use OpenEdge Management or OpenEdge Explorer to configure and administer the configuration of an OpenEdge DataServer for ODBC. The DataServer for ODBC installation provides one predefined DataServer broker (odbbroker1) and one predefined NameServer (NS1). You can use these predefined components as a starting point for creating and configuring additional DataServer brokers, and, if needed, NameServers. Each broker is referred to as an instance.
Working with the ODBC DataServer

To work with the ODBC DataServer, double-click the ODBC DataServer folder from the management console list frame. There is one pre-defined default ODBC DataServer broker (osbroker1).

Configuring an ODBC DataServer broker on page 62
Editing an ODBC DataServer broker configuration on page 69
Creating an ODBC DataServer broker on page 69
Starting or Stopping an Oracle DataServer broker on page 82
Starting an ODBC DataServer broker automatically on page 70
Viewing the status of an ODBC DataServer broker on page 70
Deleting an ODBC DataServer broker on page 69
Viewing the status of an ODBC DataServer broker on page 70

Configuring an ODBC DataServer broker

You can view or modify the following default configuration properties of an ODBC DataServer broker from the management console:

- Broker
- Server
- Environment variables

You can also create a new ODBC DataServer broker, view a broker's status, delete a broker you no longer need, or view the broker's log files.

Each of these tasks is described in the following sections.

Viewing or modifying ODBC DataServer broker properties

You can view or modify ODBC DataServer Broker properties in the following categories: general, owner information, controlling NameServer, Data Service, logging setting, and advanced features.

To view or modify broker properties for an ODBC DataServer:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker whose properties you want to view or modify.
3. Click the ODBC DataServer broker that you want to edit. The ODBC DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The ODBC DataServer Configuration page appears. From this page, you can:
   - View a read-only display of the ODBC broker properties as described in ODBC DataServer broker properties on page 63.
   - Click Edit to modify the broker properties. For details, see Editing an ODBC DataServer broker configuration on page 69.
**ODBC DataServer broker properties**

The following sections describe the ODBC DataServer broker properties.

**ODBC DataServer broker General properties**

Table 16: ODBC DataServer broker General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>Your general working directory. The default value is @WorkPath.</td>
</tr>
<tr>
<td>Operating mode</td>
<td>For DataServer brokers, the operating mode is state-aware. The default value is state-aware.</td>
</tr>
<tr>
<td>Port number</td>
<td>The type of the TCP/IP listening port that the DataServer broker uses. The default value is 4446.</td>
</tr>
<tr>
<td>Auto start</td>
<td>Starts the DataServer automatically when the AdminServer starts. The Auto start check box is cleared by default.</td>
</tr>
</tbody>
</table>

**ODBC DataServer broker Owner Information properties**

Table 17: ODBC DataServer broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group name</td>
<td>N/A</td>
</tr>
<tr>
<td>Username</td>
<td>Your user name. The default is blank.</td>
</tr>
<tr>
<td>Password</td>
<td>Your password. The default is blank.</td>
</tr>
</tbody>
</table>

**ODBC DataServer broker Controlling NameServer properties**

Table 18: ODBC DataServer broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>A check box that indicates whether to register the DataServer with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>Identifies the NameServer that the DataServer uses to find a broker that supports the correct application services. (If you did not select to register the DataServer with a NameServer, the field is dimmed and unavailable.)</td>
</tr>
</tbody>
</table>
How the broker specifies its hostname when registering with its controlling NameServer. This hostname information is passed onto a client application when it attempts to connect to a Data Service that the broker supports. The choices are:

- **Register IP** — (Default) This setting is the most efficient mechanism, and can be used in most cases. It registers with the IP address of the machine where the broker is located.

- **Register-LocalHost** — The broker registers with the hostname of the machine that it runs on. Use this setting when the broker runs on a machine with a single hostname and more than one IP address.

- **Register-HostName** — The broker registers with the values specified in the hostName property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.

**Registration host name**

Identifies the hostname value to be used if the **Registration mode** property is set to **Register-HostName**.

### ODBC DataServer broker Data Service properties

#### Table 19: ODBC DataServer broker Data Service properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data service name list</td>
<td>Data services are services that are provided by a selected broker's servers. The default service name is the broker name. The default value is blank.</td>
</tr>
<tr>
<td>Supports default service</td>
<td>A check box that indicates if the ODBC broker supports the default service. Select the check box to have the broker support the default service, or click to clear the check box and cancel support for the default service. The <strong>Supports default service</strong> check box is cleared by default.</td>
</tr>
</tbody>
</table>
## ODBC DataServer broker Logging Setting properties

Table 20: ODBC DataServer broker Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The general broker log filename. The default is <code>{WorkPath}\odbbroker1.broker.log</code>.</td>
</tr>
<tr>
<td>Broker logging level</td>
<td>The amount of information written to the broker log. The possible values are:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — No log file is written.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Only error-related information is written to the log file, which reduces the size of the log file.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Only some information is written to the log file.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — All broker activity is written to the log file. This might produce large files that use up disk space.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Information in addition to <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>Append to broker log file</td>
<td>Deselect to create a new broker log file each time a broker is started, even if the broker log file specified already exists. The <strong>Append to broker log file</strong> check box is selected by default.</td>
</tr>
<tr>
<td>Broker logging entry types</td>
<td>A single entry or comma delimited list of logging entry types.</td>
</tr>
<tr>
<td>Broker log file threshold size</td>
<td>Specifies a limit on how big the log file can get (in bytes). When the log is full, the broker creates a new log file with a sequence number.</td>
</tr>
<tr>
<td>Maximum number of broker log files</td>
<td>Limit on how many broker log files will be kept on the system when there is a threshold limit. It represents the number of log files including the current one and includes any rolled over log files that already exist when the process started.</td>
</tr>
</tbody>
</table>
## ODBC DataServer broker Advanced Features properties

Table 21: ODBC DataServer broker Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum client instances</td>
<td>The maximum number of client connections that the broker can support concurrently. The default value of 512 that is set for this parameter is high enough to ensure that the number of client connections is unlimited. Realistically, however, the system-level resources needed to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for &quot;out of space&quot; or &quot;OutOfMemory&quot; you might need to reduce the maximum number of client connections to a more reasonable value. Note, however, that lowering this value can cause some client requests to be rejected (&quot;Exceeded Max Clients&quot;). Configuring (and starting) multiple brokers to handle higher client loads alleviates the problem.</td>
</tr>
<tr>
<td>Priority weight (0-100)</td>
<td>An integer value between 0 and 100 that influences the share of the workload that the selected broker receives. The larger the value, the heavier the load that is distributed to the broker. Using the Priority weight value, the NameServer distributes client requests to the same DataServer service across all DataServer brokers. The default value is 0. You can update this property dynamically. Any changes will affect all current and new brokers.</td>
</tr>
<tr>
<td>Registration retry (in seconds)</td>
<td>After registering with its controlling NameServer, the DataServer broker periodically sends &quot;keep-alive&quot; messages to the NameServer to let the NameServer know that the broker is still active. The Registration retry value is the number of seconds that pass between &quot;keep-alive&quot; messages. The default value is 30.</td>
</tr>
</tbody>
</table>

### Viewing or modifying ODBC DataServer broker server properties

You can view or modify ODBC DataServer broker Server properties in the following categories: general and logging setting.

To view or modify server properties for an ODBC broker:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker whose server properties you want to view or modify.
3. Click the ODBC DataServer broker that you want to edit. The ODBC DataServer details page appears.

4. In the Command and control section of the page, click Configuration. The ODBC DataServer Configuration page appears. From this page, you can:
   - View a read-only display of the ODBC server properties as described in ODBC DataServer server properties on page 67.
   - Click Edit to modify the default server properties. For details, see Starting or Stopping an Oracle DataServer broker on page 82.

**ODBC DataServer server properties**

The following sections describe the ODBC DataServer server properties.

**ODBC DataServer server General properties**

Table 22: ODBC DataServer server General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Server executable file** (dynamic property) | The pathname of the DataServer executable that the broker runs. The default for the DataServer for ODBC in Windows is:  
`@{Startup\DLC}bin\_odbsrv.exe`  
You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated. |
| **Server startup parameters** (dynamic property) | The startup parameters for the server. You should not modify the default parameters, which are:  
`-svub -S -X -N TCP -U X -P x -hs 0 -s 40`  
You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated. |
| PROPATH                           | A list of directories OpenEdge searches to find procedures. |
**Minimum port number (dynamic property)**

The minimum port number that you want to make available for each DataServer broker. The default is blank. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Minimum port number</strong></td>
<td>The minimum port number that you want to make available for each DataServer broker. The default is blank. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>Maximum port number</strong></td>
<td>The maximum port number that you want to make available for each DataServer broker. The default is blank. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
</tbody>
</table>

### ODBC DataServer server Logging Setting property

#### Table 23: ODBC DataServer server Logging Setting property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server log filename</strong></td>
<td>The DataServer log filename. Enter a valid path name. Installed Default: @{WorkPath}\oddbroker1.server.log</td>
</tr>
</tbody>
</table>

### Setting ODBC DataServer broker environment variables

You can set environment variables for the DataServer broker for ODBC. You can set one or more environment variables that:

- Are accessible from your ABL application
- Affect the environment in which your application runs

For the ODBC DataServer, there are no required environment variables in Windows.

To set ODBC DataServer broker environment variables:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker whose environment variables you want to set.
3. Click the ODBC DataServer broker that you want to edit. The **ODBC DataServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **ODBC DataServer Configuration** page appears.
5. Click **Edit**.
6. Click **Environment Variables**.
7. Enter each variable name and then provide its value in the format \texttt{name=value}.
8. Click \texttt{Save} when you finish adding the variables and their values.

**Editing an ODBC DataServer broker configuration**

You can edit an ODBC DataServer broker configuration from the management console.

To edit an ODBC DataServer broker configuration:

1. Click \texttt{Resources} in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the ODBC DataServer broker whose configuration you want to edit.

3. Click the ODBC DataServer broker you want to edit. The \texttt{ODBC DataServer} details page appears.

4. In the \texttt{Command and control} section of the page, click \texttt{Configuration}. The \texttt{ODBC DataServer Configuration} page appears.

5. Click \texttt{Edit}. (For details about the broker and server properties, see \texttt{ODBC DataServer broker properties} on page 63 and \texttt{ODBC DataServer server properties} on page 67, respectively.)

6. Make the configuration changes, and then click \texttt{Save}.

**Creating an ODBC DataServer broker**

OpenEdge Management and OpenEdge Explorer provide a predefined default ODBC DataServer configuration; however, you can create a new configuration by modifying the default settings.

To create an ODBC DataServer broker:

1. From the drop-down for \texttt{Resources} on the management console menu, click \texttt{New OpenEdge Resource} > \texttt{ODBC DataServer}. The \texttt{ODBC DataServer Configuration} page appears.

2. Type the name of the new ODBC DataServer broker in the field provided.

3. Click \texttt{Save}. The \texttt{ODBC DataServer Configuration} page appears, allowing you to establish the broker's properties.

**Deleting an ODBC DataServer broker**

You can delete an inactive ODBC DataServer broker from the management console.

If a broker is running, you cannot delete it. (The \texttt{Delete} button is not available.) You must first stop the broker and then delete it.

To delete an ODBC DataServer broker:

1. Click \texttt{Resources} in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the ODBC DataServer broker.

3. From the list of ODBC DataServer brokers, click the ODBC DataServer that you want to delete. The \texttt{ODBC DataServer} details page appears.

4. Click \texttt{Delete}, and then click \texttt{OK} to confirm the deletion.
The OpenEdge Management Resources page appears. A set of next-step options that are related to resources are available for your use.

Starting or Stopping an ODBC DataServer broker

You can start or stop an ODBC DataServer broker from the management console, or you can choose to start the broker automatically when the AdminServer starts.

To start or stop an ODBC DataServer broker, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

Starting an ODBC DataServer broker automatically

You can set an ODBC DataServer broker to start automatically when the AdminServer starts.

To start an ODBC DataServer broker automatically:
1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker you want to start automatically.
3. From the list of ODBC DataServer brokers, click the ODBC DataServer that you want to edit. The ODBC DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The ODBC DataServer Configuration page appears.
5. Click Edit.
6. Select the Auto start option by clicking in the check box.
7. Click Save.

Viewing the status of an ODBC DataServer broker

You can view the status of any active DataServer broker, server groups, and auxiliary processes through the management console. Most of the broker instances provide a status view, enabling you to monitor the status of multiple brokers simultaneously.

To see the status of a broker, the broker must either be running or have been running.

To view the status of an ODBC DataServer broker:
1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker whose status you want to view.
3. Click the Edit icon. The ODBC DataServer details page appears.
4. In the Operations views section of the page, click Status.
   A Status summary and status details appear and provide the information described in ODBC DataServer broker status summary and details on page 71.
## ODBC DataServer broker status summary and details

### Table 24: ODBC DataServer broker status summary

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The broker's host machine.</td>
</tr>
<tr>
<td>Broker Name</td>
<td>The name of the broker whose status you are viewing.</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>For DataServer brokers, the operating mode is state-aware.</td>
</tr>
<tr>
<td>Broker Status</td>
<td>The current state of the broker.</td>
</tr>
<tr>
<td>Broker Port</td>
<td>The TCP/IP port number that the broker listens to.</td>
</tr>
<tr>
<td>Broker PID</td>
<td>The process ID of the broker.</td>
</tr>
<tr>
<td>Active Servers</td>
<td>The number of active servers.</td>
</tr>
<tr>
<td>Busy Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Locked Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Available Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Active Clients (now, peak)</td>
<td>N/A</td>
</tr>
<tr>
<td>Client Queue Depth (cur, max)</td>
<td>The number of clients waiting for agents to become available to service their request. The current value (cur) represents the number of waiting clients at the moment the status is displayed, and the maximum value (max) represents the largest number of clients waiting concurrently since the broker started.</td>
</tr>
<tr>
<td>Total Requests</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Wait (max, avg)</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Duration (max, avg)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Table 25: ODBC DataServer broker status details

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>The process id of the server process</td>
</tr>
<tr>
<td>State</td>
<td>The current state of the server process</td>
</tr>
<tr>
<td>Port</td>
<td>The TCP/IP port number used by the server</td>
</tr>
<tr>
<td>nRq</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Viewing the ODBC DataServer log files

You can view the log files for an ODBC DataServer by using the log file viewer. The log file viewer allows you to examine the ODBC DataServer log files through an HTML interface.

To access the log file viewer:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the ODBC DataServer broker whose log file viewer you want to access.
3. From the ODBC DataServer brokers list, click the ODBC DataServer that you want to view. The ODBC DataServer details page appears.
4. In the **Command and control** section of the page, Click **Log File Viewer of Broker** or **Log File Viewer of Servers**. You can use the Log File Viewer in the following ways:
   - Use the **Show** field to control how many database log file entries display at one time. The number entered into the **Show** field cannot be less than 10.
   - Use the **Overlap** field to control how many entries are repeated from screen to screen.

   **Note:** The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

   - Click **Reload** after changing the values in either the **Show** field or the **Overlap** field. If you do not reload, the viewer continues to display the previous values.
   - Click **Go To** to control which numbered entry in the log file the viewer display begins. For example, a value of 10 entered into the **Go To** field will begin the display from the tenth log file entry.

   **Note:** You must click **Go To** after entering a value in the **Go To** field, or the viewer does not update its display.

The default display of entries is in ascending order; choose **Descending** to change the display. Note that the **Show** field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

   - Click **First** to display the first *x* entries, where *x* is the value in the **Show** field.
   - Click **Previous** to display the previous *x* entries, where *x* is the value in the **Show** field.
   - Click **Next** to display the next *x* entries, where *x* is the value in the **Show** field.
   - Click **Last** to display the last *x* entries, where *x* is the value in the **Show** field.
• To view additional log file entries without changing your current starting log file entry, leave the **Go To** field blank, change the value in the **Show** field, and click **Reload**.

• If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the **Log file status** field.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer’s inactivity, the reloaded log file view might not match the previous log file view of that resource.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.

---

**Oracle DataServer configuration and administration**

The OpenEdge® DataServer for Oracle allows the OpenEdge Application Development Environment (ADE) and applications created with OpenEdge in Windows to access certain Oracle databases.

The DataServer for Oracle is also supported on several UNIX and LINUX platforms. For details, see Progress Communities.

You can use OpenEdge Management or OpenEdge Explorer to establish and administer the configuration of a Progress OpenEdge DataServer for Oracle. The DataServer for Oracle installation provides one predefined DataServer broker (orabroker1). You can use this predefined component as a starting point for creating and configuring additional DataServer brokers. Each broker is referred to as an instance.

---

**Working with the Oracle DataServer broker**

To work with the Oracle DataServer, double-click the **Oracle DataServer** folder from the management console list frame. There is one pre-defined default Oracle DataServer (orabroker1).

You can perform the following Oracle DataServer actions:

• **Configuring an Oracle DataServer broker** on page 74

• **Editing an Oracle DataServer broker configuration** on page 81

• **Creating an Oracle DataServer broker** on page 81

• **Starting or Stopping an Oracle DataServer broker** on page 82

• **Starting an Oracle DataServer broker automatically** on page 82

• **Viewing the status of an Oracle DataServer broker** on page 82

• **Deleting an Oracle DataServer broker** on page 81

• **Viewing the ODBC DataServer log files** on page 72
Configuring an Oracle DataServer broker

You can view or modify the following default configuration properties of an Oracle DataServer broker from OpenEdge Management or OpenEdge Explorer:

- Broker
- Server
- Environment variables

You can also create a new Oracle DataServer broker, view a broker's status, delete a broker you no longer need, or view the broker's log files.

Each of these tasks is described in the following sections.

Viewing or modifying Oracle DataServer broker properties

You can view or modify Oracle DataServer broker properties in the following categories: general properties, owner information, controlling NameServer information, Data Service, setting logging properties, and advanced features.

To view or modify broker properties for an Oracle broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker whose properties you want to view or modify.
3. Click the Oracle DataServer broker that you want to view. The Oracle DataServer details page appears.
4. In the Command and control section of the page, Click Configuration. The Oracle DataServer Configuration page appears. From this page, you can:
   - View a read-only display of the Oracle broker properties as described in Oracle DataServer broker properties on page 74.
   - Click Edit to modify the broker properties. For details, see Editing an Oracle DataServer broker configuration on page 81.

Oracle DataServer broker properties

The following sections describe the Oracle DataServer broker properties.

Oracle DataServer broker General properties

Table 26: Oracle DataServer broker General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>Your general working directory. The default value is @WorkPath.</td>
</tr>
<tr>
<td>Operating mode</td>
<td>For DataServer brokers, the operating mode is state-aware. The default value is state-aware.</td>
</tr>
</tbody>
</table>
The type of the TCP/IP listening port that the DataServer broker uses. The default value is 4446.

Starts the DataServer automatically when the AdminServer starts. The **Auto start** check box is cleared by default.

### Oracle DataServer broker Owner Information properties

Table 27: Oracle DataServer broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group name</td>
<td>(UNIX only) The name of the group.</td>
</tr>
<tr>
<td>Username</td>
<td>Your user name. The default is blank.</td>
</tr>
<tr>
<td>Password</td>
<td>Your password. The default is blank.</td>
</tr>
</tbody>
</table>

### Oracle DataServer broker Controlling NameServer properties

Table 28: Oracle DataServer broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>A check box that indicates whether to register the DataServer with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>Identifies the NameServer that the DataServer uses to find a broker that supports the correct application services. (If you did not select to register the DataServer with a NameServer, the field is dimmed and unavailable.)</td>
</tr>
</tbody>
</table>
### Registration mode

How the broker specifies its hostname when registering with its controlling NameServer. This hostname information is passed onto a client application when it attempts to connect to a Data Service that the broker supports. The choices are:

- **Register IP** — (Default) This setting is the most efficient mechanism, and can be used in most cases. It registers with the IP address of the machine where the broker is located.

- **Register-LocalHost** — The broker registers with the hostname of the machine that it runs on. Use this setting when the broker runs on a machine with a single hostname and more than one IP address.

- **Register-HostName** — The broker registers with the values specified in the hostName property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.

### Registration host name

Identifies the hostname value to be used if the Registration mode property is set to Register-HostName.

---

### Oracle DataServer broker Data Service properties

Table 29: Oracle DataServer broker Data Service properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data service name list</td>
<td>Data services are services that are provided by a selected broker’s servers. The default service name is the broker name. The default value is blank.</td>
</tr>
<tr>
<td>Supports default service</td>
<td>A check box that indicates whether the ODBC broker supports the default service. Select the check box to have the broker support the default service, or click to clear the check box and cancel support for the default service. The Supports default service check box is cleared by default.</td>
</tr>
</tbody>
</table>
## Oracle DataServer broker Logging Setting properties

Table 30: Oracle DataServer broker Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The general broker log filename. The default is @{WorkPath}\orabroker1.broker.log.</td>
</tr>
<tr>
<td>Broker logging level</td>
<td>The amount of information written to the broker log. The possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• None — No log file is written.</td>
</tr>
<tr>
<td></td>
<td>• Error Only — Only error-related information is written to the log file, which reduces the size of the log file.</td>
</tr>
<tr>
<td></td>
<td>• Basic — Only some information is written to the log file.</td>
</tr>
<tr>
<td></td>
<td>• Verbose — All broker activity is written to the log file. This might produce large files that use up disk space.</td>
</tr>
<tr>
<td></td>
<td>• Extended — Information in addition to Verbose.</td>
</tr>
<tr>
<td>Append to broker log file</td>
<td>Create a new broker log file each time a broker is started, even if the broker log file specified already exists. The Append to broker log file check box is selected by default.</td>
</tr>
<tr>
<td>Broker logging entry types</td>
<td>A single entry or comma delimited list of logging entry types.</td>
</tr>
<tr>
<td>Broker log file threshold size</td>
<td>Specifies a limit on how big the log file can get (in bytes). When the log is full, the broker creates a new log file with a sequence number.</td>
</tr>
<tr>
<td>Maximum number of broker log files</td>
<td>Limit on how many broker log files will be kept on the system when there is a threshold limit. It represents the number of log files including the current on and includes any rolled over log files that already exist when the process started.</td>
</tr>
</tbody>
</table>
Oracle DataServer broker Advanced Features properties

Table 31: Oracle Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum client instances</td>
<td>The maximum number of client connections that the broker can support concurrently. The default value of 512 that is set for this parameter is high enough to ensure that the number of client connections is unlimited. Realistically, however, the system-level resources needed to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for &quot;out of space&quot; or &quot;OutOfMemory&quot; you might need to reduce the maximum number of client connections to a more reasonable value. Note, however, that lowering this value can cause some client requests to be rejected (&quot;Exceeded Max Clients&quot;). Configuring and starting multiple brokers to handle higher client loads alleviates the problem.</td>
</tr>
<tr>
<td>Priority weight (0-100) (dynamic property)</td>
<td>An integer value between 0 and 100 that influences the share of the workload that the selected broker receives. The larger the value, the heavier the load that is distributed to the broker. Using the Priority weight value, the NameServer distributes client requests to the same DataServer service across all DataServer brokers. The default value is 0. You can update this property dynamically. Any changes will affect all current and new brokers.</td>
</tr>
<tr>
<td>Registration retry (in seconds)</td>
<td>After registering with its controlling NameServer, the DataServer broker periodically sends &quot;keep-alive&quot; messages to the NameServer to let the NameServer know that the broker is still active. The Registration retry value is the number of seconds that pass between &quot;keep-alive&quot; messages. The default value is 30.</td>
</tr>
</tbody>
</table>

Viewing or modifying Oracle DataServer server properties

You can view or modify Oracle DataServer server properties in the following categories: General and Logging Setting.

To view or modify server properties for an Oracle broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker whose server properties you want to view or modify.
3. Click the Oracle DataServer broker that you want to view. The Oracle DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The Oracle DataServer Configuration page appears. From this page, you can do the following:
- View a read-only display of the Oracle server properties as described in Oracle DataServer server properties on page 79.
- Click **Edit** to modify the server properties. For details, see Editing an Oracle DataServer broker configuration on page 81.

### Oracle DataServer server properties

The following sections describe the Oracle DataServer server properties.

#### Oracle DataServer server General properties

**Table 32: Oracle DataServer server General properties**

<table>
<thead>
<tr>
<th>Server properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server executable file</strong> (dynamic property)</td>
<td>The pathname of the DataServer executable that the broker runs. The defaults for the DataServer for Oracle on the Windows and UNIX platforms are as follows:</td>
</tr>
<tr>
<td></td>
<td>• Windows: @{Startup\DLC\bin_orasrv.exe</td>
</tr>
<tr>
<td></td>
<td>• UNIX: OpenEdge-install-dir/_orasrv</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>Server startup parameters</strong> (dynamic property)</td>
<td>The startup parameters for the server. You should not modify the default parameters, which are:</td>
</tr>
<tr>
<td></td>
<td>-svub -S -X -N TCP -U X -P x -hs 0 -s 40</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>PROPATH</strong></td>
<td>A list of directories OpenEdge searches to find procedures.</td>
</tr>
</tbody>
</table>
### Oracle DataServer server Logging Setting property

Table 33: Oracle DataServer server Logging Setting properties

<table>
<thead>
<tr>
<th>Server property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The DataServer log filename. Enter a valid path name. Installed Default:</td>
</tr>
<tr>
<td></td>
<td>@{WorkPath}\orabroker1.server.log</td>
</tr>
</tbody>
</table>

### Setting Oracle DataServer broker environment variables

You can set environment variables for the DataServer broker for Oracle. You can set one or more environment variables that:

- Are accessible from your ABL application
- Affect the environment in which your application runs

For the Oracle DataServer, there are no required environment variables in Windows.

To set Oracle DataServer broker environment variables:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker whose environment variables you want to set.
3. Click the Oracle DataServer broker that you want to view. The **Oracle DataServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **Oracle DataServer Configuration** page appears.
5. Click **Edit**.
6. Click **Environment Variables**.
7. Enter each variable name and then provide its value in the format name=value.
8. Click Save when you finish adding the variables and their values.

**Editing an Oracle DataServer broker configuration**

You can edit an Oracle DataServer configuration from the management console.

To edit an Oracle DataServer broker configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker whose configuration you want to edit.
3. Click the Oracle DataServer broker. The Oracle DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The Oracle DataServer Configuration page appears.
5. Click Edit. (For details about the broker or server properties, see Oracle DataServer broker properties on page 74 and Oracle DataServer server properties on page 79.)
6. Make the configuration changes, and then click Save.

**Creating an Oracle DataServer broker**

OpenEdge Management and OpenEdge Explorer provide a predefined default Oracle DataServer configuration; however, you can create a new configuration by modifying the default settings.

To create an Oracle DataServer broker:

1. From the drop-down for Resources on the management console menu, click New OpenEdge Resource > Oracle DataServer. The Oracle DataServer Configuration page appears.
2. Type the name of the new Oracle DataServer in the field provided.
3. Click Save. The Oracle DataServer Configuration page appears, allowing you to configure the broker's properties.

**Deleting an Oracle DataServer broker**

You can delete an inactive Oracle DataServer broker from the management console.

If a broker is running, you cannot delete it. (The Delete button is not available.) You must first stop the broker and then delete it.

To delete an Oracle DataServer broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker you want to delete.
3. From the Oracle DataServer brokers list, click the broker that you want to delete. The Oracle DataServer details page appears.
4. Click Delete, and then click OK to confirm the deletion.
Starting or Stopping an Oracle DataServer broker

You can start or stop an Oracle DataServer broker from the management console.

To start or stop an Oracle DataServer broker, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

Starting an Oracle DataServer broker automatically

You can set a Oracle DataServer broker to start automatically when the AdminServer starts.

To start an Oracle DataServer broker automatically:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer brokers.
3. From the search list, click the Oracle DataServer broker that you want to start automatically. The Oracle DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The Oracle DataServer Configuration page appears.
5. Click Edit.
6. Select the Auto start check box.
7. Click Save.

Viewing the status of an Oracle DataServer broker

You can view the status of any active Oracle DataServer broker, server groups, and auxiliary processes through the management console. Most of the broker instances provide a status view, enabling you to monitor the status of multiple brokers simultaneously.

To see the status of a broker, the broker must be running or have been running.

To view the status of an Oracle DataServer broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker you want to start automatically.
3. Click the Oracle DataServer broker. The Oracle DataServer details page appears.
4. In the Operations views section of the page, click Status.
   A Status summary and summary details appear and provide the information described in Oracle DataServer broker property Status summary and details on page 83.
### Oracle DataServer broker property Status summary and details

**Table 34: Oracle DataServer broker property Status summary**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The broker's host machine.</td>
</tr>
<tr>
<td>Broker Name</td>
<td>The name of the broker whose status you are viewing.</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>For DataServer brokers, the operating mode is state-aware.</td>
</tr>
<tr>
<td>Broker Status</td>
<td>The current state of the broker.</td>
</tr>
<tr>
<td>Broker Port</td>
<td>The TCP/IP port number that the broker listens to.</td>
</tr>
<tr>
<td>Broker PID</td>
<td>The process ID of the broker.</td>
</tr>
<tr>
<td>Active Servers</td>
<td>The number of active servers.</td>
</tr>
<tr>
<td>Busy Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Locked Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Available Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Active Clients (now, peak)</td>
<td>The number of clients waiting for agents to become available to service their request. The current value (cur) represents the number of waiting clients at the moment the status is displayed, and the maximum value (max) represents the largest number of clients waiting concurrently since the broker was started.</td>
</tr>
<tr>
<td>Client Queue Depth (cur, max)</td>
<td>N/A</td>
</tr>
<tr>
<td>Total Requests</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Wait (max, avg)</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Duration (max, avg)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Table 35: Oracle DataServer broker Status details**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>The process id of the server process</td>
</tr>
<tr>
<td>State</td>
<td>The current state of the server process</td>
</tr>
<tr>
<td>Port</td>
<td>The TCP/IP port number used by the server</td>
</tr>
<tr>
<td>nRq</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Viewing the Oracle DataServer broker log files

You can view the log files for an Oracle DataServer broker by using the log file viewer. The log file viewer allows you to examine the Oracle DataServer log files through an HTML interface.

To access the log file viewer:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the Oracle DataServer broker whose log file viewer you want to access.
3. From the list, click the Oracle DataServer broker. The **Oracle DataServer** details page appears.
4. In the **Command and control** section of the page, Click **Log File Viewer of Broker** or **Log File Viewer of Servers**.

You can use the Log File Viewer in the following ways:

- Use the **Show** field to control how many database log file entries display at one time. The number entered into the **Show** field cannot be less than 10.
- Use the **Overlap** field to control how many entries are repeated from screen to screen.

**Note:** The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

- Click **Reload** after changing the values in either the **Show** field or the **Overlap** field. If you do not reload, the viewer continues to display the previous values.
- Click **Go To** to control which numbered entry in the log file the viewer display begins. For example, a value of 10 entered into the **Go To** field will begin the display from the tenth log file entry.

**Note:** You must click **Go To** after entering a value in the **Go To** field or the viewer will not update its display.

The default display of entries is in ascending order; choose **Descending** to change the display. Note that the **Show** field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

- Click **First** to display the first x entries, where x is the value in the **Show** field.
- Click **Previous** to display the previous x entries, where x is the value in the **Show** field.
- Click **Next** to display the next x entries, where x is the value in the **Show** field.
- Click **Last** to display the last x entries, where x is the value in the **Show** field.
• To view additional log file entries without changing your current starting log file entry, leave the Go To field blank, change the value in the Show field, and click Reload.
• If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the Log file status field.
• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer’s inactivity, the reloaded log file view might not match the previous log file view of that resource.
• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.

MS SQL Server DataServer configuration and administration

The OpenEdge DataServer for MS SQL Server allows the OpenEdge Application Development Environment (ADE) and applications created with OpenEdge in Windows to access Microsoft SQL Server.

You can use OpenEdge Explorer to configure and administer the configuration of an OpenEdge DataServer for SQL Server. The DataServer for SQL Server installation provides one predefined DataServer broker (mssbroker1). You can use this predefined component as a starting point for creating and configuring additional DataServer brokers. Each broker is referred to as an instance.

Working with the MS SQL Server DataServer broker

To work with the MS SQL Server DataServer, double-click the MSS DataServer folder from the OpenEdge Explorer list frame. There is one pre-defined default MS SQL Server DataServer (mssbroker1).

You can perform the following MS SQL Server actions:
• Configuring an MS SQL Server DataServer broker on page 86
• Editing an MS SQL Server DataServer broker configuration on page 92
• Creating an MS SQL Server DataServer broker on page 92
• Starting or Stopping an MS SQL Server DataServer broker on page 93
• Starting an MS SQL Server DataServer broker automatically on page 93
• Viewing the MS SQL Server DataServer log files on page 95
• Deleting an MS SQL Server DataServer broker on page 93
• Viewing the MS SQL Server DataServer log files on page 95
Configuring an MS SQL Server DataServer broker

You can view or modify the following configuration properties of an MS SQL Server DataServer from the management console:

- Broker
- Server
- Environment variables

You can also create a new MS SQL Server DataServer broker, view a broker’s status, or delete a broker you no longer need.

Each of these tasks is described in the following sections.

Viewing or modifying MS SQL Server DataServer broker properties

You can view or modify MS SQL Server DataServer Broker properties in the following categories: general properties, owner information, controlling NameServer information, Data Service, setting logging properties, and advanced features.

To view or modify broker properties for an MS SQL Server broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer broker whose properties you want to view or modify.
3. Click the Edit icon. The MSS DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The MSS DataServer Configuration page appears. From this page, you can:
   - View a read-only display of the MS SQL Server broker properties as described in MS SQL Server DataServer broker properties on page 86.
   - Click Edit to modify the broker properties. For details, see Editing an MS SQL Server DataServer broker configuration on page 92.

MS SQL Server DataServer broker properties

The following sections describe the MS SQL Server DataServer broker properties.

MS SQL Server DataServer broker General properties

Table 36: MS SQL Server DataServer broker General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>Your general working directory. The default value is @{WorkPath}.</td>
</tr>
<tr>
<td>Operating mode</td>
<td>For DataServer brokers, the operating mode is state-aware. The default value is state-aware.</td>
</tr>
</tbody>
</table>
The type of the TCP/IP listening port that the DataServer broker uses. The default value is 4446.

Starts the DataServer automatically when the AdminServer starts. The Auto start check box is cleared by default.

### MS SQL Server DataServer broker Owner Information properties

#### Table 37: MS SQL Server DataServer broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group name</td>
<td>(UNIX only) The name of the group.</td>
</tr>
<tr>
<td>Username</td>
<td>Your user name. The default is blank.</td>
</tr>
<tr>
<td>Password</td>
<td>Your password. The default is blank.</td>
</tr>
</tbody>
</table>

### MS SQL Server DataServer broker Controlling NameServer properties

#### Table 38: MS SQL Server DataServer broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>A check box that indicates whether to register the DataServer with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>Identifies the NameServer that the DataServer uses to find a broker that supports the correct application services. (If you did not select to register the DataServer with a NameServer, the field is dimmed and unavailable.)</td>
</tr>
</tbody>
</table>
How the broker specifies its hostname when registering with its controlling NameServer. This hostname information is passed onto a client application when it attempts to connect to a Data Service that the broker supports. The choices are:

- **Register IP** — (Default) This setting is the most efficient mechanism, and can be used in most cases. It registers with the IP address of the machine where the broker is located.

- **Register-LocalHost** — The broker registers with the hostname of the machine that it runs on. Use this setting when the broker runs on a machine with a single hostname and more than one IP address.

- **Register-HostName** — The broker registers with the values specified in the hostName property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.

### MS SQL Server DataServer broker Data Service properties

**Table 39: MS SQL Server DataServer broker Data Service properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data service name list</td>
<td>Data services are services that are provided by a selected broker's servers.</td>
</tr>
<tr>
<td></td>
<td>The default service name is the broker name. The default value is blank.</td>
</tr>
</tbody>
</table>

**Supports default service**

A check box that indicates whether the MS SQL Server broker supports the default service. Select the check box to have the broker support the default service, or click to clear the check box and cancel support for the default service. The **Supports default service** check box is cleared by default.

### MS SQL Server DataServer broker Logging Setting properties

**Table 40: MS SQL Server DataServer broker Logging Setting properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The general broker log filename. The default is @{WorkPath}\mssbroker1.broker.log.</td>
</tr>
</tbody>
</table>
### Broker logging level

The amount of information written to the broker log. The possible values are as follows:

- **None** — No log file is written.
- **Error Only** — Only error-related information is written to the log file, which reduces the size of the log file.
- **Basic** — Only some information is written to the log file.
- **Verbose** — All broker activity is written to the log file. This might produce large files that use up disk space.
- **Extended** — Information in addition to Verbose.

### Append to broker log file

Create a new broker log file each time a broker is started, even if the broker log file specified already exists. The **Append to broker log file** check box is selected by default.

### Broker logging entry types

A single entry or comma delimited list of logging entry types.

### Broker log file threshold size

Specifies a limit on how big the log file can get (in bytes). When the log is full, the broker creates a new log file with a sequence number.

### Maximum number of broker log files

Limit on how many broker log files will be kept on the system when there is a threshold limit. It represents the number of log files including the current on and includes any rolled over log files that already exist when the process started.

## MS SQL Server DataServer broker Advanced Features properties

### Table 41: MS SQL Server DataServer broker Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum client instances</td>
<td>The maximum number of client connections that the broker can support concurrently. The default value of 512 that is set for this parameter is high enough to ensure that the number of client connections is unlimited. Realistically, however, the system-level resources needed to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for &quot;out of space&quot; or &quot;OutOfMemory&quot; you may need to reduce the maximum number of client connections to a more reasonable value. Note, however, that lowering this value can cause some client requests to be rejected (&quot;Exceeded Max Clients&quot;). Configuring and starting multiple brokers to handle higher client loads alleviates the problem.</td>
</tr>
</tbody>
</table>
An integer value between 0 and 100 that influences the share of the workload that the selected broker receives. The larger the value, the heavier the load that is distributed to the broker. Using the **Priority weight** value, the NameServer distributes client requests to the same DataServer service across all DataServer brokers. The default value is 0. You can update this property dynamically. Any changes will affect all current and new brokers.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority weight (0-100)</strong></td>
<td>(dynamic property)</td>
</tr>
<tr>
<td><strong>Registration retry (in seconds)</strong></td>
<td>After registering with its controlling NameServer, the DataServer broker periodically sends &quot;keep-alive&quot; messages to the NameServer to let the NameServer know that the broker is still active. The <strong>Registration retry</strong> value is the number of seconds that pass between &quot;keep-alive&quot; messages. The default value is 30.</td>
</tr>
</tbody>
</table>

**Viewing or modifying MS SQL Server DataServer server properties**

You can view or modify MS SQL Server DataServer properties in the following categories: **General** and **Logging Setting**.

To view or modify server properties for an MS SQL DataServer broker:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer broker whose server properties you want to view or modify.
3. From the list, click the MS SQL DataServer broker. The **MSS DataServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **MSS DataServer Configuration** page appears. From this page, you can:
   - View a read-only display of the MS SQL Server server properties as described in **MS SQL Server DataServer server properties** on page 90.
   - Click **Edit** to modify the server properties. For details, see **Editing an MS SQL Server DataServer broker configuration** on page 92.
   - View the MS SQL Server DataServer log file. For details, see **Viewing the MS SQL Server DataServer log files** on page 95.

**MS SQL Server DataServer server properties**

The following sections describe the MS SQL Server DataServer server properties.
### MS SQL Server DataServer server General properties

#### Table 42: MS SQL Server DataServer server properties

<table>
<thead>
<tr>
<th>Property category or property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server executable file (dynamic property)</strong></td>
<td>The pathname of the DataServer executable that the broker runs. The default for the DataServer for MS SQL Server in Windows is:</td>
</tr>
<tr>
<td></td>
<td><code>{Startup\DLC}bin\msssrv.exe</code></td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>Server startup parameters (dynamic property)</strong></td>
<td>The startup parameters for the server. You should not modify the default parameters, which are:</td>
</tr>
<tr>
<td></td>
<td><code>-svub -S -X -N TCP -U X -P x -hs 0 -s 40</code></td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>PROPATH</strong></td>
<td>A list of directories OpenEdge searches to find procedures.</td>
</tr>
<tr>
<td><strong>Minimum port number (dynamic property)</strong></td>
<td>The minimum port number that you want to make available for each DataServer broker. The default is blank.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td><strong>Maximum port number (dynamic property)</strong></td>
<td>The maximum port number that you want to make available for each DataServer broker. The default is blank.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
</tbody>
</table>

### MS SQL Server DataServer server Logging Setting properties

#### Table 43: MS SQL Server DataServer server Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server log filename</strong></td>
<td>The DataServer log filename. Enter a valid path name.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: <code>{WorkPath}\mssbroker1.server.log</code></td>
</tr>
</tbody>
</table>
Setting MS SQL Server DataServer broker environment variables

You can set environment variables for the DataServer broker for MS SQL Server. You can set one or more environment variables that:

- Are accessible from your ABL application
- Affect the environment in which your application runs

For the MS SQL Server DataServer, there are no required environment variables in Windows.

To set MS SQL Server DataServer environment variables:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer whose environment variables you want to set.
3. From the list, click the MS SQL DataServer broker. The MSS DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The MSS DataServer Configuration page appears.
5. Click Edit.
6. Click Environment Variables.
7. Enter each variable name and then provide its value in the format name=value.
8. Click Save when you finish adding the variables and their values.

Editing an MS SQL Server DataServer broker configuration

You can edit an MS SQL Server DataServer broker configuration from the management console.

To edit an MS SQL Server DataServer configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer whose configuration you want to edit.
3. From the list, click the MS SQL DataServer broker. The MSS DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The MSS DataServer Configuration page appears.
5. Click Edit. (For details about the broker and server properties, see MS SQL Server DataServer broker properties on page 86 and MS SQL Server DataServer server properties on page 90.)
6. Make the configuration changes, and then click Save.

Creating an MS SQL Server DataServer broker

OpenEdge Management and OpenEdge Explorer provide a predefined default MS SQL Server DataServer configuration; however, you can create a new configuration by modifying the default settings.

To create an MS SQL Server DataServer broker:
1. From the drop-down for Resources on the management console menu, click New OpenEdge Resource > MSS DataServer. The MSS DataServer Configuration page appears.

2. Type the name of the new DataServer in the field provided.

3. Click Save. The MSS DataServer Configuration page appears, allowing you to establish the broker's properties.

**Deleting an MS SQL Server DataServer broker**

You can delete an inactive DataServer broker for MS SQL Server from the management console. If a broker is running, you cannot delete it. You must first stop the broker and then delete it.

To delete an MS SQL Server DataServer broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer you want to delete.
3. From the list, click the MS SQL DataServer broker. The MSS DataServer details page appears.
4. Click Delete, and then click OK to confirm the deletion.

**Starting or Stopping an MS SQL Server DataServer broker**

You can start or stop an MS SQL Server DataServer broker from the management console.

To start or stop an MS SQL DataServer broker, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters

**Starting an MS SQL Server DataServer broker automatically**

You can set a broker for MS SQL Server to start automatically when the AdminServer starts.

To start an MS SQL DataServer broker automatically:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer you want to start automatically.
3. MS SQL DataServer broker. The MSS DataServer details page appears.
4. In the Command and control section of the page, click Configuration. The MSS DataServer Configuration page appears.
5. Click Edit.
6. Select the Auto start option by clicking in the check box.
7. Click Save.
Viewing the status of an MS SQL Server DataServer broker

You can view the status of any active MS SQL Server DataServer broker, server groups, and auxiliary processes through the management console. Most of the broker instances provide a status view, enabling you to monitor the status of multiple brokers simultaneously.

To see the status of a broker, the broker must be running or have been running.

To view the status of an MS SQL Server DataServer broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer broker whose status you want to view.
3. From the list, click MS SQL DataServer broker. The MSS DataServer details page appears.
4. In the Operations views section of the page, click Status.

A Status summary and summary details appear and provide the information described in MS SQL Server DataServer broker property status summary and details on page 94.

MS SQL Server DataServer broker property status summary and details

Table 44: MS SQL Server DataServer broker property status summary

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The broker’s host machine.</td>
</tr>
<tr>
<td>Broker Name</td>
<td>The name of the broker whose status you are viewing.</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>For DataServer brokers, the operating mode is state-aware.</td>
</tr>
<tr>
<td>Broker Status</td>
<td>The current state of the broker.</td>
</tr>
<tr>
<td>Broker Port</td>
<td>The TCP/IP port number that the broker listens to.</td>
</tr>
<tr>
<td>Broker PID</td>
<td>The process ID of the broker.</td>
</tr>
<tr>
<td>Active Servers</td>
<td>The number of active servers.</td>
</tr>
<tr>
<td>Busy Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Locked Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Available Servers</td>
<td>N/A</td>
</tr>
<tr>
<td>Active Clients (now, peak)</td>
<td>N/A</td>
</tr>
</tbody>
</table>
### Client Queue Depth (cur, max)

The number of clients waiting for agents to become available to service their request. The current value (cur) represents the number of waiting clients at the moment the status is displayed, and the maximum value (max) represents the largest number of clients waiting concurrently since the broker was started.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client Queue Depth</td>
<td>The number of clients waiting for agents to become available to service their request. The current value (cur) represents the number of waiting clients at the moment the status is displayed, and the maximum value (max) represents the largest number of clients waiting concurrently since the broker was started.</td>
</tr>
<tr>
<td>Total Requests</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Wait (max, avg)</td>
<td>N/A</td>
</tr>
<tr>
<td>Rq Duration (max, avg)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

#### Table 45: MS SQL Server DataServer broker status details

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>The process id of the server process</td>
</tr>
<tr>
<td>State</td>
<td>The current state of the server process</td>
</tr>
<tr>
<td>Port</td>
<td>The TCP/IP port number used by the server</td>
</tr>
<tr>
<td>nRq</td>
<td>N/A</td>
</tr>
<tr>
<td>nRcvd</td>
<td>N/A</td>
</tr>
<tr>
<td>nSent</td>
<td>N/A</td>
</tr>
<tr>
<td>Started</td>
<td>The time stamp indicating when the server process is started</td>
</tr>
<tr>
<td>Last Change</td>
<td>The time stamp indicating when the server process last changed state</td>
</tr>
</tbody>
</table>

### Viewing the MS SQL Server DataServer log files

You can view the log files for an MS SQL Server DataServer by using the log file viewer. The log file viewer allows you to examine the MS SQL Server DataServer log files through an HTML interface.

To access the log file viewer:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the MS SQL DataServer broker whose log file viewer you want to access.
3. From the list, click the MS SQL DataServer broker. The **MSS DataServer** details page appears.
4. In the **Command and control** section of the page, Click **Log File Viewer of Broker** or **Log File Viewer of Servers**. You can use the Log File Viewer in the following ways:
   - Use the **Show** field to control how many database log file entries display at one time. The number entered into the **Show** field cannot be less than 10.
• Use the **Overlap** field to control how many entries are repeated from screen to screen.

**Note:** The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

• Click **Reload** after changing the values in either the **Show** field or the **Overlap** field. If you do not reload, the viewer continues to display the previous values.

• Click **Go To** to control which numbered entry in the log file the viewer begins its display with. For example, a value of 10 entered into the **Go To** field will begin the display from the tenth log file entry.

**Note:** You must click **Go To** after entering a value in the **Go To** field or the viewer will not update its display.

The default display of entries is in ascending order; choose **Descending** to change the display. Note that the **Show** field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

• Click **First** to display the first x entries, where x is the value in the **Show** field.

• Click **Previous** to display the previous x entries, where x is the value in the **Show** field.

• Click **Next** to display the next x entries, where x is the value in the **Show** field.

• Click **Last** to display the last x entries, where x is the value in the **Show** field.

• To view additional log file entries without changing your current starting log file entry, leave the **Go To** field blank, change the value in the **Show** field, and click **Reload**.

• If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the **Log file status** field.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer’s inactivity, the reloaded log file view might not match the previous log file view of that resource.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.
Configuring NameServers

You can configure NameServer property settings and add new NameServer instances in OpenEdge Management and OpenEdge Explorer. You can also start, stop, and delete NameServer instances, as well as view their status and log files.

For details, see the following topics:

- NameServer configuration and administration
- Configuring a NameServer
- Editing a NameServer configuration
- Creating a NameServer
- Deleting a NameServer
- Configuring fault tolerance and load balancing
- Starting or Stopping a NameServer
- Viewing the status of a NameServer
- Handling UDP and firewall issues
- Viewing the NameServer log file
NameServer configuration and administration

You can optionally use the NameServer to mediate client connections for instances of OpenEdge Unified Broker products that you configure with OpenEdge Management or OpenEdge Explorer. The Unified Broker products include all OpenEdge resources that are managed through the AdminServer. For more information on the Unified Broker products and the AdminServer, see OpenEdge Getting Started: Installation and Configuration.

When you choose to use the NameServer, each component instance registers with only one NameServer, which is known as the instance's controlling NameServer. Each time an instance starts up, it registers with its controlling NameServer by sending its location and other configuration information. The NameServer uses this information to help resolve client connection requests. A particular NameServer can accept registrations from any number of instances, including instances of more than one type of component.

Depending on the way you configure and use the OpenEdge product, you can decide to specify a controlling NameServer to handle client connection requests. For example, your configuration might be simple and not require the location transparency the NameServer provides, or it might be more complex and, therefore, dependent on this and other NameServer features.

The OpenEdge Unified Broker products that you can optionally choose to register with the NameServer include:

- AppServers
- AppServer Internet Adapters
- DataServers for Oracle and MS SQL Server configured to use the NameServer
- SonicMQ Adapters
- WebSpeed Transaction Servers

NameServer features

You can use OpenEdge Management or OpenEdge Explorer to configure both local NameServer instances and remote NameServer instances. The NameServer installation provides one sample NameServer (NS1), and you can use the sample as a template for creating and configuring additional NameServers.

In its role as a connection arbitrator, the NameServer provides location transparency, connection-level fault tolerance, server-level fault tolerance, and optional load balancing. For details about these features, see Configuring fault tolerance and load balancing on page 108.

Working with the NameServer

To work with a NameServer instance, select the NameServer folder from the management console’s list pane. There is one pre-defined default NameServer (NS1).

You can perform the following NameServer actions:

- Configuring a NameServer on page 99
- Editing a NameServer configuration on page 106
- Creating a NameServer on page 107
- Starting or Stopping a NameServer on page 109
- Deleting a NameServer on page 107
NameServer as an optional client connection request handler

Configuring a NameServer to work with the AppServer, a DataServer, a SonicMQ Adapter, or a WebSpeed Transaction Server is optional. Depending on the way you configure and use the OpenEdge product, you can decide if you want to specify a controlling NameServer. For example, your configuration might be simple and not require location transparency or load balancing, or it might be more complex and, therefore, dependent on those features.

Local and remote NameServers

Before configuring a NameServer, you must know the network location where it is to run. For any network host, you can configure two types of NameServer instances:

- **Local** — An instance that runs locally on the host where it is defined
- **Remote** — An instance that references a NameServer that is defined and runs on a machine that is remote from the host where the reference is defined

The purpose of defining a remote NameServer instance is to provide an easy way for having multiple Unified Broker products refer to a controlling NameServer that runs on another machine. You cannot start, stop, obtain status, or modify any properties of a remote NameServer instance other than the host name and port number that it references.

Configuring a NameServer

The OpenEdge® NameServer controls a pool of brokers that register (with the NameServer) the application services they provide. The NameServer can then direct client connection requests to a broker that supports a requested application service. The NameServer also provides location transparency, and, when the NameServer Enterprise Edition is installed, the ability to perform load balancing. Load balancing allows you to balance client workload among multiple brokers that service the same application (that is, the same set of procedures and resources).

For more information, see Configuring local instances on page 99 and Configuring remote instances on page 105.

Configuring local instances

You configure a local NameServer to run on the selected host where it is defined. Unified Broker products running on the same host as a local NameServer instance can use it as their controlling NameServer.

You can view or edit the configuration properties of a NameServer from the management console. You can also create a new NameServer, view a NameServer's status, delete a NameServer you no longer need, or view the NameServer's log file.
To configure a local NameServer instance:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the local NameServer broker instance you want to configure.

3. Click the NameServer broker instance. The **NameServer** details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The **NameServer Configuration** page appears. From this page, you can:

   - View a read-only display of the NameServer properties as described in **NameServer properties** on page 100.
   - Click **Edit** to modify the NameServer properties. For details, see **Editing a NameServer configuration** on page 106.

### NameServer properties

The following sections describe the NameServer properties.

#### NameServer Location properties

**Table 46: NameServer Location properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>A read-only property that indicates whether the NameServer is local or remote:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Local</strong> — The NameServer runs locally on the selected host. You can configure all the properties of a local NameServer.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Remote</strong> — The NameServer runs remotely on a network machine that is separate from the selected host.</td>
</tr>
<tr>
<td></td>
<td>You can set this property only when you first create the NameServer instance. To change the location of a NameServer, you must first delete and then recreate the NameServer with the new location setting.</td>
</tr>
<tr>
<td>Host name</td>
<td>The name of the host. If the NameServer is local, this is set to <strong>localhost</strong>. Otherwise, type the name of the remote host where the NameServer is to run.</td>
</tr>
<tr>
<td>Port number</td>
<td>The number of the UDP port that the NameServer uses to listen for client connection requests and registration messages from AppServers, DataServers, and WebSpeed Transaction Servers.</td>
</tr>
</tbody>
</table>
### NameServer General properties

Table 47: NameServer General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>The NameServer working directory. Enter the pathname.</td>
</tr>
<tr>
<td>Broker keep alive timeout</td>
<td>A value, in seconds, that indicates how often the NameServer checks for Unified Broker instances that have timed out. When an OpenEdge Unified Broker instance registers with a NameServer, the instance indicates how often it sends &quot;keep-alive&quot; messages by setting a registration retry property value (a property setting in Advanced Features for the Unified Broker). Once a NameServer determines that it did not receive a &quot;keep-alive&quot; message from a Unified Broker instance within the broker's registration retry time, the NameServer automatically unregisters the instance. Note that you should use a <strong>Broker keep alive timeout</strong> value that is somewhat larger than the Unified Broker instance's registration retry value. The NameServer adjusts the specified value to allow for normal networking delays that can occur within your computer network. To keep the NameServer from using up computer resources unnecessarily, set the <strong>Broker keep alive timeout</strong> to a value that is at least 30% larger than the typical Unified Broker registration retry value. You can update this property dynamically. Any changes will affect all current and new NameServers.</td>
</tr>
<tr>
<td>Auto start</td>
<td>A check box that you can select if you want the NameServer to start automatically when the controlling AdminServer starts.</td>
</tr>
<tr>
<td>jvmargs</td>
<td>Any jvm arguments you set.</td>
</tr>
</tbody>
</table>

### NameServer Logging Setting properties

Table 48: NameServer Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The NameServer log filename. Enter a valid path name. The installed default is @WorkPath\NS1.NS.log.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NameServer logging level (dynamic property)</td>
<td>A value that specifies the amount of information that is written to the NameServer log. Select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>The installed default is <strong>Basic</strong>. You can update this property dynamically. Any changes will affect all current and new NameServers.</td>
</tr>
<tr>
<td>Append to NameServer log file</td>
<td>A check box that indicates whether a new server log file should be created when the NameServer is started.</td>
</tr>
<tr>
<td></td>
<td>To create a new NameServer log file each time the NameServer is started, even if the NameServer log file already exists, clear the check box.</td>
</tr>
<tr>
<td></td>
<td>To append log entries to the existing NameServer log file, select the check box.</td>
</tr>
<tr>
<td></td>
<td><strong>Append to NameServer log file</strong> is selected by default.</td>
</tr>
</tbody>
</table>
### Logging entry types (dynamic property)

The default and only supported entry is NSPlumbing. Specifying the NSPlumbing log entry type turns on logging for different NameServer actions based on logging level:

- **Basic** — Logs messages when a NameServer starts and stops, for load balancing status, and when registering and unregistering brokers. This is the default logging level.

- **Verbose** — Same as **Basic**, plus logs requests from clients and status information on the communication between client and NameServer and between NameServer and brokers.

- **Extended** — Same as **Verbose**, plus logs debugging messages on broker unregistering.

Log entries generated for this type contain the tag NSPlumbing. You can update this property dynamically. Any changes will affect all current and new NameServers.

### Log file threshold size

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

### Maximum number of log files

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of NameServer log files to keep. The specified number represents the maximum total number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the Log file threshold size, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:filename.#####.extension.
NameServer Advanced Features properties

### Table 49: NameServer Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Neighboring NameServers</strong></td>
<td>A list of selected NameServers to which this NameServer can forward connection requests for application services that are not registered with it (that is, the application service name is unknown). The list contains all NameServer instances defined in the management console, initially with none selected. To make a NameServer a neighboring NameServer, select the unselected NameServer to highlight it. You can select as many NameServer instances as you want to be neighboring NameServers. Each neighboring NameServer thus appears highlighted in the list. To remove a NameServer from the list of neighboring NameServers, select the already-selected NameServer to remove the highlight. The setting of this property is optional. When a NameServer receives a request for an application service name that is unknown and there are no neighboring NameServers specified, the NameServer sends a message to the requesting client indicating that the application service is unknown. When such a NameServer has neighboring NameServers specified, it forwards the request to each of the neighboring NameServers. The first neighboring NameServer that has the application service name, and responds to the connection request, provides the specified connection to the client.</td>
</tr>
<tr>
<td><strong>TCP/IP Version</strong></td>
<td>Internet Protocol for network communication:</td>
</tr>
<tr>
<td></td>
<td>• <strong>IPv4</strong> — Accepts only IPv4 connections</td>
</tr>
<tr>
<td></td>
<td>• <strong>IPv6</strong> — Allows IPv4 and IPv6 connections using mapped address (where supported)</td>
</tr>
<tr>
<td></td>
<td>The default is IPv4.</td>
</tr>
<tr>
<td></td>
<td>You must also set the <code>jvmArgs</code> property in the <code>ubroker.properties</code> file to override the default behavior of the JVM for IPv6 connections.</td>
</tr>
<tr>
<td></td>
<td><code>jvmArgs=-Djava.net.preferIPv4Stack=false</code>&lt;br&gt;<code>-Djava.net.preferIPv6Addresses=true</code></td>
</tr>
<tr>
<td><strong>Enable dynamic property updates</strong></td>
<td>Allows dynamic changes to occur to several of the instance’s properties, without requiring you to shut down the instance. This property is disabled by default.</td>
</tr>
</tbody>
</table>
Any property that is dynamic is labeled as such in this document.

### NameServer Environment Variables properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment</td>
<td>Allows you to add and delete environment variables.</td>
</tr>
<tr>
<td></td>
<td>Environment variables that you add to the list are scoped to the process in which the NameServer executes.</td>
</tr>
</tbody>
</table>

### Configuring remote instances

A remote NameServer instance is a reference to a NameServer that runs on some other host. By defining a remote instance, you allow a Unified Broker running on the local host to use the NameServer running on the remote host as its controlling NameServer. Thus, a remote NameServer configuration requires that you define:

- One local NameServer instance on the host where the NameServer runs
- One remote instance on each host where a Unified Broker uses the remote NameServer as its controlling NameServer

For example, consider that you are creating a NameServer instance on one system, which is named Sys1. The NameServer instance is actually physically running under an AdminServer on another system, which is named Sys2. The instance is remote for Sys1 and local for Sys2.

To create a remote instance, you need the host name and port number of Sys2 and its NameServer. You then use that information to create a remote NameServer on Sys1.

To configure a remote NameServer instance:

1. Configure at least one local NameServer instance on the remote host.
2. Using the management console, connect to the AdminServer on a host where you want to define the remote instance, and filter or search for the NameServer broker instances for that host.
3. Click the Edit icon for an existing remote NameServer instance. The NameServer details page appears.
   You can also create a new NameServer, specifying its name and indicating that its location is remote. For details, see Creating a NameServer on page 107.
4. In the Command and control section of the details page, click Configuration. The NameServer Configuration page appears.
5. Click Edit. The Location default is local. If you are editing an existing instance or creating a new instance, change the location to remote.
6. Set or change the host name to a machine where you have a NameServer defined as a local instance. Set the port number to the port on which the local instance you want to reference is listening.
7. Click Save.
Editing a NameServer configuration

You can edit a NameServer configuration from the management console.

To edit a NameServer configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the NameServer broker instance whose configuration you want to edit.
3. Click the NameServer broker instance. The NameServer details page appears.
4. In the Command and control section of the page, click Configuration. The NameServer Configuration page appears.
5. Click Edit. (For details about the NameServer properties, see NameServer properties on page 100.)
6. Make the configuration changes, and then click Save. Any changes you make are saved in the ubroker.properties file.

Setting NameServer environment variables

You can set or delete environment variables for a NameServer.

To set NameServer environment variables:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the NameServer broker instance whose environment variables you want to set.
3. Click the NameServer broker instance. The NameServer details page appears.
4. In the Command and control section of the page, click Configuration. The NameServer Configuration page appears.
5. Click Edit.
6. Click Environment Variables.
7. Enter each variable name and then provide its value in the format name=value.
8. Click Save when you finish adding the variables and their values.

Deleting NameServer environment variables

To delete NameServer environment variables:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the NameServer broker instance whose environment variables you want to delete.
3. Click the NameServer broker instance. The NameServer details page appears.
4. In the Command and control section of the page, click Configuration. The NameServer Configuration page appears.
5. Click Edit.
Creating a NameServer

OpenEdge Management and OpenEdge Explorer provide a sample configuration NameServer instance (NS1) to use as a default.

You can create two types of NameServer instance:

- **Local** — Where the instance definition resides on, and the NameServer itself executes on, the selected host.
- **Remote** — Where the instance definition is a reference to a NameServer that resides and executes on a machine that is remote from the selected host.

To create a NameServer:

1. From the drop-down for Resources on the management console menu, click **New OpenEdge Resource > NameServer**. The **NameServer Configuration** page appears.
2. Type the name of the new NameServer in the field provided.

   **Note:** The NameServer name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured NameServer names.

3. Click **Save**. The **NameServer Configuration** page appears, allowing you to configure the NameServer's properties. (For details about the properties, see **NameServer properties** on page 100.)

   Once you create a NameServer instance, you cannot change the location setting unless you delete and recreate the instance.

   **Note:** If you are defining a remote NameServer instance, set its host name and port number properties to reference a NameServer that is defined as a local NameServer on some other machine in your network. For details, see **Configuring remote instances** on page 105.

Deleting a NameServer

You can delete any NameServer instance from the management console as long as neither of the following conditions is true:

- The NameServer instance is running.
- One or more Unified Broker instances reference this NameServer instance as their controlling NameServer.

To delete the NameServer, you must stop the NameServer instance and delete all Unified Broker instances that reference it.

To delete a NameServer instance:
1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the NameServer broker instance you want to delete.

3. Click the NameServer broker instance. The **NameServer** details page appears.

4. Click **Delete**, and then click **OK** to confirm the deletion.

   The **OpenEdge Management Resources** page appears. A set of next-step options that are related to resources are available for your use.

---

### Configuring fault tolerance and load balancing

With the NameServer, you have access to connection-level fault tolerance, server-level fault tolerance, and load balancing.

#### Connection-level fault tolerance

Multiple NameServers can cooperate to resolve a client connection. If one NameServer becomes unavailable, another can take its place, thus providing connection-level fault tolerance. When you configure and use NameServers with connection-level fault tolerance, each requesting client uses the Unified Broker connection provided by the first responding NameServer and ignores the rest.

You can configure two types of connection-level fault tolerance:

- NameServer replication
- NameServer neighbors

#### NameServer replication

To use NameServer replication, you first configure multiple NameServers on one subnet to listen on the same UDP port. You then have each Unified Broker instance register with its controlling NameServer and have each client contact its connecting NameServer using the subnet broadcast address and the same UDP port number. Thus, Unified Broker registrations are automatically replicated in each NameServer, and client connection requests are handled by each NameServer. The first replicated NameServer to respond to a client request provides the Unified Broker connection.

#### NameServer neighbors

To use NameServer neighbors, you configure multiple NameServers on one or more subnets. For each such NameServer, you specify the instance name of each of the other NameServers that you want to act as a NameServer neighbor. When a client makes a connection request to a specific NameServer, if this NameServer determines that it cannot satisfy the request (usually because it does not recognize the specified application service), it passes the request to all of its NameServer neighbors. The first NameServer neighbor to respond to a client request provides the Unified Broker connection.

You can configure and use NameServer replication and NameServer neighbors together to provide the maximum in connection-level fault tolerance.
Server-level fault tolerance

If you have the load-balancing option installed, you can configure multiple Unified Broker instances of a given type to support the same application service and share the same controlling NameServer. In this configuration, the controlling NameServer distributes client connection requests randomly across Unified Broker instances. If one Unified Broker instance becomes unavailable to handle client requests, another is available to take its place, thus providing server-level fault tolerance.

Load balancing

If you specify a priority weight factor for each Unified Broker instance, the NameServer distributes the connection requests proportionately, thus balancing the load. For example, suppose that an Inventory service is supported by WebSpeed brokers WS1, WS2, and WS3, which have weight factors of 20, 20, and 60, respectively. The selection algorithm used by the NameServer guarantees that WS1 and WS2 are selected 20% of the time and WS3 is selected 60% of the time.

Note: You do not assign application services or weight factors to NameServers, but rather to Unified Broker instances. Each Unified Broker instance passes its weight factor and list of application service names to its controlling NameServer when it registers with the NameServer.

Location transparency

When a Unified Broker instance is started, it registers with its controlling NameServer the application services that it supports. (An application service is an arbitrary designation for the business function that a Unified Broker instance provides.) The NameServer maintains a separate list of application services for each Unified Broker type. When a client of a certain type requests a connection to an application service, the NameServer redirects the connection request to a Unified Broker instance of the corresponding type that supports the requested application service. Because the client always connects indirectly to a Unified Broker through the NameServer, the client does not need to know the location of the Unified Broker instance. It needs to know only the location of the NameServer and the name of a valid application service that it wants.

Starting or Stopping a NameServer

From the management console, you can directly start or stop a local NameServer on the machine where it runs. You can also set a NameServer to start automatically when the AdminServer starts. For details, see Starting a NameServer automatically on page 109.

Note: To start or stop a remote NameServer, access the corresponding local NameServer instance selected in the AdminServer of the host on which it runs.

To start or stop a NameServer, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

Starting a NameServer automatically

You can set a NameServer to start automatically when the AdminServer starts.
To start a NameServer automatically:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the NameServer broker instance you want to start automatically.

3. Click the NameServer broker instance. The NameServer details page appears.

4. In the Command and control section of the page, click Configuration. The NameServer Configuration page appears.

5. Click Edit.

6. Click the General tab.

7. Select the Auto start option by clicking in the check box.

8. Click Save.

**Viewing the status of a NameServer**

You can view the status of any running NameServer instance through the management console.

Each NameServer instance provides a separate status view. You can monitor the status of only those NameServer instances running locally on a selected host.

To view the status of a NameServer instance:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the NameServer broker instance whose status you want to view.

3. Click the NameServer broker instance. The NameServer details page appears.

4. In the Operations views section of the page, click Status.

A status summary and summary details appear.

The summary, as described in NameServer status summary properties on page 110, provides collective property and statistics information for the NameServer and the OpenEdge Unified Broker instances that it coordinates.

The details, as described in NameServer status detail properties on page 111, provide a table of status information for all Unified Broker instances that are registered with the NameServer. Each row in the table contains information for a single Unified Broker instance.

**NameServer status summary properties**

Table 51: NameServer status summary properties

<table>
<thead>
<tr>
<th>Property or statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The name of the host machine</td>
</tr>
<tr>
<td>Port</td>
<td>The UDP port number that the NameServer listens to</td>
</tr>
<tr>
<td>Timeout</td>
<td>The Unified Broker keep alive timeout value for the NameServer</td>
</tr>
</tbody>
</table>
### NameServer status detail properties

Table 52: NameServer status detail properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppService Name</td>
<td>The name of an application service registered with the NameServer.</td>
</tr>
<tr>
<td>UUID</td>
<td>The unique number for the Unified Broker instance.</td>
</tr>
<tr>
<td>Name</td>
<td>The Unified Broker instance name.</td>
</tr>
<tr>
<td>Host</td>
<td>The name of the host where the Unified Broker instance resides.</td>
</tr>
<tr>
<td>Port</td>
<td>The TCP/IP port number used by the Unified Broker instance to handle client connection requests.</td>
</tr>
<tr>
<td>Weight</td>
<td>The priority weight assigned to the Unified Broker instance for load balancing.</td>
</tr>
<tr>
<td>Timeout</td>
<td>The number of seconds that must pass between keep-alive messages sent by the Unified Broker instance before the NameServer assumes that the instance is no longer available. Once the timeout expires, the NameServer automatically unregisters the Unified Broker instance.</td>
</tr>
</tbody>
</table>

### Handling UDP and firewall issues

Whenever an AppServer Internet Adapter (AIA) or WebSpeed Messenger instance communicates with a NameServer, the instance is assigned a UDP port. The NameServer uses this port to send a response message to the instance. By default this port is dynamically assigned by the operating system within the UDP port range of 1024 to 65535.
If there is a firewall between the instance and the NameServer, the default setting requires that all UDP ports in that range must be open from the network where the NameServer exists to the network where the AIA or Messenger instance exists. Otherwise, communication between the instance and the NameServer cannot occur.

However, the only port that must be open in the firewall is the controlling NameServer port for UDP traffic going from the network containing the AIA or Messenger instance to the network containing the NameServer. To increase security, you can limit the range of ports from which an AIA or Messenger instance can be assigned an UDP port. The NameServer Client Port Minimum (minNSClientPort) and the NameServer Client Port Maximum (maxNSClientPort) properties can be used to reduce the number of UDP ports that you must open in the firewall. When you set these properties, OpenEdge chooses the first port in that range that is not currently used by some other application for UDP broadcast communications. If no port in this range is available, the default behavior is to retry the ports for a period of time.

**Viewing the NameServer log file**

The NameServer log file viewer allows you to examine the contents of log files through an HTML interface. For details, see the section about using the NameServer log file viewer in *OpenEdge Management: Servers, DataServers, Messengers, and Adapters*. 
Configuring AppServers and AppServer Internet Adapters

You can use OpenEdge Management or OpenEdge Explorer to configure the properties of OpenEdge resources that you are monitoring, such as AppServers or AppServer Internet Adapters. Any configuration changes that you make to either of these resources in OpenEdge Management or OpenEdge Explorer are automatically reflected in the configuration property file `ubroker.properties`.

This chapter provides information about working with AppServer and AppServer Internet Adapter configuration properties.

For details, see the following topics:

- AppServer configuration and administration
- For additional AppServer information
- AppServer Internet Adapter configuration and administration

AppServer configuration and administration

With the AppServer, you can make ABL applications available to multiple machines over a network.

You can use OpenEdge Management or OpenEdge Explorer to configure new AppServer instances and administer the configurations of existing instances. The AppServer installation provides one sample AppServer (asbroker1). You can use this sample component as a template for creating and configuring additional AppServers. Each AppServer is called an instance.
Understanding AppServer configuration

Each AppServer consists of an AppServer broker and one or more AppServer agents. An AppServer agent is a single ABL Virtual Machine (AVM) instance running on an AppServer. Each AppServer agent executes ABL procedures on behalf of AppServer clients, including ABL and Open Clients (Java, .NET, and Web services clients).

An AppServer can support clients using two different session models, which are dependent on its operating mode:

- **Session-managed** — Includes stateless, state-aware, and state-reset operating modes. These operating modes maintain a single session and connection between the AppServer and the client for all client requests.

- **Session-free** — Includes only the state-free operating mode. This operating mode maintains no session or connection between the AppServer and the client. In this operating mode, the AppServer treats any request from a client as independent from all other requests. As a result, some SESSION handle attributes in the ABL application have no valid meaning, or they provide different information.

The AppServer supports these session models for supported AppServer clients.

For more information, see *OpenEdge Application Server: Developing AppServer Applications*.

AppServer features

The AppServer broker handles the following activities:

- Optionally registers with a NameServer the application services that it provides for access by clients

- Manages connections between clients and a pool of AppServer agents

- Maintains the status of each AppServer agent in its pool and dynamically scales the number of servers according to changing demand

- When configured for stateless or state-free operating mode, dispatches remote requests to AppServer agents

- When configured for state-reset or state-aware operating mode, connects each client to a dedicated AppServer agent that handles all remote requests for that client until the connection is terminated

Working with the AppServer

To work with an AppServer instance, double-click the AppServer folder in the management console's list frame. There is one pre-defined default AppServer (asbroker1).

You can perform the following AppServer actions:

- Configuring an AppServer on page 116
- Creating an AppServer on page 135
- Starting and stopping an AppServer on page 135
- Adding or trimming AppServer agents on page 136
- Viewing the status of an AppServer on page 137
- Viewing the AppServer log files on page 142
AppServer configuration properties

An AppServer starts up by using the configuration information stored in the `ubroker.properties` file. The file contains the property settings for all server resources on the host where OpenEdge is installed.

The `ubroker.properties` file

The `ubroker.properties` file consists of a hierarchical structure of configuration entities, in which parent entities provide configuration information that you can override or extend in each child entity. Each configuration entity has a name that begins the entity definition, and the definition contains configuration settings for one or more product instances.

For example, property settings for an instance of an AppServer are determined hierarchically, as shown in the following table.

Table 53: Ubroker.properties file structure

<table>
<thead>
<tr>
<th>Configuration entity name</th>
<th>Configuration entity function</th>
</tr>
</thead>
<tbody>
<tr>
<td>[UBroker]</td>
<td>Defines default property settings for various OpenEdge brokers, including all AppServer brokers.</td>
</tr>
<tr>
<td>[UBroker.AS.product-instance-name]</td>
<td>Defines property settings for a particular instance of an AppServer. The <code>ubroker.properties</code> file can contain several of these entities, each with a unique <code>product-instance-name</code>.</td>
</tr>
</tbody>
</table>

| Note: The `product-instance-name` is unique and case-sensitive. It can contain alphanumeric and special characters, except period (".") colon (":") or square brackets ("["]"). The name must not contain blank spaces. |

Parent entities provide default values for all of their child entities. However, at any child level a redefinition of any value supersedes the default value of its parent. All children, from their definition level down, inherit this new value.

Like the `ubroker.properties` file, a similar file, `conmgr.properties`, stores all the properties for each instance of an OpenEdge database. For more information about OpenEdge Management and OpenEdge Explorer and the database configuration properties, see Configuring OpenEdge Databases on page 27.

The `ubroker.properties` file resides in the properties subdirectory of the OpenEdge install directory. The file is editable and any changes that you make to it are automatically reflected in the management console.

| Note: Although making manual edits to the `ubroker.properties` file is possible, Progress Software recommends that you use the Mergeprop utility or the management console to make property changes. For more information about the Mergeprop utility, see OpenEdge Getting Started: Installation and Configuration. |
Configuring an AppServer

From the management console, you can view or modify an AppServer broker's default configuration properties in the following overall categories:

- Broker
- Agent
- SSL
- Messaging
- Environment variables

You can also create a new AppServer instance, view an instance's status, delete an instance you no longer need, or view the instance's log files.

Each of these tasks is described in the sections that follow.

Viewing or modifying AppServer broker properties

You can view or modify AppServer broker properties in the following categories: general properties, owner information, controlling NameServer information, AppService Name List information, logging setting details, and advanced features.

To view or modify AppServer broker properties:

1. From the OpenEdge Management console menu, click **Resources**. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose properties you want to view or modify. For example, type the default AppServer broker name `asbroker1` in the filter field or select AppServer as the selection in the **Type** drop-down.
3. From the list of AppServer brokers, click the AppServer broker whose properties you want to view or modify. The **AppServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **AppServer Configuration** page appears with the **Broker** properties in focus. From this page, you can:
   - View a read-only display of the AppServer broker properties as described in **AppServer broker properties** on page 116.
   - Click **Edit** to modify the broker properties. For details, see **Editing an AppServer configuration** on page 135.

AppServer broker properties

**AppServer broker General properties**

**Table 54: AppServer Broker General properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto start</td>
<td>Indicates whether the server automatically starts when the controlling AdminServer starts.</td>
</tr>
</tbody>
</table>
### AppServer broker Owner Information properties

Table 55: AppServer Broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>The username of an account that has system-administrative rights.</td>
</tr>
<tr>
<td>Group name</td>
<td>(UNIX only) The name of the group.</td>
</tr>
<tr>
<td>Password</td>
<td>The username account password in Windows.</td>
</tr>
<tr>
<td>Password Confirm</td>
<td>Confirmation of the username account password. This field appears only if</td>
</tr>
<tr>
<td></td>
<td>you are modifying the properties.</td>
</tr>
<tr>
<td></td>
<td>Note that this field does not appear if the AdminServer is running on a</td>
</tr>
<tr>
<td></td>
<td>non-Windows host machine.</td>
</tr>
</tbody>
</table>

### AppServer broker Controlling NameServer properties

Table 56: AppServer Broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>Indicates whether or not to register the AppServer with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>The name of the NameServer with which this AppServer registered. If you did not select to register the broker with a NameServer, the field is disabled.</td>
</tr>
</tbody>
</table>
How the broker specifies its hostname if it is registering with a controlling NameServer. This hostname information is passed onto a client application when it attempts to connect to an application service that the AppServer broker supports. The choices are:

- **Register-IP** — (Default) This setting is the most efficient mechanism and can be used in most cases. It registers with the IP address of the machine where the broker is located.

- **Register-LocalHost** — The broker registers with the hostname of the machine that it runs on. Use this setting when the broker runs on a machine with a single hostname and more than one IP address.

- **Register-HostName** — The broker registers with the values specified in the HostName property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.

### AppServer Broker AppService Name List properties

Table 57: AppServer Broker AppService Name List properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application service names</td>
<td>A list to which you can add or delete application service names</td>
</tr>
<tr>
<td>Supports default service</td>
<td>Indicates whether or not the AppServer broker supports the default service</td>
</tr>
</tbody>
</table>

### AppServer Broker Logging Setting properties

Table 58: AppServer Broker Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The broker log filename.</td>
</tr>
<tr>
<td>Broker logging level (dynamic property)</td>
<td>A value that specifies the amount of information that is written to the broker log. The following values are available:</td>
</tr>
<tr>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
<td></td>
</tr>
<tr>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>• Basic — Broker logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• Verbose — Logging entry types determine the logged information, but it is typically more information than Basic.</td>
</tr>
<tr>
<td></td>
<td>• Extended — Logging entry types determine the logged information, but it is typically more information than Verbose.</td>
</tr>
</tbody>
</table>

You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.

Append to broker log file

Indicates whether or not a new broker log file should be created when the AppServer is started.

Broker logging entry types (dynamic property)

A comma-separated list of the following valid entry types for agent logging:

• UBroker.AutoTrim — Logs messages about automatic agent thread trimming by the broker based on the Auto trim timeout property. Information is logged at logging level Basic and higher.

• UBroker.Basic — Logs messages about broker and agent startup and shutdown at the following logging levels:
  • Basic — Logs NameServer registration and connections from clients.
  • Verbose — Logs information about all the property values read from the ubroker.properties file for that broker instance, and more information on the connection from clients.

• UBroker.ThreadPool — Logs messages about the pool of threads managed by the broker. These threads are used to control the client requests and the agent processes. This entry type logs messages when adding, removing, and communicating with these threads. Information is logged at logging level Basic and higher.

You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.

Turn on the following logging entry types only at the request of Progress Software Corporation Technical Support:

• UBroker.ClientFSM — Logs messages about the state of the client thread as it processes the requests, from the moment it receives a connection request to the end of the request. Information is logged at logging level Basic and higher.

• UBroker.ServerFSM — Logs messages about the state of the agent thread as it processes the requests, from the moment it receives a connection request to the end of the request. Information is logged at logging level Basic and higher.
• **UBroker.ClientMsgStream** — Logs messages about messages exchanged between the broker and client threads in binary format. Information is logged at logging level **Basic** and higher.

• **UBroker.ServerMsgStream** — Logs messages about messages exchanged between the broker and agent threads in binary format. Information is logged at logging level **Basic** and higher.

• **UBroker.ClientMsgQueue** — Logs messages about the client thread as it processes queued messages. Information is logged at logging level **Basic** and higher.

• **UBroker.ServerMsgQueue** — Logs messages about the agent thread as it processes queued messages. Information is logged at logging level **Basic** and higher.

• **UBroker.ClientMemTrace** — Traces messages being processed by the client threads. Information is logged at logging level **Basic** and higher.

• **UBroker.ServerMemTrace** — Traces messages being processed by the agent threads. Information is logged at logging level **Basic** and higher.

• **UBroker.Stats** — Logs statistics about the number of requests, plus maximum, minimum, and average request wait and duration times. Information is logged at logging level **Basic** and higher.

• **UBroker.All** — Logs all Unified Broker entry types.

You can specify a separate logging level for each log entry type and use a wildcard (*) to specify multiple entry types. For example:

```
UBroker.Basic:3,UBroker.ThreadPool:4,
UBroker.Client*
```

For details, see the information about client logging in *OpenEdge Deployment: Startup Command and Parameter Reference*. 
**Description**

**Property**

**Broker log file threshold size**

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

**Maximum number of broker log files**

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of broker log files to keep. The specified number represents the maximum number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the Broker log file threshold size, the client process renames it and creates a new log file. The file is renamed as follows, where ####### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:

`filename.#######.extension`

---

**AppServer Broker Advanced Features properties**

**Table 59: AppServer Broker Advanced Features properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum client instances</strong></td>
<td>The maximum number of client connections that the broker can support concurrently. The default is 512, which is high enough to ensure that the number of client connections is virtually unlimited. Realistically, however, the system-level resources needed to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for &quot;out of space&quot; or &quot;OutOfMemory&quot; you might need to reduce the maximum number of client connections to a more reasonable value. Note, however, that lowering this value can cause some client requests to be rejected (&quot;Exceeded Max Clients&quot;). Configuring and starting multiple brokers to handle higher client loads alleviates the problem.</td>
</tr>
<tr>
<td><strong>Priority weight (0-100) (dynamic property)</strong></td>
<td>An integer value between 0 and 100 that influences the share of the workload that the selected AppServer receives. The larger the value, the heavier the load that is distributed to the server. The NameServer distributes client requests across all AppServers that have the same application service in proportion to the Priority weight value. The default is 0. You can update this property dynamically. Any changes will affect all current and new brokers.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Registration retry (in seconds)</td>
<td>After registering with its controlling NameServer, the AppServer periodically sends &quot;keep-alive&quot; messages to the NameServer to let the NameServer know that it is still active. The Registration retry value is the number of seconds that pass between &quot;keep-alive&quot; messages. The default is 30.</td>
</tr>
<tr>
<td>Server startup timeout (dynamic property)</td>
<td>Specifies the amount of time, in seconds, that the broker waits for an active agent to become available before starting a new agent process. The default is 3. You can update this property dynamically. Any changes will affect only new agents that start after the property value has change; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td>Request timeout (dynamic property)</td>
<td>Specifies the amount of time, in seconds, that the broker will wait for an agent to become available for processing a request. The broker waits for this period only if the Maximum servers setting has been reached. After the timeout has expired, the client receives a &quot;no servers available&quot; error message. The default is 15. You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td>Auto trim timeout (dynamic property)</td>
<td>The amount of time, in seconds, that the broker waits before automatically trimming the number of running agents. The broker keeps track of the maximum number of agents that are simultaneously busy during the interval you specify. At the end of the interval, the broker attempts to trim the number of agents to match the Maximum servers for the interval, or the Minimum servers, whichever is greater. You can disable this feature by setting the timeout to zero (0). The default Server Auto-trim Interval is 1800 seconds. Progress Software recommends that you use this default setting to avoid unnecessary process management that might result from frequent starting and trimming of agents. You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| TCP/IP version | Internet Protocol for network communication:  
  - IPv4 — Accepts only IPv4 connections  
  - IPv6 — Allows IPv4 and IPv6 connections using mapped address (where supported)  
  The default is IPv4.  
  You must also set the jvmArgs property in the ubroker.properties file to override the default behavior of the JVM for IPv6 connections.  
  jvmArgs=-Djava.net.preferIPv4Stack=false -Djava.net.preferIPv6Addresses=true. |
| jvmargs | Java System Properties. |
| Enable dynamic property updates | Allows dynamic changes to occur to several of the instance's properties, without requiring you to shut down the instance.  
  This property is disabled by default.  
  **Note:** Any property that is dynamic is labeled as such in this document. |
| AppServer Keepalive | Allows the AppServer to recognize that a client bound to it is no longer connected. |
| serverASK Activity Timeout | Specifies the number of seconds with no activity that the AppServer waits before sending the client a message requesting that the connection be maintained.  
  The property is used only if the AppServer Keepalive property is set to allowServerASK. |
| serverASK Response Timeout | Specifies the number of seconds with no activity after the AppServer sends the client a message (requesting that the connection be maintained) before the client is disconnected. This property is used only if the AppServer Keepalive property is set to allowServerASK. |
| Enable debugging through the broker (dynamic property) | Specifies whether or not the debugger is enabled; select the check box to enable it.  
  If you disable this property while the debugger is connected, the debugger connection terminates.  
  **Note:** The Broker debugger port number property must also be set to a valid value for the debugger to be properly enabled.  
  This property is disabled by default. |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Broker debugger port number</strong>(dynamic property)</td>
<td>Specifies the port number that the broker debugger will use when it starts. You can choose any valid value; if the port you specify is already in use, the debugger service will fail to start. If you change the port number while the debugger service is active, the debugger service will be shut down and restarted using the new port number. The default port number is 3099.</td>
</tr>
<tr>
<td><strong>Enable SSL for broker debugger</strong> (dynamic property)</td>
<td>Specifies whether the broker debugger should accept only SSL/TLS-encrypted connections. If you enable this property, the Progress Developer Studio debugger will be required to establish an SSL/TLS session. If you enable or disable the property while the debugger service is active, the debugger will be disconnected. The debugger service will be shut down and then restarted using the new selection. This option is disabled by default.</td>
</tr>
<tr>
<td><strong>Use the broker's private key/digital certificate for debugger</strong> (dynamic property)</td>
<td>When enabled, indicates that, when using SSL, the broker debugger service should use the same private key as used by the AppServer agents. If this property is enabled, then the <strong>Debugger private key/digital certificate alias name</strong> and <strong>Password to access the debugger's key/certificate</strong> property settings are ignored. This property is enabled by default.</td>
</tr>
<tr>
<td><strong>Debugger private key/digital certificate alias name</strong> (dynamic property)</td>
<td>The name of the private key that should be used for the debugger service when the agent's private key is not being used. This property must be defined if <strong>Enable SSL for broker debugger</strong> is enabled.</td>
</tr>
<tr>
<td><strong>Password to access the debugger's key/certificate</strong> (dynamic property)</td>
<td>The password required to access the private key specified by <strong>Debugger private key/digital certificate alias name</strong>. The default for this property is blank.</td>
</tr>
<tr>
<td><strong>Broker debugger passphrase</strong> (dynamic property)</td>
<td>An optional property that represents the passphrase required by the debugger to connect to the debugger service in the broker. If the debugger is connected to the debugger service when the passphrase is changed, the debugger remains connected. The next time the debugger attempts to connect to the debugger service, the new password is required. The default for this property is blank.</td>
</tr>
</tbody>
</table>
### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Broker debugger passphrase Confirm</strong></td>
<td>Confirms the passphrase provided in <strong>Broker debugger passphrase</strong>.</td>
</tr>
<tr>
<td><strong>Publish directory</strong></td>
<td>The name of the local directory where you want r-code to be placed when you are publishing from Progress Developer Studio. If you are publishing remotely from Progress Developer Studio, you must set this property.</td>
</tr>
</tbody>
</table>

### Viewing or modifying AppServer Agent properties

You can modify the following Agent properties for the AppServer Broker: general properties, logging setting details, pool range, and advanced features.

To view or modify AppServer Agent properties:

1. From the OpenEdge Management console menu, click **Resources**. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose properties you want to view or modify. For example, for the default AppServer broker name, type `asbroker1` in the filter field or select AppServer from the **Type** drop-down.
3. From the list of AppServer broker instances, click the AppServer broker whose properties you want to view or modify. The **AppServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **AppServer Configuration** page appears with the **Broker** properties in focus.
5. Click the **Agent** tab. From this page, you can do the following:
   - View a read-only display of the AppServer Agent properties as described in **AppServer Agent properties** on page 125.
   - Click **Edit** to modify the broker properties. For details, see **Editing an AppServer configuration** on page 135.

### AppServer Agent properties

#### AppServer Agent General properties

#### Table 60: AppServer Agent General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server executable file</strong> (dynamic property)</td>
<td>Either the default agent executable pathname or a different agent executable file of your own choosing. You generally need to specify a nondefault value only if you generated a new executable using the OEBUILD utility. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
</tbody>
</table>
### Server startup parameters (dynamic property)

The OpenEdge startup parameters that you want to specify to start each agent.

You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

**Note:** It is recommended to set the Stack size startup parameter value as -s 128 in the Server startup parameters field, when working with databases having large schemas.

### PROPATH

The search path that agents use to locate ABL procedures that they execute. Specify a list of directory pathnames in the form of an OpenEdge PROPATH. Make sure that you copy all of your AppServer procedures to one or more of the directories in the PROPATH. You can enter the pathname directly.

### Minimum port number (dynamic property)

The minimum TCP/IP port number from a range that each agent process can listen on. When each agent starts, it allocates a port that is not being used from the specified range up to the Maximum port number setting.

You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

### Maximum port number (dynamic property)

The maximum TCP/IP port number from a range that each agent can listen on. When each agent starts, it allocates a port that is not being used from the specified range down to the Minimum port number setting.

You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

### Flush statistical data (dynamic property)

How often the data is flushed by the agent to the broker. This is expressed as the number of remote procedure calls between flushes.

You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.

### AppServer Agent Logging Setting properties

#### Table 61: AppServer Agent Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The log file name.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Server logging level** (dynamic property) | A value that specifies the amount of information to be written to the agent log. Each logging level name has the indicated numeric value. You can select from the following values in the drop-down list:  
  • None — Log no entries. This is equivalent to turning logging off.  
  • Error Only — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.  
  • Basic — Server logging entry types determine the logged information. Each entry type generates at least some output. This is the default.  
  • Verbose — Logging entry types determine the logged information, but it is typically more information than Basic.  
  • Extended — Logging entry types determine the logged information, but it is typically more information than Verbose.  
  You can update this property dynamically. Any changes will affect all current and new brokers and/or agents. |
| **Server logging entry types** (dynamic property) | A comma-separated list of valid entry types for agent logging:  
  • 4GLTrace — Turns on the logging for the execution of the following ABL statements: RUN, FUNCTION (user-defined functions), PUBLISH, and SUBSCRIBE. Logs information at Basic or higher.  
  • ASDefault — Combines the ASPlumbing and DB.Connects log entry types. This is the default value for AppServer agents.  
  • ASPlumbing — Turns on logging for different actions depending on the logging level specified:  
    • Basic — Logs messages when an AppServer agent starts and stops, and when client processes connect and disconnect from the AppServer agent.  
    • Verbose — Same as Basic, plus logging of the execution of the top level procedure (that is, the procedure that the client process asks the AppServer agent to execute on its behalf, but not any procedure that the top level procedure executes).  
    • Extended — Same as Verbose, plus messages about agent state, and status messages about read and write socket operations. Do not turn this on unless directed to do so by Progress Software Technical Support.  
    • DB.Connects — Turns on logging of database connections and disconnections. The messages include database name and user ID number. Logs information at logging level Basic or higher. |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• <strong>DynObjects.DB</strong> —</td>
<td>Turns on logging for dynamic database-related objects (for example, TEMP-TABLE or DATASET) as they are created and destroyed. Logs information at <strong>Basic</strong> or <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>• <strong>DynObjects.XML</strong> —</td>
<td>Turns on logging for dynamic XML-related objects (for example, x-document or x-noderef) as they are created and destroyed. Logs information at level <strong>Basic</strong> or <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td>• <strong>DynObjects.Other</strong></td>
<td>Turns on logging for dynamic objects that do not match the other dynamic object categories (for example, procedure or server socket) as they are created and destroyed. Logs information at <strong>Basic</strong> or <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>• <strong>ProEvents.UI.Char</strong></td>
<td>Turns on logging of keystroke events for printable characters. Logs information at <strong>Basic</strong> or higher.</td>
</tr>
<tr>
<td>• <strong>ProEvents.UI.Command</strong></td>
<td>Turns on logging of keystroke events for nonprintable characters at the following logging levels:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logs nonprintable keystrokes and a subset of Windows GUI events</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logs nonprintable keystrokes and all Windows GUI events</td>
</tr>
<tr>
<td>• <strong>ProEvents.Other</strong></td>
<td>Turns on logging of COM, asynchronous, and server socket events. Logs information at <strong>Basic</strong> or higher, beginning with the events for which you have written triggers.</td>
</tr>
<tr>
<td>• <strong>QryInfo</strong></td>
<td>Turns on database query information logging at the following logging levels, with the same amount of information logged for each level:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logs static queries and initial dynamic queries.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logs static queries and multiple instances of dynamic queries.</td>
</tr>
<tr>
<td>• <strong>SAX</strong></td>
<td>Turns on logging for various stages of execution using the ABL SAX parser.</td>
</tr>
<tr>
<td></td>
<td>You can specify a separate logging level for each log entry type and use a wildcard (*) to specify multiple entry types. For example:</td>
</tr>
<tr>
<td></td>
<td>4GLTrace:3, ASPlumbing:2,DynObjects.*</td>
</tr>
<tr>
<td></td>
<td>For details, see the information about client logging in <em>OpenEdge Deployment: Startup Command and Parameter Reference</em>.</td>
</tr>
</tbody>
</table>

| Append to server log file | Indicates whether or not a new agent log file should be created when the AppServer is started.                                           |
### AppServer configuration and administration

**Property** | **Description**
---|---
**Server log file threshold size** | A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the agent log file size other than what the operating system imposes.

**Maximum number of server log files** | The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of agent log files to keep. The specified number represents the maximum total number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the **Server log file threshold size**, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001: filename.######.extension

**Server watch dog interval** | The interval (in seconds) for the server log file watchdog thread. If the **Server log file threshold size** property is specified, the broker will roll over the server log file when its size is larger than the logThreshold value.

### AppServer Agent Pool Range properties

**Table 62: AppServer Agent Pool Range properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initial number of servers to start</strong></td>
<td>The number of agents you want the broker to start initially.</td>
</tr>
<tr>
<td><strong>Minimum servers</strong></td>
<td>The minimum number of agents before the broker starts additional agents. If you trim the number of agents below this value, the server starts any additional agents needed to maintain the specified minimum when the next client connects.</td>
</tr>
<tr>
<td><strong>Maximum servers</strong></td>
<td>The maximum number of agents that this server can have running at the same time.</td>
</tr>
</tbody>
</table>

### AppServer Agent Advanced Features properties

**Table 63: AppServer Agent Advanced Features properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4GL debugger enabled</strong></td>
<td>Indicates if the AppServer remote debugging facility is enabled. If the facility is enabled, ABL client applications have the ability to step into remote AppServer procedures and WebSpeed procedures have the ability to step into remote AppServer procedures.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Activate procedure</strong></td>
<td>The name of an ABL procedure that executes before the AppServer executes a remote procedure request for a client connection that is unbound. <strong>Activate procedure</strong> is available only for an AppServer running in stateless operating mode. You can update this property dynamically. Any changes will affect all current and new agents.</td>
</tr>
<tr>
<td><strong>Deactivate procedure</strong></td>
<td>The name of an ABL procedure that executes after the AppServer executes a remote procedure request for a client connection that is unbound. The <strong>Deactivate procedure</strong> property is available only for an AppServer running in stateless operating mode. You can update this property dynamically. Any changes will affect all current and new agents.</td>
</tr>
<tr>
<td><strong>Connect procedure</strong></td>
<td>The name of an ABL procedure that executes when a client attempts to connect to this AppServer. You can update this property dynamically. Any changes will affect all current and new agents.</td>
</tr>
<tr>
<td><strong>Disconnect procedure</strong></td>
<td>The name of an ABL procedure that executes when a client disconnects from this AppServer. You can update this property dynamically. Any changes will affect all current and new agents.</td>
</tr>
<tr>
<td><strong>Startup procedure</strong></td>
<td>The name of an ABL procedure that executes when an AppServer agent starts up for this AppServer. The <strong>Startup procedure</strong> property is not available for an AppServer running in state-reset operating mode. You can update this property dynamically. Any changes will affect only new agents; existing (running) agents are not updated.</td>
</tr>
<tr>
<td><strong>Shutdown procedure</strong></td>
<td>The name of an ABL procedure that executes in an AppServer agent when the AppServer agent shuts down. AppServer agents shut down when you use the trim feature and also when you shut down the AppServer. The <strong>Shutdown procedure</strong> property is not available for an AppServer running in state-reset operating mode. You can update this property dynamically. Any changes will affect all current and new agents.</td>
</tr>
</tbody>
</table>
### Startup procedure parameters

The parameters to pass the startup procedure when it executes. The **Startup procedure parameters** property takes a character string as an input parameter. For example:

```
DEFINE INPUT PARAMETER startup-data AS CHARACTER NO-UNDO.
```

You can set this parameter to any arbitrary value. If you do not specify a parameter in this field, the parameter is set to the unknown value (\?) when the AppServer agent executes the startup procedure.

You can update this property dynamically. Any changes will affect only new agents; existing (running) agents are not updated.

### Execution Time Limit

Sets the maximum time in seconds that a remote procedure can execute on a given AppServer.

The time limit applies to all remote procedures that execute on the AppServer.

The default value is zero, which indicates that no time limit exists.

---

### Viewing or modifying AppServer SSL properties

You can view or modify the following SSL properties for an AppServer Broker: general properties and advanced features.

To view or modify AppServer SSL properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose SSL properties you want to view or modify.
3. Click the Edit icon. The **AppServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **AppServer Configuration** page appears with the **Broker** properties in focus.
5. Click the **SSL** tab. From this page, you can:
   - View a read-only display of the AppServer SSL properties as described in **AppServer SSL properties** on page 131. These tables follow this procedure.
   - Click **Edit** to modify the SSL properties. For details, see **Editing an AppServer configuration** on page 135.

---

### AppServer SSL properties

The following sections describe the AppServer SSL properties.

#### AppServer SSL General properties

**Table 64: AppServer SSL General properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL client connections</td>
<td>If the check box is selected, specifies that all connections to this AppServer must use SSL tunneling.</td>
</tr>
</tbody>
</table>
### AppServer SSL Advanced Features properties

#### Table 65: AppServer SSL Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable SSL session caching</td>
<td>If the check box is selected, caching for the SSL client session is disabled.</td>
</tr>
<tr>
<td>SSL session cache timeout</td>
<td>The length of time, in seconds, that an SSL client session is held in the session cache, during which an SSL client can resume its session. The default is 180.</td>
</tr>
</tbody>
</table>

### Viewing or modifying AppServer Messaging properties

You can view or modify several AppServer Messaging properties.

To view or modify AppServer Messaging properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose messaging properties you want to view or modify.
3. Click the Edit icon. The **AppServer** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **AppServer Configuration** page appears with the **Broker** properties in focus.

5. Click the **Messaging** tab. From this page, you can:
   - View a read-only display of the AppServer Messaging properties as described in **AppServer Messaging properties** on page 133.
   - Click **Edit** to modify the AppServer Messaging properties. For details, see **Editing an AppServer configuration** on page 135.

**AppServer Messaging properties**

The following table lists the AppServer Messaging properties.

**Table 66: AppServer Messaging properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SonicMQ ServerConnect enabled</strong></td>
<td>Allows the asbroker or wsbroker agent to start a personal SonicMQ Adapter instead of going to a separate one running elsewhere.</td>
</tr>
<tr>
<td><strong>SonicMQ ServerConnect broker log filename</strong></td>
<td>A valid path and file name.</td>
</tr>
<tr>
<td><strong>Broker logging level</strong> (dynamic property)</td>
<td>A value that specifies the amount of information to be written to the SonicMQ ServerConnect broker log. You can select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Broker logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>You can update this property dynamically.</td>
<td>Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td><strong>Append to broker log file</strong></td>
<td>If the check box is selected, Indicates that a new broker log file should be created, even if the broker log file already exists, when the SonicMQ ServerConnect broker is started.</td>
</tr>
<tr>
<td><strong>SonicMQ ServerConnect server log filename</strong></td>
<td>A valid path and file name.</td>
</tr>
</tbody>
</table>
### Setting or deleting AppServer broker environment variables

You can set or delete environment variables. For the AppServer, environment variables that appear in the list are scoped to:

- The process in which the AppServer broker executes
- Every AppServer agent that the AppServer broker starts

To set or delete AppServer broker environment variables:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose environment variables you want to set.
3. Click the Edit icon. The AppServer details page appears.
4. In the Command and control section of the page, click Configuration. The AppServer Configuration page appears.
5. Click Edit.
6. Click Environment Variables.
7. Enter each variable name and then provide its value in the format name=value. Alternatively, you can remove an existing variable by deleting the name and value pair.
8. Click Save when you finish setting (or deleting) the variables and their values.
Editing an AppServer configuration

You can edit the configuration properties for an AppServer resource from the management console.

To edit AppServer configuration properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose configuration properties you want to edit.
3. Click the Edit icon. The AppServer details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The AppServer Configuration page appears.
5. Click **Edit**, and make the changes you want to the properties. (For details about the properties, see AppServer broker properties on page 116, AppServer Agent properties on page 125, AppServer SSL properties on page 131, AppServer Messaging properties on page 133 and AppServer Internet Adapter (AIA) properties on page 144.)
6. Click **Save**. Any changes you make are also reflected in the ubroker.properties file.

Creating an AppServer

OpenEdge Management and OpenEdge Explorer provide a sample AppServer instance (asbroker1) with a default configuration. Each new instance you create uses the same default configuration, which you can edit.

To create an AppServer:

1. From the drop-down for **Resources** on the management console menu, click **New OpenEdge Resource > AppServer**.
   
The AppServer Configuration page appears.
2. Type the name of the new AppServer in the field provided.

   **Note**: The AppServer name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured AppServer names.

3. Click **Save**. The AppServer Configuration page appears, allowing you to configure the AppServer's properties.

   Each new AppServer instance you create uses the default configuration. However, the port number must be unique for each AppServer instance in order for the instance to operate properly. Once you create a new instance, you are prompted to change the port number.

Starting and stopping an AppServer

From the management console, you can start or stop either a local or a remote AppServer. You must first connect to the AdminServer that manages the AppServer you want to start.

You can also choose to start an AppServer automatically when the AdminServer starts.
Note: Prior to starting an AppServer using SonicMQ ServerConnect, you must first configure the AppServer as SonicMQ ServerConnect enabled. To stop a remote AppServer, access the corresponding local AppServer instance selected in the AdminServer of the host where it runs.

To start or stop an AppServer, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

Starting an AppServer automatically

You can set an AppServer to start automatically when the AdminServer starts.

To start an AppServer automatically:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance you want to start automatically.
3. Click the Edit icon. The AppServer details page appears.
4. In the Command and control section of the page, click Configuration. The AppServer Configuration page appears.
5. Click Edit.
6. Select the Auto start option by clicking in the check box.
7. Click Save.

Adding or trimming AppServer agents

At run time, you can increase the number of AppServer agents running and available to service AppServer client requests by having the AppServer add a specified number of AppServer agents to its process pool.

You can also trim running AppServer agents, down to the Minimum Servers property setting, from the management console.

More information about trimming AppServer agents

To better understand the use of the Agent Auto-trim feature, consider a scenario involving a broker serving a client/server application running in stateless mode. Assume that the following property values are set:

- Minimum Agent/Server Instances: 10
- Initial Agent/Server Instances to Start: 10
- Maximum Agent/Server Instances: 50
- Agent/Server Auto-trim Interval: 1800

When the broker starts, it starts 10 agents. This also marks the start of the first Agent/Server Auto-trim Interval. Assume that the broker is idle for 30 minutes (it is just before people arrive for work, for instance). At the end of the 30-minute interval, the highest number of agents that were busy at the same time would be zero. (Since there were no requests made of any of the agents, none of the agents was busy.) However, since zero is less than the Minimum Agent/Servers Instances value of 10, no automatic trimming takes place. This marks the start of the second interval.
As the second interval starts, employees begin their workday. There is a sudden demand for agents; in fact, the broker receives so many requests that it must start 20 additional agents. Assume that the high volume of workload, and demand on the agents, continues through the second interval. At the interval's end, the maximum number of busy agents is 30, which is the number the broker scaled up to in response to high demand. Thirty agents are running, and the maximum busy level is 30. No trimming occurs.

During the next interval, the request load diminishes somewhat. At the end of the interval, only 20 agents are concurrently busy. The maximum number of busy agents for the interval remains at 30, since 30 were busy at the start of the interval. Thirty agents are running, and the maximum busy level is 30. No trimming occurs.

Assume that all employees have to attend a company-wide meeting that begins midway through the interval. The highest number of agents busy at that same time was 20. At the end of the interval, there are 30 agents running, with zero agents currently busy. Since the maximum busy level is 20 (at the start of the interval), and the number of running agents is 30 (all are available since everyone is at the company meeting), the broker attempts to trim back the number of agents to 20.

Assume that the company meeting ran for another half-hour, and the entire next interval elapsed without any requests. The maximum busy agents would be zero, with 20 currently running. Since the Minimum Agent/Server Instances property value is set to 10, the broker attempts to trim back to 10 agents.

Keep in mind that the lower you set the Agent/Server Auto-trim Interval value, the more quickly the broker reacts to a drop in the workload. Under a reasonably steady work load, the broker should trim the number of running agents to the number needed. However, if the workload fluctuates a great deal and the Agent/Server Auto-trim Interval is set too low, the broker may trim agents that it must then restart. As stated earlier, repetitive trimming and restarting causes the system to waste resources on unnecessary process management.

Adding or trimming AppServer agents

To add or trim AppServer agents:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance to which you want to add or trim agents.
3. Click the Edit icon. The AppServer details page appears.
4. If the AppServer is not started, click Broker Control. Then click Start AppServer.
5. When the AppServer is running, click the backlink for the AppServer broker instance on the breadcrumb trail to return to the details page.
6. In the Command and control section of the page, click Server Pool Control.
7. Review the Server pool initial configuration, the Servers state, and the Server pool summary information. (You can view detailed information about an agent by clicking on its PID.)
8. To add or trim servers, choose the corresponding option in the Add/Trim dropdown menu, and type the number of servers in the box provided.
9. Click Submit. A confirmation message appears.

Viewing the status of an AppServer

You can view the status of any running AppServer instance through the management console. Status and/or statistical details appear in the following different views:

- The AppServer instance's Details page
- The Broker Control link on the AppServer instance's Details page
• The Broker PID link within the instance’s AppServer Control page
• The Server Pool Control link on the AppServer instance’s Details page
  You can also add and trim AppServer agents from the Server Pool Control page

**Viewing AppServer instance status from the Details page**

To view the status of an AppServer instance from the Details page:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose status you want to view.
3. Click the Edit icon. The AppServer details page appears and displays the following information:
   • **Host** — The name of the host machine
   • **Broker** — Whether the broker is active or not running
   • **Operating mode** — One of four possible modes can be reported: Stateless, State-reset, State-aware, and State-free
   • **Broker statistics available** — Whether broker statistics are being collected
   • **Servers available** — The number of AppServers running and available to fulfill a connection request from a client to an AppServer through this broker when the broker’s status is ACTIVE
   • **Should register with NameServer** — The status of True or False to indicate whether or not the broker resource is registered with a NameServer

**Viewing AppServer instance status from the Broker Control link**

To view the status of an AppServer instance from the Broker Control link:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance whose status you want to view.
3. Click the Edit icon. The AppServer details page appears.
4. In the Command and control section of the page, click Broker Control. The AppServer Control page for that instance appears and displays the following information:
   • **Broker name** — The name of the selected AppServer broker.
   • **Host** — The name of the host machine.
   • **Port** — The port number.
   • **Broker PID** — The broker’s process identification number (PID).
   • **Status** — The running status of the broker. Possible values are:
     • ACTIVE — The broker is currently running.
     • Not Running — The broker is not currently running.
   The broker can also report Starting and Shutting Down values; however, depending on the speed of the machine on which your management console is running, you may not see these intermediary states.
• **Operating mode** — One of four possible modes can be reported: **Stateless, State-free, State-aware,** or **State-reset.**

• **Broker statistics available** — Whether collected broker statistics are available.

• **Enabled** — Whether the broker is enabled.

• **Collect Statistics** — Whether statistics are being collected.

### Viewing AppServer instance status from the Server Pool Control link

To view the status of an AppServer instance from the Server Pool Control link:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the AppServer broker instance whose status you want to view.

3. Click the Edit icon. The **AppServer** details page appears.

4. In the **Command and control** section of the page, click **Server Pool Control**. The **Server Pool Control** page for that instance appears and displays the following information:

   • **Initial number of servers to start** — The value referenced when the AppServer broker starts AppServers.

   • **Minimum servers** — The minimum number of AppServers that must be simultaneously running before the AppServer broker starts additional servers. The broker strives to maintain this specified minimum. If at any time the number of servers falls below the specified minimum, the broker automatically starts the additional servers needed to maintain the minimum.

   If you set a trim value that requires trimming the number of servers to below the number specified for this field, a message appears.

   • **Maximum servers** — The maximum number of AppServer processes that can be running simultaneously. Add requests you initiate that will exceed the specified maximum are not fulfilled.

   • **Active servers** — The number of AppServers currently running.

   • **Busy servers** — The number of AppServers handling ABL client requests.

   • **Locked servers** — The number of AppServers servicing a bound connection. (This state applies to a stateless AppServer.)

   • **Available servers** — The number of AppServers available to handle broker requests.

A server pool summary provides details about the following:

• **PID** — The process identifier for this AppServer. Click the PID number to display a detail page that provides specific information about this server process and, as necessary, to kill the process. See the section about killing an AppServer process in *OpenEdge Management: Servers, DataServers, Messengers, and Adapters* for more information.

• **State** — The current execution state of the agent.

• **Port** — The TCP/IP port number used by the agent.

• **nRq** — The number of messages sent to the agent.

• **nRcvd** — The number of messages received by the agent.

• **nSent** — The number of requests sent by the agent.

• **CPU Use** — The percentage of CPU user and system time consumed by a process.

• **Memory Use** — The amount of virtual memory (in Kbytes) consumed by a process.
Viewing AppServer process details

You can access real-time details and statistics that provide you with snapshot information about a running AppServer process at the point you access this information from the Control page.

To view process details:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance to which you want to add or trim agents.
3. Click the Edit icon. The AppServer details page appears.
4. In the Command and control section of the page, click Server Pool Control.
5. Click a PID link. The following sections appear:
   - The Process summary section identifies the Process name and Process start time. User id and Group id values appear when UNIX-based data is shown. The Parent pid provides the identifier number associated with the process that spawned this current process.
   - The Process statistics section presents details about the database's real-time operational status. Values presented without parentheses identify that the processing time determined since the last scheduled polling interval, as noted, has occurred. Values presented within parentheses have been calculated based on information obtained since the start of the process.

The following table identifies and describes the fields of information presented in the Process statistics section.

Table 67: Process statistics data

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resident size</td>
<td>The physical size of the process as defined by the host system.</td>
</tr>
<tr>
<td>Virtual size</td>
<td>The virtual size of the process as defined by the host system.</td>
</tr>
<tr>
<td>CPU</td>
<td>The percentage of time spent using the CPU in either the user or kernel mode since the last scheduled poll.</td>
</tr>
<tr>
<td>Weighted CPU</td>
<td>The percentage of time spent using the CPU in either the user or kernel mode since the last scheduled poll divided by the number of CPU processors on the system. This value appears only when there is more than one CPU process on the system where the process is running.</td>
</tr>
<tr>
<td>User time</td>
<td>The amount of CPU time spent in the user mode since the last scheduled poll.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Kernel time</td>
<td>The amount of CPU time spent in the kernel mode since the last scheduled poll.</td>
</tr>
<tr>
<td>Process time</td>
<td>The sum of the values that appear in the User time and Kernel time fields.</td>
</tr>
</tbody>
</table>

**Stopping an AppServer agent**

When you stop an AppServer agent, the related process is completed and then stopped.

To stop an AppServer agent:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance which you want to stop.
3. Click the required instance icon. The AppServer details page appears.
4. In the Command and control section of the page, click Server Pool Control.
5. Click a PID link.
6. Click Stop. Click OK when you get a confirmation message.

**Killing an AppServer agent**

When you kill an AppServer agent, the agent is stopped immediately without waiting for any related processes to complete.

To kill an AppServer agent:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer broker instance which you want to kill.
3. Click the required instance icon. The AppServer details page appears.
4. In the Command and control section of the page, click Server Pool Control.
5. Click a PID link.
6. Click Kill. Click OK when you get a confirmation message.

**Deleting an AppServer**

You can delete any AppServer instance listed in the management console as long as the instance is not running. (The Delete button is not available if the instance is running.) If the instance is running, first stop it and then delete it.

To delete an AppServer instance:
1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter, or search for, the AppServer broker instance you want to delete.

3. Click the AppServer broker instance. The **AppServer** details page appears.

4. Click **Delete**, and then click **OK** to confirm the deletion.

The **OpenEdge Management Resources** page appears. A set of next-step options that are related to resources are available for your use.

### Viewing the AppServer log files

The AppServer log file viewer allows you to examine the contents of a log file through an HTML interface. For details, see the section about using the AppServer log file viewer in *OpenEdge Management: Servers, DataServers, Messengers, and Adapters*.

### For additional AppServer information

You can find additional information about using OpenEdge Management with the AppServer in *OpenEdge Management: Servers, DataServers, Messengers, and Adapters*. See that document for details, including specifics about the following:

- Working with the AppServer log file monitor and viewer
- Killing an AppServer process
- Listing AppServer Client connections

### AppServer Internet Adapter configuration and administration

With the AppServer Internet Adapter (AIA) you can make AppServer or SonicMQ Adapter application services available over the Web to ABL applications, and you can make AppServer application services available over the Web to .NET and Java Open Clients. The AIA is a Java Servlet that is invoked by a Java Servlet Engine (JSE).

When you use the URL syntax for one of the following connection protocols to connect standard ABL clients (including AppServer and WebSpeed agents, and WebClients) or .NET and Java Open Clients to an AppServer, or to connect standard ABL clients to a SonicMQ Adapter:

- **HTTP** — Directs the client to connect through an AIA instance using an unsecured Internet connection to the Web server that hosts the AIA
- **HTTPS** — Directs the client to connect through an AIA instance using a Secure Sockets Layer (SSL) Internet connection to the Web server that hosts the AIA
Note: You can also use the AppServer protocol to allow a client to connect directly to an AppServer or a SonicMQ Adapter using the TCP/IP protocol with or without SSL tunneling. This is the preferred option for client access over an intranet.

However, if you want to use a secure Internet connection to the Web server and also secure the connection between the AIA and the AppServer (or SonicMQ Adapter), you can configure the AIA to use SSL for its own connection to the AppServer (or SonicMQ Adapter).

Working with the AppServer Internet Adapter

To work with an AIA instance, double-click the AppServer Internet Adapter folder from the management console’s list pane. There is one pre-defined default AppServer Internet Adapter (Aia1).

The following sections describe these AppServer Internet Adapter actions:

- Configuring an AppServer Internet Adapter on page 143
- Editing an AppServer Internet Adapter configuration on page 149
- Creating an AppServer Internet Adapter on page 149
- Deleting an AppServer Internet Adapter broker on page 151
- Viewing the AppServer Internet Adapter log file on page 150

Configuring an AppServer Internet Adapter

You can view or modify the following configuration properties of an AppServer Internet Adapter broker from the management console:

- General
- Controlling NameServer
- Logging Setting
- SSL
- Advanced Features

You can also create a new AppServer Internet Adapter instance, delete an instance you no longer need, or view its log file.

To view or modify AppServer Internet Adapter configuration properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer Internet Adapter broker instance whose configuration properties you want to view or modify.
3. Click the Edit icon. The AppServer Internet Adapter details page appears.
4. In the Command and control section of the page, click Configuration. The AppServer Internet Adapter Configuration page appears with the General properties in focus. From this page, you can:
   - View a read-only display of the AppServer Internet Adapter instance’s properties, as described in AppServer Internet Adapter (AIA) properties on page 144. All of these tables follow this procedure.
AppServer Internet Adapter (AIA) properties

The following sections describe the AppServer Internet Adapter (AIA) properties.

AIA General properties

Table 68: AIA General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle connection timeout</td>
<td>Timeout value, in seconds, between an AppServer client and an AIA instance. If a connection is idle for more than the specified timeout value, then the AIA instance automatically disconnects from the AppServer. The default is 3600.</td>
</tr>
<tr>
<td>Secure port</td>
<td>Identifies the port on which the Web server listens for requests that specify the HTTPS protocol. This value is used only when the HTTPS enabled property check box is selected. If you do not specify a value for this property, and the HTTPS enabled property check box is selected, the default value for Secure port is 443.</td>
</tr>
<tr>
<td>HTTPS enabled</td>
<td>Determines how an AppServer connection using the AIA connects to the AIA’s Web server host. If you select this option, clients must use HTTPS to connect and send encrypted data across the Internet to the Web server. If you do not select this option, clients can use HTTP to connect and send data across the Internet without encryption. When the HTTPS enabled check box is selected, if an AppServer client uses HTTP to connect to an AppServer, the AIA returns an HTTPS redirection URL to the client. The client automatically retries the connection using the HTTPS URL. The default setting for this check box is enabled.</td>
</tr>
</tbody>
</table>

AIA Controlling NameServer properties

Table 69: AIA Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect using NameServer</td>
<td>A check box that indicates whether to connect to an AppServer using a Controlling NameServer.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Controlling NameServer</strong></td>
<td>Identifies the NameServer that the AIA uses to find a broker that supports the correct application services.</td>
</tr>
</tbody>
</table>
| **Minimum NameServer client port** and **Maximum NameServer client port** | Properties used to limit the range of ports of an AIA communicating with the NameServer via UPD. This is important when the AIA and NameServer are separated by a firewall.  
  The value for these two properties must be a number between 1024 and 65535 inclusive (or 0), and **Minimum NameServer client port** must be less than or equal to **Maximum NameServer client port**.  
  If the minimum and maximum values are both set to zero, then a random port number within the implicit fixed range will be dynamically assigned.  
  If both minimum and maximum values are set to the same port number, then this UDP port number will be used when communicating with the NameServer.  
  The default value for both minimum and maximum is 0. |
| **AppServer host name** | If you did not select the **Connect using NameServer** check box and the AIA will be directly connected to an AppServer or SonicMQ Adapter, you must specify the AppServer's or SonicMQ Adapter's host location. |
| **AppServer port** | If you did not select the **Connect using NameServer** check box and the AIA will be directly connected to an AppServer or SonicMQ Adapter, you must specify the AppServer's or SonicMQ Adapter's port. |

**AIA Logging Setting properties**

**Table 70: AIA Logging Setting properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Log filename | The AIA log filename. Enter a valid path name.  
  The installed default is `{WorkPath}\Aia\aia.log` |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Logging level</strong></td>
<td>A value that specifies the amount of information to be written to the AIA log. Select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>The default is <strong>Basic</strong>.</td>
</tr>
<tr>
<td><strong>Append to log file</strong></td>
<td>A check box that indicates whether or not a new server log file should be created when the AIA is started.</td>
</tr>
<tr>
<td></td>
<td>To create a new AIA log file each time the AIA is started, even if the AIA log file already exists, clear the <strong>Append to log file</strong> check box.</td>
</tr>
<tr>
<td></td>
<td>To append log entries to the existing AIA log file, select the <strong>Append to log file</strong> check box.</td>
</tr>
<tr>
<td></td>
<td>The <strong>Append to log file</strong> check box is selected by default.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Logging entry types</td>
<td>The AIA log entry type turns on logging for different AIA actions based on the logging level. The AIA log entry types provide the following information:</td>
</tr>
<tr>
<td></td>
<td>• <strong>AiaDefault</strong> — Combines AiaMgmt and AiaProp. This is the default entry type for the AIA component.</td>
</tr>
<tr>
<td></td>
<td>• <strong>AiaMgmt</strong> — Logs messages about regular AIA operations, such as AIA version information, instantiation, informational details, and information on the AIA connection pool according to the following logging level settings:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logs instantiation and informational messages</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logs the connection pool information</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logs the same information as <strong>Verbose</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>AiaProp</strong> — Logs the properties set for the specific AIA instance. These properties are set in the ubroker.properties file. Information is logged at <strong>Basic</strong> and higher.</td>
</tr>
<tr>
<td></td>
<td>• <strong>AiaRqst</strong> — Logs information on the GET and POST commands sent by the client and the response back to the client. Information is logged at <strong>Basic</strong> and higher.</td>
</tr>
<tr>
<td></td>
<td>• <strong>AiaUBroker</strong> — Logs information on the communication between the AIA and the Unified Broker (AppServer or SonicMQ Adapter) according to the following Logging level settings:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Provides most of the information</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logs a binary dump of messages exchanged between the client and the broker/server process</td>
</tr>
</tbody>
</table>
### Log file threshold size

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

### Maximum number of log files

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of AIA log files to keep. The specified number represents the maximum total number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the log file threshold size, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:filename.#####.extension

### AIA SSL properties

Table 71: AIA SSL properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL client connections</td>
<td>If the check box is selected, this property directs that the connection to the AppServer or SonicMQ Adapter use SSL tunneling.</td>
</tr>
<tr>
<td>Disable client verification of SSL host name</td>
<td>If the check box is selected, this property turns off host verification for an SSL connection. If the check box is cleared, the client compares the host name for the AppServer or SonicMQ Adapter specified by AppServer Location Host Name (or as returned by any Controlling NameServer) with the Common Name specified in the server digital certificate, and raises an error if they do not match. With this parameter specified, the client never raises the error.</td>
</tr>
<tr>
<td>Disable SSL session reuse</td>
<td>If the check box is selected, the connection does not reuse the SSL session ID when reconnecting to the same AppServer or SonicMQ Adapter.</td>
</tr>
</tbody>
</table>
AIA Advanced Features properties

Table 72: AIA Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal administrative command</td>
<td>Provides permission to see whether or not the AIA is running successfully. The check box is cleared by default.</td>
</tr>
<tr>
<td>Authorized IP list</td>
<td>If the Internal administrative command check box is selected, this box shows a comma-separated list of IP addresses that are allowed to access the AIA Adapter's internal administrative commands. To add or delete an IP address, click Edit and type or remove an IP address in the box. To allow any IP address to access the AIA Adapter's internal commands, leave the box blank. This is the default.</td>
</tr>
<tr>
<td>AppServer Keepalive</td>
<td>Allows the AppServer to recognize that a client bound to it is no longer connected.</td>
</tr>
<tr>
<td>Enable dynamic property updates</td>
<td>If the check box is selected, dynamic property updates are allowed.</td>
</tr>
</tbody>
</table>

Editing an AppServer Internet Adapter configuration

You can edit the configuration of an AppServer Internet Adapter. To edit an AppServer Internet Adapter configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer Internet Adapter broker instance whose configuration you want to edit.
3. Click the AppServer Internet Adapter broker instance. The AppServer Internet Adapter details page appears.
4. In the Command and control section of the page, click Configuration. The AppServer Internet Adapter Configuration page appears.
5. Click Edit. (For details about the properties, see AppServer Internet Adapter (AIA) properties on page 144.)
6. Make the configuration changes, and then click Save.

Creating an AppServer Internet Adapter

You can create an AppServer Internet Adapter instance. To create an AppServer Internet Adapter instance:
1. From the drop-down for Resources on the management console menu, click **New OpenEdge Resource** > **AppServer Internet Adapter**.

   The **AppServer Internet Adapter Configuration** page appears.

2. Type the name of the new instance in the field provided.

   **Note**: The AppServer Internet Adapter name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured AppServer Internet Adapter names.

3. Click **Save**. The **AppServer Internet Adapter Configuration** page appears, showing the configuration details and allowing you to configure the instance's properties.

### Viewing the AppServer Internet Adapter log file

You can view the log file for an AppServer Internet Adapter log file by using the log file viewer. The log file viewer allows you to examine the log file through an HTML interface.

To access the log file viewer:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer Internet Adapter broker instance whose log file viewer you want to access.
3. Click the Edit icon. The **AppServer Internet Adapter** details page appears.
4. In the **Command and control** section of the page, click **Log File Viewer**.

You can work with the Log File Viewer in the following ways:

- Use the **Show** field to control how many AIA log file entries display at one time. The number entered into the **Show** field cannot be less than 10.

- Use the **Overlap** field to control how many entries are repeated from screen to screen.

   **Note**: The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

- **Reload** after changing the values in either the **Show** field or the **Overlap** field. If you do not reload, the viewer continues to display the previous values.

- **Go To** to control which numbered entry in the log file the viewer begins its display with. For example, a value of 10 entered into the **Go To** field will begin the display at the tenth log file entry.

   **Note**: You must click **Go To** after entering a value in the **Go To** field or the viewer will not update its display.

The default display of entries is in ascending order; choose **Descending** to change the display. Note that the **Show** field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

- **First** to display the first \( x \) entries, where \( x \) is the value in the **Show** field.

- **Previous** to display the previous \( x \) entries, where \( x \) is the value in the **Show** field.

- **Next** to display the next \( x \) entries, where \( x \) is the value in the **Show** field.
• Click Last to display the last x entries, where x is the value in the Show field.

• To view additional log file entries without changing your current starting log file entry, leave the Go To field blank, change the value in the Show field, and click Reload.

• If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the Log file status field.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer's inactivity, the reloaded log file view might not match the previous log file view of that resource.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.

### Enabling or disabling an AppServer Internet Adapter broker

You can enable or disable an AppServer Internet Adapter broker.

To enable or disable an AppServer Internet Adapter broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer Internet Adapter broker instance you want to enable or disable.
3. Click the Edit icon. The AppServer Internet Adapter details page appears.
4. In the Command and control section of the page, click Control. The broker's Control page appears.
5. Click Edit, and either select the Enabled check box to enable the broker or clear the check box to disable the broker.
6. Click Save.

### Deleting an AppServer Internet Adapter broker

You can delete an AppServer Internet Adapter (AIA) instance using the management console. The instance is also deleted from the ubroker.properties file.

If you delete an AIA instance that is running, the instance and all associated properties are deleted from the ubroker.properties file. However, the AIA instance continues to run until you use the Java servlet engine to shut down the AIA.

To delete an AppServer Internet Adapter broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the AppServer Internet Adapter broker instance you want to delete.
3. Click the Edit icon. The AppServer Internet Adapter details page appears.
4. Click Delete, and then click OK to confirm the deletion.
Configuring OpenEdge SOAP Web Services

You can work with Web Services Adapter instances (for OpenEdge SOAP Web Services) in OpenEdge Management and OpenEdge Explorer.

For details, see the following topics:

• Configuring and managing SOAP Web services
• Managing Web Services Adapter (for OpenEdge SOAP Web Services) instances
• Working with SOAP Web services

Configuring and managing SOAP Web services

The Web Services Adapter (WSA) (for OpenEdge SOAP Web Services) is a Java servlet that exposes OpenEdge AppServer applications as SOAP Web services. The WSA is installed and runs in the context of a Java servlet engine (JSE) that runs in the context of a Web server.

To expose AppServer applications as SOAP Web services, the WSA serves a dual role:

• As a gateway between the SOAP request messages, which SOAP Web services and SOAP Web service clients exchange, and ABL applications on the AppServer, which execute SOAP Web service requests
• As an application server that hosts, manages, and provides communications and run-time support for multiple deployed SOAP Web service applications
SOAP-AVM gateway

For any given SOAP Web service, the WSA is a gateway between SOAP and the ABL Virtual Machine (AVM) running on the AppServer. As such, the WSA receives SOAP Web service method calls and parameters sent as SOAP messages from SOAP Web service clients and converts them into ABL function and procedure calls that the WSA invokes on a specified AppServer. In response, the WSA also receives the return and output parameter values sent from the called ABL functions and procedures on the AppServer and converts them into SOAP messages that it sends back to the calling SOAP Web service clients.

Web Service Application Server

The WSA is where you deploy a SOAP Web service that you want to make available to clients. Once deployed, it also allows you to manage the SOAP Web service individually and as part of a group of SOAP Web services. This includes such options as making SOAP Web service definitions available to potential clients as downloadable Web Services Description Language (WSDL) files and making one or more SOAP Web services available as applications for access by clients.

To initially create SOAP Web service definitions, you use ProxyGen to define an existing set of AppServer procedures and functions in terms of SOAP Web service objects. ProxyGen generates a Web Service Mapping (WSM) file. This WSM file stores the SOAP Web service object definitions in a form similar to a WSDL file, but without the final deployment information targeted to a specific WSA installation.

Thus, when you deploy a SOAP Web service to a WSA, you specify the information including the WSM file into a WSDL file that a specific WSA can use to provide the SOAP Web service definition to Web service clients. This same process also creates a Web Service Application Descriptor (WSAD) file that provides the SOAP Web service definition to a specific WSA in a form that allows it to host the deployed SOAP Web service and manage its run-time behavior.

Managing SOAP Web services with the WSA

The WSA is installed in a JSE as a single Web application that you can instantiate as one or more servlets (WSA instances). Each WSA instance provides a single point of management for one or more deployed SOAP Web services. Thus, to begin working with the WSA in OpenEdge Management or OpenEdge Explorer:

1. Create a new WSA instance if none exists or you need another, and configure it for operation. (The WSA comes installed with one local WSA instance, wsa1.)

2. For a WSA instance where you want to deploy a new SOAP Web service, optionally review and update the default property settings that govern the run-time behavior of new SOAP Web service deployments. You can also adjust these settings for each SOAP Web service individually after you deploy it.

3. If you want to import an existing SOAP Web service, first export the SOAP Web service from the context of the WSA instance where it is currently hosted.

4. Deploy the new SOAP Web service or import the existing SOAP Web service into the context of a WSA instance that you have configured.

5. As you client-test the SOAP Web service, review the WSA instance status and statistics, and update any WSA instance properties as needed. Such changes affect all SOAP Web services deployed to the same WSA instance.
Note: To apply changes to most WSA properties, you must restart the JSE where the WSA is installed. You can make temporary changes to selected WSA properties at run time without restarting the JSE. However, you must update the WSA instance configuration to make these changes permanent.

6. As you client-test the SOAP Web service, review the SOAP Web service status and statistics, and update any SOAP Web service properties as needed. Such changes affect only the selected SOAP Web service. For most of these changes, you must disable the SOAP Web service before you make them, then enable the SOAP Web service afterward to evaluate the effect.

SOAP Web service session models and the WSA

SOAP Web service applications in OpenEdge are designed to run in one of two AppServer session models:

• Session-Managed
• Session-Free

The session model of a SOAP Web service application significantly affects how the WSA manages the SOAP Web service at run time. In any case, the session model specified for a SOAP Web service (when defined using ProxyGen) must match the session model of the AppServer to which it is bound. Otherwise, the WSA returns an error when a client attempts to access the SOAP Web service.

Session-managed SOAP Web services

The WSA handles session-managed SOAP Web services in exactly the same way that other OpenEdge clients access an AppServer running in state-aware, state-reset, or stateless operating mode. That is, session-managed SOAP Web services are single-threaded. Thus, the WSA maintains a single network session to connect the SOAP Web service and all its objects to a single AppServer on behalf of a single SOAP Web service client. All requests from the client to the same SOAP Web service objects use the same AppServer connection. This session model provides full support on the AppServer side for context management, and makes available the capabilities of the AppServer to support transaction-oriented interactions across multiple client requests.

Session-free SOAP Web services

The WSA handles session-free SOAP Web services in exactly the same way that other OpenEdge clients access a traditional AppServer running in state-free operating mode. For session-free SOAP Web services, the WSA maintains a pool of AppServer TCP/IP connections (connection pool) for use by all clients of a single SOAP Web service. This allows multiple clients to access the same SOAP Web service concurrently; that is, session-free SOAP Web services are multi-threaded. The WSA can do this because a session-free AppServer maintains no client connections, but accepts and responds to client requests one at a time regardless of origin and with no context maintained from one request to another. This session model thus supports the industry standard for SOAP Web service interactions that focuses on small, atomic, and complete transactions in a single client request.

The connection pool maintains a list of available connections to session-free AppServers that support the same application service on behalf of all clients using the corresponding SOAP Web service. The WSA can then service as many client requests at one time as AppServer connections in the connection pool. It also queues any additional client requests for the first available AppServer that completes a client request.

Note: Session-free SOAP Web services can support SOAP Web service objects that require dedicated AppServer connections. However, typical session-free SOAP Web services avoid the use of these types of SOAP Web service objects because they undercut the benefits of connection-less interactions.
Managing Web Services Adapter (for OpenEdge SOAP Web Services) instances

You can manage Web Services Adapters (for OpenEdge SOAP Web Services) by:

- Creating a new Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 156
- Configuring a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 157
- Editing a Web Services Adapter (for OpenEdge SOAP Web Services) instance configuration on page 166
- Starting Web Services Adapter (for OpenEdge SOAP Web Services) instances on page 167
- Setting SOAP Web service defaults on page 167
- Deploying a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 168
- Importing a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 170
- Listing the deployed SOAP Web services for a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 171
- Viewing the status of a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 171
- Viewing SOAP Web service statistics on page 175
- Changing Web Services Adapter (for OpenEdge SOAP Web Services) instance run-time properties on page 172
- Stopping Web Services Adapter (for OpenEdge SOAP Web Services) instances on page 173
- Deleting a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 173

Each of these tasks is described in the following sections.

Creating a new Web Services Adapter (for OpenEdge SOAP Web Services) instance

OpenEdge Management and OpenEdge Explorer provide a sample Web Services Adapter (WSA) instance (wsa1). You can edit the wsa1 sample configuration, and you can create new WSA instances.

**Note:** To complete the creation of a WSA instance after creating it in the management console and before you can deploy SOAP Web services to it, you must update your JSE configuration and edit the WSA’s `web.xml` file. Add a servlet and servlet-mapping element for the instance, then restart your JSE. This creates the root sub-path for the new instance, initializes and runs it. For more information, see *OpenEdge Development: Web Services* and your JSE documentation.

To create a new WSA instance:
1. From the drop-down for Resources on the management console menu, click New OpenEdge Resource > WebServices Adapter. The WebServices Adapter Configuration page appears.

2. Type the name of the new Web Services Adapter (for OpenEdge SOAP Web Services) in the New Web Services name field.

Note: The Web Services Adapter (for OpenEdge SOAP Web Services) name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured NameServer names.

3. Type the URL in the Url field.

This is a URL that directly addresses the WSA instance on the Internet and serves as the root for other URLs required to access SOAP Web services and other functions of the WSA.

Form the root URL according to the following format:

http://host[:port] [/jse-context]/webapp-context/servlet-instance

Where:

- host— The name of the machine where the WSA is installed (Default: localhost).
- port — The HTTP port number on the WSA machine (Default: 80).
- jse-context — The JSE context path name is present only when you have a Web server between your Java Servlet Engine (JSE) and the client application. It specifies the path name that you define in your Web server configuration to send client requests to your JSE.
- webapp-context — The Web application context path name always exists and specifies the path name that you define in your JSE configuration to identify the WSA as a servlet application.
- servlet-instance — The servlet-mapping name for the WSA instance (WSA Web application) specified in OpenEdge-install-dir\servlets\wsa\WEB-INF\web.xml. This is also the name by which the WSA instance appears in the Web Services Adapter folder in the management console.

4. Select the WSA installation location for this instance from these choices:

- local — The WSA is installed on the machine where the selected AdminServer is running.
- remote — The WSA is installed on a network machine that is remote from the selected AdminServer.

5. Click Save. The new WSA instance appears under the Web Services Adapter folder.

You can now configure the properties of the new instance. Note that when you create a new instance, it has a copy of the default configuration. Initially, the instance is enabled only for administration, allowing you to modify its properties and deploy SOAP Web services to it without allowing premature user access to an incomplete configuration.

Configuring a Web Services Adapter (for OpenEdge SOAP Web Services) instance

Before deploying SOAP Web services, you must configure a Web Services Adapter (for OpenEdge SOAP Web Services) instance to host the SOAP Web services that you want to deploy or import. To configure a WSA instance, you set properties for it similar to any other Unified Broker product.
After the WSA is configured, you can import or deploy a SOAP Web service and configure additional properties that affect execution of the SOAP Web service. The SOAP Web service properties that are available for configuration depend on the session model of the SOAP Web service (session managed or session free). To verify the session model of a SOAP Web service, check its status. For details on checking SOAP Web service status, see Viewing the status of a SOAP Web service on page 174. For details about configuring SOAP Web service properties, see Updating a deployed SOAP Web service on page 175. For details about the properties available on a SOAP Web service, see OpenEdge Application Server: Administration.

You must disable a SOAP Web service before you can configure most properties on it. You can verify if the SOAP Web service is disabled by checking its status. The only properties that you can configure for an enabled SOAP Web service include:

- serviceFaultLevel
- serviceLoggingLevel

If you set other properties while the SOAP Web service is enabled, the property value changes take effect only after you disable and then enable the SOAP Web service again. For details about enabling or disabling a SOAP Web service, see Enabling or disabling a SOAP Web service for client access on page 174.

To configure a Web Services Adapter (for OpenEdge SOAP Web Services) instance, you can view or edit properties in the following categories:

- Location
- Proxy Server Setting
- WSDL
- Logging Setting
- Security
- Advanced Features

You can also create a new Web Services Adapter (for OpenEdge SOAP Web Services) instance, view its status, or delete an instance you no longer need.

Each of these tasks is described in the following sections.

### Setting Web Services Adapter (for OpenEdge SOAP Web Services) instance properties

You can set location, proxy server setting, WSDL, logging setting, security, and advanced features properties for a Web Services Adapter (for OpenEdge SOAP Web Services) instance.

To set WebServices Adapter instance properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose properties you want to set.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. If your Web server requires that you log in, click Login in the Command and control section of the page. Type your user name and your password, and then click Submit.
5. In the Command and control section of the page, click Configuration. The WebServices Adapter Configuration page appears. From this page, you can:
• View a read-only display of the Web Services Adapter (for OpenEdge SOAP Web Services) properties as described in Web Services Adapter (for OpenEdge SOAP Web Services) properties on page 159. All of these tables follow this procedure.

• Click Edit to modify the properties. For details, see Editing a Web Services Adapter (for OpenEdge SOAP Web Services) instance configuration on page 166.

Web Services Adapter (for OpenEdge SOAP Web Services) properties

The following sections describe the Web Services Adapter (for OpenEdge SOAP Web Services) properties.

Web Services Adapter (for OpenEdge SOAP Web Services) Location properties

You set the properties, shown in the following table, in this category when you first create the WSA instance. For more information see Creating a new Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 156.

You cannot change the location properties. Instead, you must delete and then recreate the WSA instance with the new values. For more information, see Deleting a Web Services Adapter (for OpenEdge SOAP Web Services) instance on page 173.

Table 73: Web Services Adapter (for OpenEdge SOAP Web Services) Location properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>A read-only property that indicates where the WSA is installed:</td>
</tr>
<tr>
<td></td>
<td>• local — The WSA is installed on the machine where the selected AdminServer is running. You can configure all the properties of a local WSA.</td>
</tr>
<tr>
<td></td>
<td>• remote — The WSA is installed on a network machine that is remote from the selected AdminServer.</td>
</tr>
<tr>
<td>URL</td>
<td>A read-only property set to the base URL for the WSA.</td>
</tr>
<tr>
<td>Web Service Protocol Value</td>
<td>The protocol for all SOAP Web service URLs inserted into WSDL documents that are generated for this WSA instance during SOAP Web service deployment. The host, port, and other path information specified for the WSA URL when the instance was originally created remains unchanged. You can select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• Duplicate WSA Url value — Uses the original protocol specified for the WSA URL</td>
</tr>
<tr>
<td></td>
<td>• Use HTTP protocol — Replaces the original WSA URL protocol with HTTP</td>
</tr>
<tr>
<td></td>
<td>• Use HTTPS protocol — Replaces the original WSA URL protocol with HTTPS</td>
</tr>
</tbody>
</table>

Installed Default: Duplicate WSA Url value
### Web Services Adapter (for OpenEdge SOAP Web Services) Proxy Server Setting properties

The properties in this category, as shown in the following table, apply to both local and remote WSA instances. They specify the information required to access any proxy server for the network on which the WSA resides.

**Table 74: Web Services Adapter (for OpenEdge SOAP Web Services) Proxy Server Setting properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy host</td>
<td>The TCP host name for the proxy server. Installed Default: None</td>
</tr>
<tr>
<td>Proxy port</td>
<td>The TCP port number of the proxy service. Installed Default: 0</td>
</tr>
<tr>
<td>Proxy username</td>
<td>A valid username required to access the proxy server. Installed Default: None</td>
</tr>
<tr>
<td>Proxy password</td>
<td>The password for the specified username. Installed Default: None</td>
</tr>
<tr>
<td>Proxy password Confirm</td>
<td>The password retyped for confirmation</td>
</tr>
</tbody>
</table>

### Web Services Adapter (for OpenEdge SOAP Web Services) WSDL properties

The properties in this category, as shown in the following table, control access to WSDL files and information.

**Table 75: Web Services Adapter (for OpenEdge SOAP Web Services) WSDL properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable WSDL retrieval</td>
<td>If selected, this check box setting allows network users to access WSDL files deployed to this WSA instance. Otherwise, users cannot access any WSDL files deployed to this WSA instance. Installed Default: Activated</td>
</tr>
<tr>
<td>WSDL inaccessible page</td>
<td>The relative path and filename of the Web page that you want the WSA instance to return when users attempt to access WSDL files deployed to this instance. The Enable WSDL Retrieval check box is cleared to prevent that access. The path name is relative to your Web application (WSA) servlet root directory. Installed Default: None</td>
</tr>
</tbody>
</table>
If selected, this check box setting allows network users to access a Web page that lists the WSDL files deployed to this WSA instance. Otherwise, users cannot access the list of WSDL files deployed to this WSA instance.

Installed Default: Activated

The path and filename of the Web page that you want the WSA instance to return when users request a listing of the WSDL files deployed to this instance. The WSA returns this page only when the Enable WSDL listing retrieval check box is selected. The path name is relative to your Web application (WSA) servlet root directory.

Installed Default: WSDLListing.html

### Web Services Adapter (for OpenEdge SOAP Web Services) Logging Setting properties

The properties in this category, as shown in the following table, control the location and content of the WSA log file.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>The directory where the WSA writes temporary files when required. Enter a valid path name.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: ${WorkPath}</td>
</tr>
<tr>
<td>Log filename</td>
<td>The WSA log filename. Enter a valid path name.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: ${WorkPath}\wsa1.wsa.log</td>
</tr>
</tbody>
</table>
### Logging level

A value that specifies the amount of information that is written to the WSA log. Select from the following values in the drop-down list:

- **None** — Log no entries. This is equivalent to turning logging off.
- **Error Only** — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.
- **Basic** — Logging Entry Types determine the logged information. Each entry type generates at least some output. This is the default.
- **Verbose** — Logging entry types determine the logged information, but it is typically more information than **Basic**.
- **Extended** — Logging entry types determine the logged information, but it is typically more information than **Verbose**.

**Note:** A value of **Other** in this field indicates an option manually specified in the `ubroker.properties` file that provides additional diagnostic logging features. You can reset this by choosing from the values in the drop-down list.

**Installed Default:** Basic

### Append to log file

A check box that indicates whether or not a new server log file should be created when the WSA is started.

To create a new WSA log file each time the WSA is started, even if the WSA log file already exists, clear the **Append to log file** check box.

To append log entries to the existing WSA log file, select the **Append to log file** check box.

**Installed Default:** Selected

### Log message threshold size

An integer value that limits the amount of SOAP message detail written to the log file for clients specified using the **Debug Clients** property in the **Advanced Features** category. A value of -1 indicates that all SOAP message detail is written to the file. A positive value indicates the number of characters written to the file for each logged SOAP message. Practical values range from 100 to 10000, but any positive value is valid.

**Installed Default:** -1
### Logging entry types

A comma-separated list of valid entry types for WSA logging. The only standard entry is WSADefault. Set additional entry types only if instructed to do so by Progress Software Corporation Technical Support.

You can specify a separate logging level for each log entry type, and you can use a wildcard (*) to specify multiple entry types. For example:

```
UBroker.Basic:3,UBroker.ThreadPool:4, UBroker.Client*
```

### Log file threshold size

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

### Maximum number of log files

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of WSA log files to keep. The specified number represents the maximum number of log files to keep on disk at any time, including the current log file being written to.

The default value is 3.

When the file becomes equal to or greater than the log file threshold size, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:

```
filename.######.extension
```

### Web Services Adapter (for OpenEdge SOAP Web Services) Security properties

The properties in this category, as shown in the following table, control options for authorizing and authenticating access to WSA functions and SOAP Web services.

**Table 77: Web Services Adapter (for OpenEdge SOAP Web Services) Security properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable WSA administration</td>
<td>If the check box is selected, indicates that administration functions are available for the selected WSA instance. Installed Default: Selected</td>
</tr>
</tbody>
</table>
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSA administration authentication</td>
<td>A drop-down list that indicates the criteria used to authenticate users for access to WSA administrative functions. The choices include:</td>
</tr>
<tr>
<td></td>
<td>• <strong>No user authentication</strong> — Any user can perform WSA administration functions.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use Admin Server user authentication</strong> — The AdminServer forwards the same user credentials supplied at connection time to the Web server (or JSE) where the WSA is running in order to perform administration functions.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Prompt user for Web Server username and password</strong> — Only users authorized to manage the Web server (or JSE) where the WSA is running can perform WSA administration functions.</td>
</tr>
<tr>
<td>Require WSA administration authorization</td>
<td>If the check box is selected, indicates that each user should be validated against selected Administration Role definitions when the user attempts to perform an administrative function on the WSA instance or on any of its deployed SOAP Web services.</td>
</tr>
<tr>
<td></td>
<td>For details, see Viewing or modifying the Admin or Operator role on page 164.</td>
</tr>
<tr>
<td>To enable administration roles, select from the following list</td>
<td>Predefined Roles: PSCAdmin, PSCOpre</td>
</tr>
<tr>
<td></td>
<td>For details, see Viewing or modifying the Admin or Operator role on page 164.</td>
</tr>
<tr>
<td>Enable Web Services</td>
<td>If the check box is selected, indicates that users can access the SOAP Web services deployed to the selected WSA instance.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: Cleared</td>
</tr>
<tr>
<td>Require Web Services authorization</td>
<td>If the check box is selected, indicates that users must be authorized to access SOAP Web services.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: Cleared</td>
</tr>
<tr>
<td>Require WSDL retrieval authorization</td>
<td>If the check box is selected, indicates that users must be authorized to retrieve WSDL files and information.</td>
</tr>
<tr>
<td></td>
<td>Installed Default: Cleared</td>
</tr>
</tbody>
</table>

### Viewing or modifying the Admin or Operator role

You can define what level of access is allowed for the Web Services Adapter (for OpenEdge SOAP Web Services) instance: Administrator or Operator. Under Admin Role Links, click either AdminRole.PSCAdmin or AdminRole.PSCOpe to see the current read, write, delete, and execute settings for the following:

- **Application defaults** — Governs which operations the role may perform on the default SOAP Web service properties that are used to initialize a newly deployed SOAP Web service

- **Application enablement** — Governs which operations the role may perform on the enable state of SOAP Web service applications
• **Application properties** — Governs which operations the role may perform on a SOAP Web service's properties

• **Application statistics** — Governs which operations the role may perform on a SOAP Web service's statistical counters

• **Servlet properties** — Governs which operations the role may perform on the WSA's run-time changeable properties

• **Servlet services** — Governs which operations the role may perform on the registry of deployed SOAP Web service applications

• **Servlet statistics** — Governs which operations the role may perform on the WSA's statistical counters

To edit the settings, click **Edit**. To create a new AdminRole name, click **Create**.

### Web Services Adapter (for OpenEdge SOAP Web Services) Advanced Features properties

The properties in this category, shown in the following table, support less commonly-used or special-purpose features, some of which might relate to the properties of other categories.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSA administration SOAP action</td>
<td>A string placed in the SOAPAction HTTP header when the AdminServer forwards administration operations to the WSA. The SOAPAction HTTP header is required for all HTTP messages that carry SOAP messages and is used by intervening security servers (such as firewalls) to determine if each HTTP message is allowed to pass through to its destination. The installed default is a Universal Resource Number URN identifying the WSA administration SOAP Web service, but you can also specify any string required by intervening security servers on the network.</td>
</tr>
<tr>
<td></td>
<td><strong>Installed Default</strong>: <code>urn:services-progress-com:wsa-admin:01</code></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HTTP error page</td>
<td>The path and filename of a static Web page that you want returned to a SOAP Web service user if the request results in an HTTP error, such as a service unavailable or an authorization error. The path name is relative to your web application (WSA) servlet root directory. Installed Default: httperror.html</td>
</tr>
<tr>
<td>Debug clients</td>
<td>A list of dot-formatted IP addresses for SOAP Web service client machines whose HTTP and SOAP messages you want dumped to the WSA log file. The WSA dumps each SOAP message up to the limit specified by the Log message threshold property setting in the Logging Setting category. To add a client machine to the list, enter its dot-formatted IP address in the Client Host field, then choose the Add button to add it to the list. You can also enter multiple, comma-separated addresses in the field at one time to dump messages for multiple clients, or enter a single asterisk (*) to dump the messages for all clients. To remove whole lines of addresses from the list, select one or more lines in the list and click Delete. To edit a single line of addresses, select the line in the list, and after modifying the line click Add to add it to the list. The original line remains. Select the original line and click Delete to remove it from the list and leave the edited line in its place. Note: The list automatically adds new lines in increasing order of collation (numerical, for valid IP addresses). If you add an invalid IP address to the list, the WSA never matches it with a message to dump in the log file, and thus effectively ignores it. Installed Default: None</td>
</tr>
</tbody>
</table>

**Editing a Web Services Adapter (for OpenEdge SOAP Web Services) instance configuration**

You can edit the configuration properties of a Web Services Adapter (WSA) (for OpenEdge SOAP Web Services) instance from the management console.

To edit Web Services Adapter (for OpenEdge SOAP Web Services) properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose properties you want to edit.
4. In the Command and control section of the page, click Configuration. The WebServices Adapter Configuration page appears.
5. Click **Edit**, and make the changes you want to the properties. (For details about the properties, see **Web Services Adapter (for OpenEdge SOAP Web Services) properties** on page 159.)

6. Click **Save**. Any changes you make are also reflected in the **ubroker.properties** file.

You can also make run-time changes in the management console to the values of the following WSA properties:

- **debugClients** — Controls whether to dump HTTP and SOAP protocol message information to the WSA log file, and specifies the clients (if any) whose SOAP messages are to be dumped
- **enableWsdl** — Controls the WSA's ability to respond to inquiries about WSDL documents for any of the SOAP Web service applications it hosts
- **enableWsdlListings** — Controls the WSA's ability to respond to a request for a list of WSDL documents it has available
- **loggingLevel** — Specifies the amount of information to be written to the broker log
- **logMsgThreshold** — Controls the amount of SOAP message information that can be written to the WSA log file
- **webAppEnabled** — Controls the WSA's ability to accept and process SOAP requests to any of its hosted SOAP Web service applications

These changes last only for the duration of the current Java servlet engine (JSE) session. To make these changes permanent across sessions, you must make the changes in the property settings.

### Editing the WSA instance configuration

To edit the WSA instance configuration:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose configuration you want to edit.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **WebServices Adapter Configuration** page appears.
5. Click **Edit**.
6. Click **Save**.

### Starting Web Services Adapter (for OpenEdge SOAP Web Services) instances

The Web Services Adapter (WSA) (for OpenEdge SOAP Web Services) and its instances start up automatically when you start your Web server or Java servlet engine (JSE). For more information, see the documentation for your Web server or JSE.

### Setting SOAP Web service defaults

You can view and set the default values used to initialize properties for SOAP Web services that you deploy. Any changes to these defaults affect only SOAP Web services that you deploy after confirming the change. To set the values of properties for deployed SOAP Web services, set the property values individually for each SOAP Web service.
Note: Unlike WSA instance properties, the defaults for SOAP Web service properties are not stored in the `ubroker.properties` file. They are stored in the `default.props` XML file located in the Web application servlet directory for each WSA instance.

To view and set SOAP Web service defaults:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose default values you want to view.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. In the Command and control section of the page, click Defaults. The WebServices Adapter Defaults page appears.
5. Click Edit.
6. Select the property whose default value you want to change. For details on the available properties, see OpenEdge Application Server: Administration.
7. Type the new value in the available field. For logical values, typing any value other than true results in a setting of false.
8. Click Submit to change the default value.

To return to the default values, you can click Reset from the Defaults page.

Deploying a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance

You can deploy a SOAP Web service to the context of any available WSA instance using the Web Service Mapping (WSM) file generated using ProxyGen. For more information on generating a WSM file, see OpenEdge Development: Web Services.

A deployed SOAP Web service receives its initial property values from the `default.props` file for the WSA instance. You can either change these default values before you deploy the SOAP Web service, or you can change the deployed SOAP Web service property values after you deploy the SOAP Web service. A SOAP Web service always deploys with a disabled status to prevent premature or unintended client access.

To deploy a SOAP Web service:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where you want to deploy the SOAP Web service.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. In the Command and control section of the page, click Deploy.
5. Type the path and name of the WSM file for the SOAP Web service that you want to deploy.
6. Click Submit. Verify that the deployment information shown (and which is based on information in the WSM file) is correct. You can modify information in the following fields:
   - WSM Filename — The path and name of the WSM file.
   - Name — The name used to identify the SOAP Web service in the management console and to name the SOAP Web service files that are deployed to the WSA instance.
• **Web Service Namespace** — Any value you choose to uniquely qualify the names for operations and parameters used to define the SOAP Web service.

• **SOAP Action** — A string that the client application is required to place in the SOAPAction HTTP header when accessing a SOAP Web service hosted by the WSA instance. The SOAPAction HTTP header is required for all HTTP messages that carry SOAP messages and is used by intervening security servers (such as firewalls) to determine if each HTTP message is allowed to pass through to its destination. The default is a blank string, but it can be any string required by the intervening security servers on the network.

• **Append to SOAP Action** — Indicates whether to append the specified SOAP Action value to any other SOAP Action values generated for the SOAP Web service messages by Web service clients.

• **WSDL Style/Use** — Specifies the SOAP format to use when sending or receiving SOAP messages for this SOAP Web service. The value that you choose is dependent on what your anticipated SOAP Web service clients can support and can be any of the following: **RPC/Encoded**, **RPC/Literal**, and **Document/Literal**.

7. Click **Submit** to deploy the SOAP Web service with these settings. A confirmation message appears, and the WSM file name is identified in the **Deployed Web Services** section of the WSA instance's **Details** page.

You might need to document additional information about these settings, especially **WSDL Style/Use**, for use by SOAP Web service client developers. For more information, see the chapter on client requirements in *OpenEdge Development: Web Services*.

Deployment generates the following files in the WSA instance directory:

• **WebServiceFriendlyName.props** — An XML file containing the current SOAP Web service property settings (initially set from **default.props**)

• **WebServiceFriendlyName.wsad** — The Web Service Adapter Descriptor (WSAD) XML file that defines the SOAP Web service to the WSA instance

• **WebServiceFriendlyName.wsdl** — The Web Services Description Language (WSDL) XML file that defines the SOAP Web service to potential SOAP Web service clients

For more information about these files, see *OpenEdge Development: Web Services*.

**Caution:** Once you deploy a SOAP Web service during development, you can change its definition and deployment information using a SOAP Web service update. However, once you deploy and enable a SOAP Web service for client access under production conditions, avoid making any changes to this information, as client implementations depend on its stability.

After production deployment, to make the same SOAP Web service available using different information (for example, to add a new operation or use a different **WSDL Style/Use**), deploy a new SOAP Web service with the new information, using a different SOAP Web service name in the same WSA context. You could also deploy the SOAP Web service to a different WSA context.

However, you can change the run-time properties of a deployed SOAP Web service at any time.

**Exporting a SOAP Web service**

You can export an existing SOAP Web service definition to save its deployment information, properties, and other settings. This creates a Web Service Definition (WSD) XML file with a name in the form **filename.wsd**, with a filename that you specify. You can use this file to copy (import) the SOAP Web service to another WSA instance or to save it as a backup for the SOAP Web service definition.

To export a SOAP Web service:
1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service you want to export is deployed.

3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.

4. In the **Deployed Web Services** section of the page, click **Export**.

5. Type the path and file name, ending in `.wsd`, of the file for the SOAP Web service you want to export.

6. Click **Submit**. A confirmation message appears.

### Importing a SOAP Web service to a Web Services Adapter (for OpenEdge SOAP Web Services) instance

You can import a SOAP Web service to any available WSA instance using the Web Service Description (WSD) file saved during a previous export of the SOAP Web service. The import function generates the appropriate SOAP Web service properties (`WebServiceFriendlyName.props`), WSAD, and WSDL files to duplicate the original SOAP Web service in the context of the selected WSA instance. However, a SOAP Web service always imports with a disabled status to prevent premature or unintended client access.

To import a SOAP Web service:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the WebServices Adapter broker instance where you want to import the SOAP Web service.

3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.

4. In the **Command and control** section of the page, click **Import**.

5. Type the path name of the `.wsd` file for the SOAP Web service you want to import.

6. Choose **Submit**. Verify that the information shown (and which is based on information in the WSD file) is correct. You can modify information in the following fields:
   
   - **Name** — The name used to identify the SOAP Web service and to name the SOAP Web service files that are deployed to the WSA instance.
   - **Web Service Namespace** — Any value you choose to uniquely qualify the names for operations and parameters used to define the SOAP Web service.
   - **SOAP Action** — A string that the client application is required to place in the SOAPAction HTTP header when accessing a SOAP Web service hosted by the WSA instance. The SOAP Action HTTP header is required for all HTTP messages that carry SOAP messages and is used by intervening security servers (such as firewalls) to determine if each HTTP message is allowed to pass through to its destination. The default is a blank string, but can be any string required by the intervening security servers on the network.
   - **Append to SOAP Action** — Indicates whether to append the specified SOAP Action value to any other SOAP Action values generated for the SOAP Web service messages by SOAP Web service clients.
   - **WSDL Style/Use** — Specifies the SOAP format to use when sending or receiving SOAP messages for this SOAP Web service. Supported formats are **RPC/Encoded**, **RPC/Literal**, and **Document/Literal**.

7. Click **Submit**. A confirmation message appears.
Listing the deployed SOAP Web services for a Web Services Adapter (for OpenEdge SOAP Web Services) instance

You can list all SOAP Web services in the management console that are deployed to a selected WSA instance.

To list all SOAP Web services deployed to a WSA instance:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose deployed SOAP Web services you want to see.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. Under the Command and control section, click List. The list of all SOAP Web services deployed to the WSA instance appears.

Viewing the status of a Web Services Adapter (for OpenEdge SOAP Web Services) instance

You can view the execution status of a selected WSA instance to see if the instance is running. If the instance is running, the status indicates if administration, SOAP Web service application, or WSDL document access is enabled.

To view the status of a WSA instance:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose status you want to view.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. Under Operations views, click Status. The following status information appears:
   - Whether the WSA instance is running
   - Whether access to administrative functions, SOAP Web service applications (by clients), and WSDL document retrieval is enabled

Viewing Web Services Adapter (for OpenEdge SOAP Web Services) instance statistics

You can view (and reset) run-time statistics for a selected WSA instance. Each WSA instance creates and begins accumulating these statistics (Start Time) when it is first started by the Java Servlet Engine (Creation Time). You can also reset the statistics at any point during WSA execution, so that the statistics begin accumulating again from a new Start Time.

To view the statistics of a WSA instance:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose statistics you want to view.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.

4. Under **Operations views**, click **Statistics**. The following information appears:

- **Number of SOAP Requests** — Requests for SOAP Web service applications
- **Number of Active Requests** — Requests queued and being acted upon
- **Number of HTTP Requests** — Total requests received from the HTTP listener, including administrative, WSDL, and SOAP Web service requests
- **Number of WSDL Requests** — Requests for WSDL documents
- **Number of SOAP Faults** — Error returns from SOAP Web service applications
- **Number of Services Disabled** — SOAP Web services deployed to this WSA that are disabled from client access
- **Number of Errors** — Total errors returned by the WSA, with error counts broken out at the bottom of the list for each of several error categories when total errors are greater than zero (0)

5. To reset the statistics and start accumulating all statistics from zero, click **Reset** and confirm that you want to restart the statistics.

**Viewing statistics for deployed SOAP Web services**

You can view statistical details for deployed SOAP Web services.

To view statistics for deployed SOAP Web services:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose runtime statistics you want to view.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.
4. Under **Deployed Web Services**, click **Statistics**. Read-only statistics information appears.

You can reset the statistical display by clicking **Reset** on the adapter instance’s **Statistics** page.

**Changing Web Services Adapter (for OpenEdge SOAP Web Services) instance run-time properties**

You can temporarily change some Web Services Adapter (WSA) (for OpenEdge SOAP Web Services) instance properties at run time without restarting your Java servlet engine (JSE). This is most useful for testing and debugging. The next time you restart your JSE, these settings revert to the current configuration settings for these properties in the `ubroker.properties` file.

To change WSA instance runtime properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance whose runtime properties you want to change.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.
5. Click **Edit** to modify any of the following properties:

- **adminEnabled** — Specifies whether administrator privileges are in effect, either **true** or **false**.
Managing Web Services Adapter (for OpenEdge SOAP Web Services) instances

- **debugClients** — Controls whether to dump HTTP and SOAP protocol message information to the WSA log file, and specifies the clients (if any) whose SOAP messages are to be dumped.
- **enableWsdl** — Controls the WSA's ability to respond to inquiries about WSDL documents for any of the SOAP Web service applications it hosts.
- **enableWsdlListings** — Controls the WSA's ability to respond to a request for a list of WSDL documents it has available.
- **loggingLevel** — Specifies the amount of information to be written to the broker log.
- **logMsgThreshold** — Controls the amount of SOAP message information that can be written to the WSA log file.
- **webAppEnabled** — Controls the WSA's ability to accept and process SOAP requests to any of its hosted SOAP Web service applications.

6. Click **Submit**.

**Stopping Web Services Adapter (for OpenEdge SOAP Web Services) instances**

The Web Services Adapter (for OpenEdge SOAP Web Services) (WSA) and its instances stop execution automatically when you shut down your Web server or Java servlet engine (JSE). For more information, see the documentation for your Web server or JSE.

**Deleting a Web Services Adapter (for OpenEdge SOAP Web Services) instance**

You can delete any Web Services Adapter (WSA) (for OpenEdge SOAP Web Services) instance listed in the management console as long as it has no SOAP Web services deployed to it. Undeploy all of its deployed SOAP Web services before you attempt to delete the instance.

For more information, see *OpenEdge Development: Web Services* and your JSE documentation.

**Note:** To completely remove a WSA instance after deleting it in the management console, you must also remove its corresponding servlet and servlet-mapping elements from the WSA's `web.xml` file and restart your Web server or Java Servlet Engine (JSE).

To delete a Web Services Adapter (for OpenEdge SOAP Web Services) instance:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter, or search for, the WebServices Adapter broker instance you want to delete.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.
4. Click **Delete**, and then click **OK** to confirm the deletion.

The **OpenEdge Management Resources** page appears. A set of next-step options that are related to resources are available for your use.
Working with SOAP Web services

You can perform the following SOAP Web services administration tasks from the management console:

- Enabling or disabling a SOAP Web service for client access on page 174
- Viewing the status of a SOAP Web service on page 174
- Viewing SOAP Web service statistics on page 175
- Updating a deployed SOAP Web service on page 175
- Exporting a SOAP Web service on page 169
- Undeploying a SOAP Web service on page 177

Enabling or disabling a SOAP Web service for client access

You can enable any deployed SOAP Web service, making it accessible or inaccessible to user requests on the network.

To enable or disable a SOAP Web service for client access:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service you want to enable or disable is deployed.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. In the Deployed Web Services section, click Status Enablement. The list of all SOAP Web services deployed to the WSA instance appears.
5. Click Enable or Disable.
6. Click OK. A confirmation message appears.

Viewing the status of a SOAP Web service

You can view status information on a deployed SOAP Web service.

To view the status of a SOAP Web service:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service whose status you want to view is deployed.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. Under Deployed Web Services, click Status Enablement. This read-only status presents the following information in a list of label-value pairs separated by a colon:
   - Target NameSpace — The target name space identifier, typically a Universal Resource Name (URN)
   - Status — Indicates if the SOAP Web service is enabled or disabled for client access
• **AppServerInfo** — The AppServer connection information, expressed as a URL
• **Session Model** — Indicates if the SOAP Web service requires an AppServer configured with session-managed or session-free AppServer operating modes
• **Style/Use** — SOAP message style (RPC or Document)

In addition, the status includes the a list of current application properties.

### Viewing SOAP Web service statistics

You can view (and reset) run-time statistics for a selected SOAP Web service. Each SOAP Web service creates and begins accumulating these statistics (Start Time) from the time that its parent WSA instance starts up (Creation Time). You can reset the SOAP Web service statistics at any point while the WSA instance is running. A reset clears and begins accumulating the statistics again with a new Start Time.

To view the run-time statistics for a SOAP Web service:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service whose runtime statistics you want to view is deployed.
3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.
4. Under **Deployed Web Services**, click **Statistics**. A read-only statistics presentation with the following information appears:

   • **Number of Requests** — Total requests handled by the SOAP Web service
   • **Number of Active Requests** — Requests currently being serviced by the SOAP Web service
   • **Number of SOAP Faults** — Errors returned by the SOAP Web service application
   • **Number of Objects Not Found** — Calls to unidentified SOAP Web service methods
   • **Number of Errors** — Total errors returned by the SOAP Web service, with error counts broken out at the bottom of the list for each of several error categories when the total errors in each category are greater than 0
   • **Number of AppServer Connections** — Total AppServer connections in the connection pool for this SOAP Web service
   • **Number of AppObjects** — Total AppObjects registered for all clients of this SOAP Web service (only one AppObject is shared by all clients of a session-free SOAP Web service; no AppObjects are ever shared for a session-managed SOAP Web service)
   • **Number of SubAppObjects** — Total SubAppObjects registered for all clients of this SOAP Web service (SubAppObjects never shared)
   • **Number of ProcObjects** — Total ProcObjects registered for all clients of this SOAP Web service (ProcObjects never shared)

5. To reset the statistics and start accumulating from zero, click **Reset** and confirm that you want to restart the statistics.
6. Click **OK** on the confirmation message.

### Updating a deployed SOAP Web service

You can update the definition and deployment information for a deployed SOAP Web service at any time.
Caution: You can freely update the definition and deployment information for a deployed SOAP Web service during development. However, once you deploy and enable a SOAP Web service for client access under production conditions, avoid making any changes to this information, as client implementations depend on its stability.

After production deployment, to make the same SOAP Web service available using different information (for example, to add a new operation or use a different SOAP Message Style), deploy a new SOAP Web service with the new information, using a different SOAP Web service name in the same WSA context. You can also deploy the SOAP Web service to a different WSA context.

Updating the definition and deployment information

You must disable a SOAP Web service before you can update it. You can verify if the SOAP Web service is disabled by checking its status.

Once you enable the SOAP Web service for client access, avoid making any changes to this deployment information, as client implementations depend on its stability. To make the same SOAP Web service available using different information (for example, to add a new operation or use a different WSDL Style/Use), deploy a new SOAP Web service for the application with a different name or in a different WSA context, and with the new information you want to use.

To update a SOAP Web service:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service you want to update is currently deployed.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
4. Under Deployed SOAP Web services, click Update.
5. Type the path and name of the WSM file for the SOAP Web service that you want to update.
6. Click Submit to update deployment information. The SOAP Web service name for an update is read-only and preset from the initial deployment. Initial values for other fields are set from existing information in the new WSM file, and you can change them as necessary. The fields that you can update include:
   - **SOAP Web service Namespace** — Any value you choose to uniquely qualify the names for operations and parameters used to define the SOAP Web service.
   - **SOAP Action** — A string that the client application is required to place in the SOAPAction HTTP header when accessing a SOAP Web service hosted by the WSA instance. The SOAPAction HTTP header is required for all HTTP messages that carry SOAP messages and is used by intervening security servers (such as firewalls) to determine if each HTTP message is allowed to pass through to its destination. The default is a blank string, but can be any string required by the intervening security servers on the network.
   - **Append to SOAP Action** — Indicates whether or not to append the specified SOAP Action value to any other SOAP Action values generated for the SOAP Web service SOAP messages by SOAP Web service clients.
   - **WSDL Style/Use** — Specifies the SOAP format to use when sending or receiving SOAP messages for this SOAP Web service. The value that you choose is dependent on what your anticipated SOAP Web service clients can support.
7. Click Submit. A confirmation message appears.

You might need to document additional information about the updated SOAP Web service for use by SOAP Web service client developers. For more information, see the chapter on client requirements in OpenEdge Development: Web Services.
Setting SOAP Web service application properties

You can view and set the value of each property for a deployed SOAP Web service, or you can reset the values of all properties to the current SOAP Web service defaults (stored in the default.props file for the WSA instance where the SOAP Web service is deployed).

You must disable a SOAP Web service before you can set most properties for it. You can verify if the SOAP Web service is disabled by checking its status. The only properties that you can set for an enabled SOAP Web service are:

- serviceFaultLevel
- serviceLoggingLevel

If you set other properties while the SOAP Web service is enabled, the property value changes take effect only after you disable and then enable the SOAP Web service again.

**Note:** Unlike WSA instance properties, the values for SOAP Web service properties are not stored in the ubroker.properties file. They are stored in the WebServiceFriendlyName.props XML file located in the Web application servlet directory for the WSA instance where the SOAP Web service is deployed.

To view and set SOAP Web service properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service whose properties you want to view and set is deployed.
3. Click the WebServices Adapter broker instance. The WebServices Adapter details page appears.
   
   For details on the properties that can appear in this list, see OpenEdge Application Server: Administration.

   **Note:** The actual list of properties displayed is a subset of all the properties, and changes depending on whether the session model of the SOAP Web service is session managed or session free. To verify the session model of a SOAP Web service, check its status. For more information, see Viewing the status of a SOAP Web service on page 174.

5. Click Edit.
6. Type the new value in the available field. For logical values, entering any value other than true results in a setting of false.
7. Click Submit when you finish. The changes are then written to the WebServiceFriendlyName.props file for the selected SOAP Web service.

Undeploying a SOAP Web service

You can undeploy a deployed SOAP Web service, which removes its deployment information, its properties, and other settings from the WSA instance context.

You must disable a SOAP Web service before you can undeploy it. You can verify if the SOAP Web service is disabled by checking its status.

To undeploy a SOAP Web service:
1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the WebServices Adapter broker instance where the SOAP Web service you want to undeploy is deployed.

3. Click the WebServices Adapter broker instance. The **WebServices Adapter** details page appears.

4. Under **Deployed Web Services**, click **Undeploy**.

If the SOAP Web service is already disabled, it is now undeployed from the WSA instance context and removed from the management console. If the SOAP Web service is still enabled, an error appears indicating that you must disable the SOAP Web service before undeploying it.

The changes are then written to the **WebServiceFriendlyName.props** file for the selected SOAP Web service.
Configuring WebSpeed Messengers and WebSpeed Transaction Servers

You can configure property settings for WebSpeed Messengers and examine their log files in OpenEdge Management and OpenEdge Explorer.

You can configure WebSpeed Transaction Server property settings and add new WebSpeed instances in OpenEdge Management and OpenEdge Explorer. You can also start, stop, and delete WebSpeed instances, as well as view their status and log files.

This chapter provides information about WebSpeed Messenger and WebSpeed Transaction Server configuration.

For details, see the following topics:

• WebSpeed Messenger administration
• WebSpeed Transaction Server administration
• Additional WebSpeed information

WebSpeed Messenger administration

The WebSpeed Messenger resides on your Web server machine. It picks up incoming application service requests from WebSpeed clients and directs them to a WebSpeed broker that supports that application service. The Messenger is either a CGI program, or an ISAPI or NSAPI process.

There are four different WebSpeed Messengers:

• **CGIIP Messenger** — Runs on almost all Web servers, but tends to have the slowest response times.
• **WSASP Messenger** — Is used to call WebSpeed applications from a Microsoft Active Server Page. It cannot coexist with any other Messenger on your Web server.

• **WSISA Messenger** — Runs on Microsoft IIS Web servers.

• **WSNSA Messenger** — Runs on Netscape Web servers.

You cannot create or delete WebSpeed Messengers from the management console. You can, however, edit the Messenger's properties, enable or disable the Messenger, and examine the Messenger's log file from the console.

### Configuring a WebSpeed Messenger

You can view or edit properties in the following categories for a WebSpeed Messenger in the management console:

- General
- Controlling NameServer
- Logging Setting
- SSL
- Advanced

You cannot create or delete WebSpeed Messengers from the management console. You can, however, edit the Messenger's properties, enable or disable the Messenger, and examine the Messenger's log file from the console.

Procedures for viewing or editing the configuration are described in the sections that follow.

### Viewing or modifying WebSpeed Messenger properties

You can view or modify General, Controlling NameServer, Logging Setting, SSL and Advanced properties for each WebSpeed Messenger.

To view or modify WebSpeed Messenger properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Messenger whose properties you want to view or modify.
3. Click the WebSpeed Messenger. The **Messenger** details page appears.
4. Under **Command and control**, click **Configuration**. From this page, you can do the following:
   - View a read-only display of the Messenger properties as described in WebSpeed Messenger properties on page 181.
   - Click **Edit** to modify the broker properties. For details, see Editing a WebSpeed Messenger configuration on page 184.
WebSpeed Messenger properties

WebSpeed Messenger General properties

Table 79: WebSpeed Messenger General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>The Messenger's working directory, where it stores temporary files and the Messenger log file</td>
</tr>
<tr>
<td>Messenger executable filename</td>
<td>The full path where the Messenger executable file resides</td>
</tr>
</tbody>
</table>

WebSpeed Messenger Controlling NameServer properties

Table 80: WebSpeed Messenger Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>If the check box is selected, the messenger is registered with the controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>The NameServer with which the WebSpeed Messenger registers. The WebSpeed Messenger then uses the NameServer to find a WebSpeed broker that supports the current application service. (If you did not choose to register the messenger with a NameServer, the field is dimmed and unavailable.)</td>
</tr>
<tr>
<td>Minimum NameServer client port</td>
<td>These properties are used to specify a range of UDP ports for the messenger instance. From this range, a messenger instance can select which UDP port to use when communicating with the NameServer. To limit the range, network administrators can specify a Minimum and a Maximum value. Specifying a range using Minimum and Maximum allows network administrators to limit the number of UDP ports needed to be open in the firewall when the NameServer client (in this case, the messenger) and the NameServer are separated by a firewall. The value for Minimum and Maximum can be zero (0) or a number between 1024 and 65535 inclusive. The Minimum value must be less than or equal to the Maximum value. If both the Minimum and Maximum values are set to zero (0), then a random port number within the implicit fixed range is dynamically assigned. How the port is assigned depends on the platform on which the messenger instance is running. If both Minimum and Maximum values are set to the same valid port number, then this UDP port number is used when communicating with the NameServer. The default value for both Minimum and Maximum is 0.</td>
</tr>
<tr>
<td>Maximum NameServer client port</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WebSpeed transaction server host</td>
<td>The WebSpeed Transaction Server host location.</td>
</tr>
<tr>
<td>WebSpeed transaction server port</td>
<td>The WebSpeed Transaction Server port location.</td>
</tr>
</tbody>
</table>

### WebSpeed Messenger Logging Setting properties

For complete information about these logging settings, see *OpenEdge Development: Debugging and Troubleshooting*.

Table 81: WebSpeed Messenger Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log filename</td>
<td>Location and name of the log file.</td>
</tr>
<tr>
<td>Logging level</td>
<td>A value that specifies the amount of information to be written to the Messenger log. Select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>Logging entry types</td>
<td>The one valid entry type for WebSpeed Messenger logging is MsgrTrace, which turns on logging for WebSpeed Messengers. The information logged depends on which Messenger is running and the logging level specified. For example:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — For CGIIP, logs the start and end of the request, and the values of the CONTENT_LENGTH, PATH_INFO and QUERY_STRING variables. For other Messengers, logs the environment variables.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Same as <strong>Basic</strong> but also includes the CGIIP environment variables and the environment variables in binary format for other Messengers.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Same as <strong>Verbose</strong>, but also includes the environment variables in binary format for CGIIP.</td>
</tr>
</tbody>
</table>

You can specify a logging level on the entry type. For example: `MsgrTrace:3`
Log file threshold size

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

Maximum number of log files

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of server log files to keep. The specified number represents the maximum total number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the Log file threshold size, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:

```
filename.######.extension
```

WebSpeed Messenger SSL properties

Table 82: WebSpeed Messenger SSL properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL client connections</td>
<td>If the check box is selected, this property directs that the connection to the WebSpeed Transaction Server use SSL tunneling.</td>
</tr>
<tr>
<td>Disable client verification of SSL host name</td>
<td>If the check box is selected, this property turns off host verification for an SSL connection. If cleared, the client compares the host name for the WebSpeed Transaction Server specified by Host Name (or as returned by any Controlling NameServer) with the Common Name specified in the server digital certificate, and raises an error if they do not match. With this parameter specified, the client never raises the error.</td>
</tr>
<tr>
<td>Disable SSL session reuse</td>
<td>If the check box is selected, the connection does not reuse the SSL session ID when reconnecting to the same WebSpeed Transaction Server.</td>
</tr>
<tr>
<td>Disable SSL session caching</td>
<td>Disables session caching. (Session caching allows a client to reuse a previously established session if it reconnects prior to the session cache timeout expiring.)</td>
</tr>
<tr>
<td>SSL session cache timeout</td>
<td>Specifies in seconds the length of time an SSL session will be held in the session cache. The default is 180 seconds.</td>
</tr>
</tbody>
</table>
### WebSpeed Messenger Advanced properties

#### Table 83: WebSpeed Messenger Advanced properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal administration command</td>
<td>If the check box is selected, the Messenger allows access to the WSMAdmin Web page. The WSMAdmin Web page provides links to customize error messages, enable or disable session logging, and validate the traffic flow between the Messenger and the WebSpeed broker/agents.</td>
</tr>
<tr>
<td>Session connection ID</td>
<td>If the check box is selected, WebSpeed Session Connection IDs are enabled.</td>
</tr>
<tr>
<td>Authorized IP list</td>
<td>The list of all IP addresses that are allowed to access the WSMAdmin Web page, if it has been enabled. If you add any addresses to this list, then only requests coming from addresses that are on the list will reach the WSMAdmin Web page. This gives you the ability to tailor access to the WSMAdmin Web page, so that only users connecting from certain sites can access the administration functions available on that page.</td>
</tr>
</tbody>
</table>

### Editing a WebSpeed Messenger configuration

You can edit the configuration of a WebSpeed Messenger.

To edit a WebSpeed Messenger configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Messenger whose configuration you want to edit.
3. From the list, click the WebSpeed Messenger. The Messenger details page appears.
4. Under Command and control, click Configuration. The messenger's Configuration page appears.
5. Click Edit, and make the changes to the properties. (For details about the properties, see WebSpeed Messenger properties on page 181.)
6. Click Save.

### Viewing the WebSpeed Messenger log file

You can view the log file for a WebSpeed Messenger by using the log file viewer. The log file viewer allows you to examine the log file through an HTML interface.

To access the log file viewer:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Messenger whose log file viewer you want to access.
3. From the list, click the WebSpeed Messenger. The Messenger details page appears.

4. Under Command and control section, click Log File Viewer. You can use the Log File Viewer in the following ways:
   - Use the Show field to control how many database log file entries display at one time. The number entered into the Show field cannot be less than 10.
   - Use the Overlap field to control how many entries are repeated from screen to screen.

   **Note:** The value in the Overlap field cannot be more than the number in the Show field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

   - Click Reload after changing the values in either the Show field or the Overlap field. If you do not reload, the viewer continues to display the previous values.
   - Click Go To to control which numbered entry in the log file the viewer begins its display with. For example, a value of 10 entered into the Go To field will begin the display at the tenth log file entry.

   **Note:** You must click Go To after entering a value in the Go To field or the viewer will not update its display.

   The default display of entries is in ascending order; choose Descending to change the display. Note that the Show field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

   - Click First to display the first x entries, where x is the value in the Show field.
   - Click Previous to display the previous x entries, where x is the value in the Show field.
   - Click Next to display the next x entries, where x is the value in the Show field.
   - Click Last to display the last x entries, where x is the value in the Show field.
   - To view additional log file entries without changing your current starting log file entry, leave the Go To field blank, change the value in the Show field, and click Reload.
   - If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the Log file status field.
   - OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer’s inactivity, the reloaded log file view might not match the previous log file view of that resource.
   - OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.

**WebSpeed Transaction Server administration**

With the WebSpeed Transaction Server, you can make WebSpeed applications available over the Web, an Intranet, or an Extranet.
You can use OpenEdge Management or OpenEdge Explorer to configure new WebSpeed Transaction Server instances and administer the configurations of existing instances. The WebSpeed Transaction Server installation provides one sample WebSpeed Transaction Server (wsbroker1). You can use this sample component as a template for creating and configuring additional WebSpeed Transaction Servers. Each WebSpeed Transaction Server is called an instance.

Working with the WebSpeed Transaction Server

Each WebSpeed Transaction Server consists of a WebSpeed broker and one or more WebSpeed agents. A WebSpeed agent is a single ABL Virtual Machine (AVM) instance running on a WebSpeed Transaction Server. Each WebSpeed agent executes ABL procedures on behalf of WebSpeed clients.

To work with a WebSpeed Transaction Server instance, double-click the WebSpeed folder from the management console's list frame. There is one pre-defined default WebSpeed Transaction Server (wsbroker1).

The following sections describe these WebSpeed Transaction Server actions:

- Configuring a WebSpeed Transaction Server on page 186
- Editing a WebSpeed Transaction Server configuration on page 205
- Creating a WebSpeed broker on page 205
- Starting and stopping a WebSpeed broker on page 205
- Adding or trimming WebSpeed agents on page 206
- Deleting a WebSpeed broker on page 207

Configuring a WebSpeed Transaction Server

You can view or edit the following configuration properties of a WebSpeed Transaction Server broker from the management console:

- Broker
- Agent
- SSL
- Messaging
- Environment variables

You can also create a new WebSpeed Transaction Server broker, delete a broker you no longer need, or view the broker's log files.

Each of these tasks is described in the sections that follow.

Viewing or modifying WebSpeed Transaction Server broker properties

You can view or modify the following WebSpeed Transaction Server Broker properties: general properties, owner information, controlling NameServer information, AppService Name List information, logging setting details, and advanced features.

To view or modify WebSpeed Transaction Server broker properties:
1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the WebSpeed Transaction Server broker whose properties you want to view or modify.

3. Click the WebSpeed Transaction Server broker. The WebSpeed details page appears.

4. In the Command and control section of the page, click Configuration. The WebSpeed Configuration page opens with that Broker's properties in focus. From this page, you can:

   - View a read-only display of the WebSpeed broker configuration properties as described in WebSpeed Transaction Server broker properties on page 187. All these tables follow this procedure.
   - Click Edit to modify the broker properties. For details, see Editing a WebSpeed Transaction Server configuration on page 205.

WebSpeed Transaction Server broker properties

The following sections describe the WebSpeed Transaction Server broker properties.

WebSpeed Broker General properties

Table 84: WebSpeed Broker General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto start</td>
<td>Indicates whether or not the server automatically starts when the controlling AdminServer starts.</td>
</tr>
<tr>
<td>Port number</td>
<td>The number of the TCP/IP port that the server broker listens on.</td>
</tr>
<tr>
<td>Working directory</td>
<td>Your working directory. The default is the working directory set during OpenEdge installation.</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>The operating mode for this server.</td>
</tr>
</tbody>
</table>

WebSpeed Broker Owner Information properties

Table 85: WebSpeed Broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group name</td>
<td>(UNIX only) The group name on UNIX.</td>
</tr>
<tr>
<td>Username</td>
<td>The username of an account that has system-administrative rights.</td>
</tr>
<tr>
<td>Password</td>
<td>The username account password.</td>
</tr>
<tr>
<td>Password Confirm</td>
<td>Confirmation of the username account password. This field appears only if you are editing the configuration. Note that this field does not appear if the AdminServer is running on a non-Windows host machine.</td>
</tr>
</tbody>
</table>
### WebSpeed Broker Controlling NameServer properties

Table 86: WebSpeed Broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>Indicates whether or not to register the broker with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>The name of the NameServer with which this broker registers. (If you did not select to register the broker with a NameServer, the field is unavailable.)</td>
</tr>
<tr>
<td>Registration mode</td>
<td>How the broker specifies its hostname if it is registering with a controlling NameServer. This hostname information is passed onto a client application when it attempts to connect to an application service that the broker supports. The choices are:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-IP</strong> — (Default) This setting is the most efficient mechanism, and can be used in most cases. It registers with the IP address of the machine where the broker is located.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-LocalHost</strong> — The broker registers with the hostname of the machine that it runs on. Use this setting when the broker runs on a machine with a single hostname and more than one IP address.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-HostName</strong> — The broker registers with the values specified in the HostName property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.</td>
</tr>
<tr>
<td>Registration host name</td>
<td>The name of the non-local host.</td>
</tr>
</tbody>
</table>

### WebSpeed Broker AppService Name List properties

Table 87: WebSpeed Broker AppService Name List properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application service names</td>
<td>A list to which you can add or delete application service names.</td>
</tr>
<tr>
<td>Supports default service</td>
<td>Indicates whether the AppServer broker supports the default service.</td>
</tr>
</tbody>
</table>

### WebSpeed Broker Logging Setting properties

Table 88: WebSpeed Broker Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The broker log filename.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Broker logging level (dynamic property)</td>
<td>A value that specifies the amount of information that is written to the broker log. The following values are available:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Broker logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td>Append to broker log file</td>
<td>Indicates whether or not a new broker log file should be created when the broker is started.</td>
</tr>
<tr>
<td>Broker logging entry types (dynamic property)</td>
<td>A comma-separated list of the following valid entry types for agent logging:</td>
</tr>
<tr>
<td></td>
<td>• <strong>UBroker.AutoTrim</strong> — Logs messages about automatic agent thread trimming by the broker based on the Auto trim timeout property. Information is logged at <strong>Basic</strong> and higher.</td>
</tr>
<tr>
<td></td>
<td>• <strong>UBroker.Basic</strong> — Logs messages about broker and agent startup and shutdown at the following logging levels:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logs NameServer registration and connections from clients.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logs information about all the property values read from the ubroker.properties file for that broker instance, and more information on the connection from clients.</td>
</tr>
<tr>
<td></td>
<td>• <strong>UBroker.ThreadPool</strong> — Logs messages about the pool of threads managed by the broker. These threads are used to control the client requests and the agent processes. This entry type logs messages when adding, removing, and communicating with these threads. Information is logged at <strong>Basic</strong> and higher.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td></td>
<td>Turn on the following logging entry types only at the request of Progress Software Corporation Technical Support:</td>
</tr>
</tbody>
</table>
UBroker.ClientFSM — Logs messages about the state of the client thread as it processes the requests, from the moment it receives a connection request to the end of the request. Information is logged at Basic and higher.

UBroker.ServerFSM — Logs messages about the state of the agent thread as it processes the requests, from the moment it receives a connection request to the end of the request. Information is logged at Basic and higher.

UBroker.ClientMsgStream — Logs messages about messages exchanged between the broker and client threads in binary format. Information is logged at Basic and higher.

UBroker.ServerMsgStream — Logs messages about messages exchanged between the broker and agent threads in binary format. Information is logged at Basic and higher.

UBroker.ClientMsgQueue — Logs messages about the client thread as it processes queued messages. Information is logged at Basic and higher.

UBroker.ServerMsgQueue — Logs messages about the agent thread as it processes queued message. Information is logged at Basic and higher.

UBroker.ClientMemTrace — Traces messages being processed by the client threads. Information is logged at Basic and higher.

UBroker.ServerMemTrace — Traces messages being processed by the agent threads. Information is logged at Basic and higher.

UBroker.Stats — Logs statistics about the number of requests, plus maximum, minimum, and average request wait and duration times. Information is logged at Basic and higher.

UBroker.All — Logs all UBroker entry types.

You can specify a separate logging level for each log entry type and use a wildcard (*) to specify multiple entry types. For example:

UBroker.Basic:3,UBroker.ThreadPool:4, UBroker.Client*
### Broker log file threshold size

A value of 0 or a value between 500,000 and 2,147,483,647, where 0 means there is no limit on the log file size other than what the operating system imposes.

### Maximum number of broker log files

The number of rolled-over log files to keep. The value can be 0 or a value between 2 and 999999, where 0 means there is no limit on the number of broker log files to keep. The specified number represents the maximum total number of log files to keep on disk at any time, including the current log file being written to.

When the file becomes equal to or greater than the **Broker log file threshold size**, the client process renames it and creates a new log file. The file is renamed as follows, where ###### is a number starting at 000001 and increasing to 999999, after which it rolls back over to 000001:

\[
\text{filename.######.extension}
\]

### WebSpeed Broker Advanced Features properties

#### Table 89: WebSpeed Broker Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum client instances</td>
<td>The maximum number of client connections that the broker can support concurrently. The default is 512, which is high enough to ensure that the number of client connections is virtually unlimited. Realistically, however, the system-level resources required to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for &quot;out of space&quot; or &quot;OutOfMemory&quot; you might need to reduce the maximum number of client connections to a more reasonable value. Note, however, that lowering this value can cause some client requests to be rejected (&quot;Exceeded Max Clients&quot;). Configuring and starting multiple brokers to handle higher client loads will alleviate the problem.</td>
</tr>
<tr>
<td>Priority weight (0-100) (dynamic property)</td>
<td>An integer value between 0 and 100 that influences the share of the workload that the selected AppServer receives. The larger the value, the heavier the load that is distributed to the server. The NameServer distributes client requests across all WebSpeed Transaction Servers that have the same application service in proportion to the <strong>Priority weight</strong> value. The default is 0. You can update this property dynamically. Any changes will affect all current and new brokers.</td>
</tr>
</tbody>
</table>
### Property | Description
--- | ---
**Registration retry (in seconds)** | After registering with its controlling NameServer, the WebSpeed Transaction Server periodically sends "keep-alive" messages to the NameServer to let the NameServer know that it is still active. The **Registration retry** value is the number of seconds that pass between "keep-alive" messages. The default is 30.

**Server startup timeout (dynamic property)** | Specifies the amount of time, in seconds, that the broker waits for an active agent to become available before starting a new agent process. The default is 3. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

**Request timeout (dynamic property)** | Specifies the amount of time, in seconds, that the broker waits for an agent to become available for processing a request. The broker waits for this period only if the **Maximum servers** setting has been reached. After the timeout has expired, the client receives a "no servers available" error message. The default is 15. You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.

**Auto trim timeout (dynamic property)** | The amount of time, in seconds, that the broker waits before automatically trimming the number of running agents. The broker keeps track of the maximum number of agents that are simultaneously busy during the interval you specify. At the end of the interval, the broker attempts to trim the number of agents to match either the **Maximum servers** for the interval, or the **Minimum servers**, whichever is greater. You can disable this feature by setting the timeout to zero (0). The default **Server Auto-trim Interval** is 1800 seconds. Progress Software recommends that you use this default setting to avoid unnecessary process management that might result from frequent starting and trimming of agents. You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| TCP/IP version                               | Internet Protocol for network communication:  
  • IPv4 — Accepts only IPv4 connections  
  • IPv6 — Allows IPv4 and IPv6 connections using mapped address (where supported)  
  The default is IPv4.  
  You must also set the `jvmArgs` property in the `ubroker.properties` file to override the default behavior of the JVM for IPv6 connections.  
  ```  
  jvmArgs=-Djava.net.preferIPv4Stack=false  
  -Djava.net.preferIPv6Addresses=true  
  ```                                                                                                                                              |
| jvmargs                                      | Java System Properties.                                                                                                                                                                                                 |
| Enable dynamic property updates              | Allows dynamic changes to occur to several of the instance's properties, without requiring you to shut down the instance.  
  This property is disabled by default.  
  **Note:** Any property that is dynamic is labeled as such in this document.                                                                     |
| Enable debugging through the broker (dynamic property) | Specifies whether or not the debugger is enabled; select the check box to enable it.  
  If you disable this property while the debugger is connected, the debugger connection terminates.  
  **Note:** The Broker debugger port number property must also be set to a valid value for the debugger to be properly enabled.  
  This property is disabled by default.                                                                                                                                 |
| Broker debugger port number (dynamic property) | Specifies the port number that the broker debugger will use when it starts. You can choose any valid value; if the port you specify is already in use, the debugger service will fail to start.  
  If you change the port number while the debugger service is active, the debugger service will be shut down and restarted using the new port number.  
  The default port number is 3099.                                                                                                                                 |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL for broker debugger (dynamic property)</td>
<td>Specifies whether the broker debugger should accept only SSL/TLS-encrypted connections. If you enable this property, the Progress Developer Studio debugger will be required to establish an SSL/TLS session. If you enable or disable the property while the debugger service is active, the debugger will be disconnected. The debugger service will be shut down and then restarted using the new selection. This option is disabled by default.</td>
</tr>
<tr>
<td>Use the broker's private key/digital certificate for debugger (dynamic property)</td>
<td>When enabled, indicates that, when using SSL, the broker debugger service should use the same private key as used by the AppServer agents. If this property is enabled, then the Debugger private key/digital certificate alias name and Password to access the debugger's key/certificate property settings are ignored. This property is enabled by default.</td>
</tr>
<tr>
<td>Debugger private key/digital certificate alias name (dynamic property)</td>
<td>The name of the private key that should be used for the debugger service when the agent's private key is not being used. This property must be defined if Enable SSL for broker debugger is enabled.</td>
</tr>
<tr>
<td>Password to access the debugger's key/certificate (dynamic property)</td>
<td>The password required to access the private key specified by Debugger private key/digital certificate alias name. The default for this property is blank.</td>
</tr>
<tr>
<td>Broker debugger passphrase (dynamic property)</td>
<td>An optional property that represents the passphrase required by the debugger to connect to the debugger service in the broker. If the debugger is connected to the debugger service when the passphrase is changed, the debugger remains connected. The next time the debugger attempts to connect to the debugger service, the new password is required. The default for this property is blank.</td>
</tr>
<tr>
<td>Broker debugger passphrase Confirm</td>
<td>Confirms the passphrase provided in Broker debugger passphrase.</td>
</tr>
<tr>
<td>Publish directory</td>
<td>The name of the local directory where you want r-code to be placed when you are publishing from Progress Developer Studio. If you are publishing remotely from Progress Developer Studio, you must set this property.</td>
</tr>
</tbody>
</table>
Viewing or modifying WebSpeed Agent properties

You can set the following agent properties for a WebSpeed Broker: general properties, logging setting details, pool range, and advanced features.

To view or modify WebSpeed Agent properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker whose agent properties you want to view or modify.
3. Click the WebSpeed Transaction Server broker. The WebSpeed details page appears.
4. In the Command and control section of the page, click Configuration. The WebSpeed Configuration page opens with the Broker properties in focus.
5. Click the Agent tab. From this page, you can do the following:
   - View a read-only display of the WebSpeed Agent properties as described in WebSpeed Agent properties on page 195.
   - Click Edit to modify the broker properties. For details, see Editing a WebSpeed Transaction Server configuration on page 205.

WebSpeed Agent properties

The following sections describe the WebSpeed Agent properties.

WebSpeed Agent General properties

Table 90: WebSpeed Agent General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server executable file</td>
<td>Either the default agent executable pathname or a different agent executable file of your choosing. You generally only need to specify a nondefault value if you have generated a new executable using the OEBUILD utility. If you update this property dynamically, any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td>Server startup parameters</td>
<td>The OpenEdge startup parameters that you want to specify to start each agent. You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.</td>
</tr>
<tr>
<td>PROPATH</td>
<td>The search path that agents use to locate ABL procedures that they execute. Specify a list of directory pathnames in the form of an OpenEdge PROPATH. Make sure that you have copied all of your WebSpeed Transaction Server procedures to one or more of the directories in the PROPATH.</td>
</tr>
</tbody>
</table>
### Minimum port number (dynamic property)

The minimum TCP/IP port number from a range that each agent process can listen on. When each agent starts, it allocates a port that is not being used from the specified range, up to the **Maximum port number** setting.

You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

### Maximum port number (dynamic property)

The maximum TCP/IP port number from a range that each agent can listen on. When each agent starts, it allocates a port that is not being used from the specified range, down to the **Minimum port number** setting.

You can update this property dynamically. Any changes will affect only new agents that start after the property value has changed; existing (running) brokers and agents are not updated.

### Server application mode

Development or Production.

### Flush statistical data (dynamic property)

How often the data is flushed by the agent to the broker. This is expressed as the number of remote procedure calls between flushes.

You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.

---

### WebSpeed Agent Logging Setting properties

Table 91: WebSpeed Agent Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The server log file name.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Server logging level</strong> (dynamic property)</td>
<td>A value that specifies the amount of information to be written to the agent log. You can select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Server logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Server logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td><strong>Append to server log file</strong></td>
<td>Indicates whether a new agent log file should be created when the WebSpeed Transaction Server is started.</td>
</tr>
<tr>
<td><strong>Server logging entry types</strong> (dynamic property)</td>
<td>A comma-separated list of valid entry types for agent logging:</td>
</tr>
<tr>
<td></td>
<td>• <strong>4GLTrace</strong> — Turns on the logging for the execution of the following ABL statements: RUN, FUNCTION (user-defined functions), PUBLISH and SUBSCRIBE. Logs information at <strong>Basic</strong> or higher.</td>
</tr>
<tr>
<td></td>
<td>• <strong>ASDefault</strong> — Combines the <strong>ASPlumbing</strong> and <strong>DB.Connects</strong> log entry types. This is the default value for AppServer agents.</td>
</tr>
<tr>
<td></td>
<td>• <strong>ASPlumbing</strong> — Turns on logging for different actions depending on the logging level specified:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logs messages when an AppServer agent starts and stops, and when client processes connect and disconnect from the AppServer agent.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Same as <strong>Basic</strong>, plus logging of the execution of the top level procedure (that is, the procedure that the client process asks the AppServer agent to execute on its behalf, but not any procedure that the top level procedure executes).</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Same as <strong>Verbose</strong>, plus messages about agent state, and status messages about read and write socket operations. Do not turn this on unless instructed to do so by Progress Software Technical Support.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>DB Connects</strong> —</td>
<td>Turns on logging of database connections and disconnections. The messages include database name and user ID number. Logs information at Basic or higher.</td>
</tr>
<tr>
<td><strong>DynObjects.DB</strong> —</td>
<td>Turns on logging for dynamic database-related objects (for example, TEMP-TABLE or DATASET) as they are created and destroyed. Logs information at Basic or Verbose.</td>
</tr>
<tr>
<td><strong>DynObjects.XML</strong> —</td>
<td>Turns on logging for dynamic XML-related objects (for example, x-document or x-noderef) as they are created and destroyed. Logs information at Basic or Verbose.</td>
</tr>
<tr>
<td><strong>DynObjects.Other</strong> —</td>
<td>Turns on logging for dynamic objects that do not match the other dynamic object categories (for example, procedure or server socket) as they are created and destroyed. Logs information at Basic or Verbose.</td>
</tr>
<tr>
<td><strong>ProEvents.UI.Char</strong> —</td>
<td>Turns on logging of keystroke events for printable characters. Logs information at Basic or higher.</td>
</tr>
<tr>
<td><strong>ProEvents.UI.Command</strong> —</td>
<td>Turns on logging of keystroke events for nonprintable characters at the following logging levels:</td>
</tr>
<tr>
<td></td>
<td>* Basic — Logs nonprintable keystrokes and a subset of Windows GUI events</td>
</tr>
<tr>
<td></td>
<td>* Verbose — Logs nonprintable keystrokes and all Windows GUI events</td>
</tr>
<tr>
<td><strong>ProEvents.Other</strong> —</td>
<td>Turns on logging of COM, asynchronous, and server socket events. Logs information at Basic or higher, beginning with events for which you have written triggers.</td>
</tr>
<tr>
<td><strong>QryInfo</strong> —</td>
<td>Turns on database query information logging at the following logging levels, with the same amount of information logged for each level:</td>
</tr>
<tr>
<td></td>
<td>* Basic — Logs static queries and initial dynamic queries</td>
</tr>
<tr>
<td></td>
<td>* Verbose — Logs static queries and multiple instances of dynamic queries</td>
</tr>
<tr>
<td><strong>SAX</strong> —</td>
<td>Turns on logging for various stages of execution using the ABL SAX parser.</td>
</tr>
</tbody>
</table>

You can update this property dynamically. Any changes will affect all current and new brokers and/or agents. You can specify a separate logging level for each log entry type, and you can use a wildcard (*) to specify multiple entry types. For example:

```
4GLTrace:3,ASPlubing:2,DynObjects.*
```
### WebSpeed Agent Pool Range properties

Table 92: WebSpeed Agent Pool Range properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial number of servers to start</td>
<td>The number of agents you want the broker to start initially.</td>
</tr>
<tr>
<td>Minimum servers</td>
<td>The minimum number of agents before the broker starts additional agents. If you trim the number of agents below this value, the server starts any additional agents needed to maintain the specified minimum when the next client connects.</td>
</tr>
<tr>
<td>Maximum servers</td>
<td>The maximum number of agents that this server can have running at the same time.</td>
</tr>
</tbody>
</table>
### WebSpeed Agent Advanced Features properties

#### Table 93: WebSpeed Agent Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application URL</strong></td>
<td>A server-relative URL (starting with a &quot;) to the path of the application. If the Web server redirects an incoming URL such that the SCRIPT_NAME and PATH_INFO variables do not refer to the application, setting this property allows an application to create self-referencing URLs when SCRIPT_NAME and/or PATH_INFO are not correct. If this option is set, the value of the applicationURL global variable available to applications is set to this value instead of a combination of SCRIPT_NAME and PATH_INFO.</td>
</tr>
<tr>
<td><strong>Default cookie path</strong></td>
<td>A server-relative URL (starting with a &quot;) to use for all cookies if not explicitly specified in the set-cookie( ) or delete-cookie( ) functions. The default is the value of the AppURL global variable.</td>
</tr>
<tr>
<td><strong>Default cookie domain</strong></td>
<td>Domain to send cookies. The default is blank, causing the Web browser to set cookies only for the current host. If cookies must be available to multiple hosts in the same domain by default, set that domain name here, for instance, .progress.com. Unless a value is specified for the domain option for set-cookie( ) and delete-cookie( ) functions, this value is used.</td>
</tr>
<tr>
<td><strong>File upload directory</strong></td>
<td>Specifies a directory where WebSpeed servers can upload files. Refer to the ubroker.properties file for an HTML snippet that shows how to use this feature. By default this capability is disabled. To enable it, specify an upload directory. Only text file upload is supported, not binary.</td>
</tr>
</tbody>
</table>
| **Binary upload max size** | Specifies the maximum size (in bytes) of binary files that WebSpeed handles. The application can access the contents of the binary files through the ABL language. Note that this property is not related to the File upload directory property; therefore, WebSpeed will not upload the file to a directory. You can set any of the following options:  
  - 0 — Binary upload is disabled.  
  - -1 — No maximum size of a binary upload; binary upload enabled.  
You can alternatively type any non-zero value greater than 0 as the maximum size of the binary upload. |
| **Debug mode**        | Whether the debug mode is enabled, disabled, or using the default. |
### Viewing or modifying WebSpeed SSL properties

You can set properties in the following WebSpeed Broker SSL categories: general properties and advanced features.

To view or modify WebSpeed SSL properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker whose SSL properties you want to view or modify.
3. Click the WebSpeed Transaction Server broker. The **WebSpeed** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **WebSpeed Configuration** page opens with the **Broker** properties in focus.
5. Click the **SSL** tab. From this page, you can:
   - View a read-only display of the WebSpeed SSL properties as described in **WebSpeed SSL properties** on page 201. Both of these tables follow this procedure.
   - Click **Edit** to modify the SSL properties. For details, see **Editing a WebSpeed Transaction Server configuration** on page 205.

### WebSpeed SSL properties

The following sections describe the WebSpeed SSL properties.

#### WebSpeed SSL General properties

**Table 94: WebSpeed SSL General properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL client connections</td>
<td>If the check box is selected, this property specifies that all connections to this WebSpeed Transaction Server must use SSL tunneling.</td>
</tr>
</tbody>
</table>
The alias name within the OpenEdge keystore of the private key and digital certificate entry to use to authenticate all connections to this WebSpeed Transaction Server. If not otherwise selected, the WebSpeed Transaction Server uses the default_server server certificate alias.

The password to use for accessing the private key and digital certificate. You must specify a password when you specify the select an OpenEdge keystore alias name.

Note: Your password is encrypted in the ubroker.properties file. If you set the password in ubroker.properties manually, you must specify the password as an encrypted value. You can obtain this value using the genpassword utility located in the bin directory of your OpenEdge installation.

If you use the default_server server certificate, it also has a default password that you do not need to specify.

Confirmation of the username account password. This field appears only if you are editing the configuration. Note that this field does not appear if the AdminServer is running on a non-Windows host machine.

WebSpeed SSL Advanced Features properties

Table 95: WebSpeed SSL Advanced Features properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable SSL session caching</td>
<td>If this property is checked, caching for the SSL client session is disabled.</td>
</tr>
<tr>
<td>SSL session cache timeout</td>
<td>The length of time, in seconds, that an SSL client session is held in the session cache, during which an SSL client can resume its session. The default is 180.</td>
</tr>
</tbody>
</table>

Viewing or modifying WebSpeed Messaging properties

You can view or modify several WebSpeed Messaging properties.

To view or modify WebSpeed Messaging properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker whose messaging properties you want to view or modify.
3. Click the WebSpeed Transaction Server broker. The WebSpeed details page appears.

4. In the Command and control section of the page, click Configuration. The WebSpeed Configuration page opens with the Broker properties in focus.

5. Click the Messaging tab. From this page, you can:
   - View a read-only display of the WebSpeed Messaging properties as described in, which follows this procedure.
   - Click Edit to modify the WebSpeed Messaging properties. For details, see Editing a WebSpeed Transaction Server configuration on page 205.

### WebSpeed Messaging properties

Table 96: WebSpeed Messaging properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SonicMQ ServerConnect enabled</td>
<td>Allows the asbroker or wsbroker agent to start a personal SonicMQ Adapter instead of going to a separate one running elsewhere.</td>
</tr>
<tr>
<td>SonicMQ ServerConnect broker log filename</td>
<td>A valid path and file name.</td>
</tr>
<tr>
<td>Broker logging level (dynamic property)</td>
<td>A value that specifies the amount of information to be written to the SonicMQ ServerConnect broker log. You can select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• None — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• Error Only — Log error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• Basic — Broker logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>•Verbose — Broker logging entry types determine the logged information, but it is typically more information than Basic.</td>
</tr>
<tr>
<td></td>
<td>• Extended — Broker logging entry types determine the logged information, but it is typically more information thanVerbose.</td>
</tr>
<tr>
<td>Append to broker log file</td>
<td>Indicates whether a new broker log file should be created when the SonicMQ ServerConnect broker is started, even if the broker log file already exists.</td>
</tr>
<tr>
<td>SonicMQ ServerConnect server log filename</td>
<td>A valid path and file name.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Server logging level</strong></td>
<td>A value that specifies the amount of information to be written to the SonicMQ ServerConnect server log. Each logging level name has the indicated numeric value. You can select from the following values in the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Broker logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Broker logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Broker logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers and/or agents.</td>
</tr>
<tr>
<td><strong>Append to server log file</strong></td>
<td>Indicates whether a new server log file should be created, even if the broker log file already exists, when the SonicMQ ServerConnect instance is started.</td>
</tr>
</tbody>
</table>

**Setting or deleting WebSpeed environment variables**

You can set or delete environment variables. For WebSpeed, environment variables that appear in the list are scoped to:

- The process in which the WebSpeed Transaction Server broker executes
- Every WebSpeed Transaction Server agent that the WebSpeed Transaction Server broker starts

To set or delete WebSpeed environment variables:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker whose environmental variables you want to set.
3. Click the WebSpeed Transaction Server broker. The **WebSpeed** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The broker’s **Configuration** page appears.
5. Click **Edit**.
6. Click **Environment Variables**.
7. Enter each variable name and then either provide its value in the format name=value, or remove an existing variable.
8. Click **Save** when you finish adding (or deleting) the variables and their values.
Editing a WebSpeed Transaction Server configuration

You can edit a WebSpeed Transaction Server configuration.

To edit a WebSpeed Transaction Server configuration:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker whose configuration you want to edit.
3. Click the WebSpeed Transaction Server broker. The WebSpeed details page appears.
4. In the Command and control section of the page, click Configuration. The WebSpeed Configuration page opens.
5. Click Edit. Make the changes you want to the properties. (For details about the properties, see WebSpeed Messenger properties on page 181, WebSpeed Transaction Server broker properties on page 187, WebSpeed Agent properties on page 195, WebSpeed SSL properties on page 201, WebSpeed Messaging properties on page 203.)
6. Click Save. Any changes you make are also saved in the ubroker.properties file.

Creating a WebSpeed broker

OpenEdge Management and OpenEdge Explorer provide a sample WebSpeed broker instance (wsbroker1) with a default configuration. Each new instance you create uses the same default configuration, which you can edit.

To create a WebSpeed broker instance:

1. From the drop-down for Resources on the management console menu, click New OpenEdge Resource > WebSpeed. The WebSpeed Configuration page appears.
2. Type the name of the new WebSpeed broker instance in the field provided.
3. Click Save. The WebSpeed Configuration page appears, allowing you to configure the WebSpeed broker's properties.

   Note: The WebSpeed broker instance name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured WebSpeed broker instance names.

Each new WebSpeed broker instance you create uses the default configuration. However, several properties, such as port numbers, must have unique values for each WebSpeed broker in order for the instance to operate properly.

Starting and stopping a WebSpeed broker

From the management console, you can start either a local WebSpeed broker or a WebSpeed broker running under a remote AdminServer on another machine. You must first connect to the AdminServer that manages the WebSpeed broker you want to start.

You can also choose to start a WebSpeed broker automatically when the AdminServer starts.
Prior to starting a WebSpeed broker using SonicMQ ServerConnect, you must first configure the WebSpeed broker as SonicMQ ServerConnect enabled.

To start or stop a WebSpeed broker, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

Starting a WebSpeed broker automatically

You can set a WebSpeed broker to start automatically when the AdminServer starts.

To start a WebSpeed broker automatically:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker you want to start automatically.
3. Click the Edit icon. The **WebSpeed** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **WebSpeed Configuration** page appears.
5. Click **Edit**.
6. Select the **Auto start** option by clicking in the check box.
7. Click **Save**.

Adding or trimming WebSpeed agents

Using OpenEdge Management or OpenEdge Explorer, you can start additional WebSpeed agents, up to the number specified in the Maximum Agents property setting.

You can also trim running WebSpeed agents, down to the Minimum Agents property setting.

More information about trimming WebSpeed Agents

To better understand the use of the Agent Auto-trim feature, consider a scenario involving a broker serving a client/server application running in stateless mode. Assume that the following property values are set:

- Minimum Agent/Server Instances: 10
- Initial Agent/Server Instances to Start: 10
- Maximum Agent/Server Instances: 50
- Agent/Server Auto-trim Interval: 1800

When the broker starts, it starts 10 agents. This also marks the start of the first Agent/Server Auto-trim Interval. Assume that the broker is idle for 30 minutes (it is just before people arrive for work, for instance). At the end of the 30-minute interval, the highest number of agents that were busy at the same time would be zero. (Since there were no requests made of any of the agents, none of the agents was busy.) However, since zero is less than the Minimum Agent/Servers Instances value of 10, no automatic trimming takes place. This marks the start of the second interval.
As the second interval starts, employees begin their workday. There is a sudden demand for agents; in fact, the broker receives so many requests that it must start 20 additional agents. Assume that the high volume of workload, and demand on the agents, continues through the second interval. At the interval’s end, the maximum number of busy agents is 30, which is the number the broker scaled up to in response to high demand. Thirty agents are running, and the maximum busy level is 30. No trimming occurs.

During the next interval, the request load diminishes somewhat. At the end of the interval, only 20 agents are concurrently busy. The maximum number of busy agents for the interval remains at 30, since 30 were busy at the start of the interval. Thirty agents are running, and the maximum busy level is 30. No trimming occurs.

Assume that all employees have to attend a company-wide meeting that begins midway through the interval. The highest number of agents busy at that same time was 20. At the end of the interval, there are 30 agents running, with zero agents currently busy. Since the maximum busy level is 20 (at the start of the interval), and the number of running agents is 30 (all are available since everyone is at the company meeting), the broker attempts to trim back the number of agents to 20.

Assume that the company meeting ran for another half-hour, and the entire next interval elapsed without any requests. The maximum busy agents is zero, with 20 currently running. Since the Minimum Agent/Server Instances property value is set to 10, the broker attempts to trim back to 10 agents.

Keep in mind that the lower you set the Agent/Server Auto-trim Interval value, the more quickly the broker reacts to a drop in the workload. Under a reasonably steady work load, the broker should trim the number of running agents to the number needed. However, if the workload fluctuates a great deal and the Agent/Server Auto-trim Interval is set too low, the broker may trim agents that it must then restart. As stated earlier, repetitive trimming and restarting causes the system to waste resources on unnecessary process management.

### Adding or trimming WebSpeed agents

To add or trim WebSpeed agents:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the WebSpeed Transaction Server broker to which you want to add or trim agents.
3. Click the WebSpeed Transaction Server broker. The **WebSpeed** details page appears.
4. If the WebSpeed broker is not started, click **Broker Control**, then click **Start WebSpeed**.
5. When the WebSpeed broker is running, return to the **Details** page.
6. In the **Command and control** section of the page, click **Agent Pool Control**.
7. Review the **Agent pool initial configuration**, the **Agents state**, and the **Agents pool summary** information.
8. To add or trim agents, choose the corresponding option in the **Add/Trim** dropdown menu, and type the number of agents in the box provided.
9. Click **Submit**.

### Deleting a WebSpeed broker

You can delete any WebSpeed broker instance listed in the management console as long as the instance is not running. If the instance is running, you must stop it and then delete it.

To delete a WebSpeed broker instance:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter, or search for, the WebSpeed Transaction Server broker instance you want to delete.
3. Click the WebSpeed Transaction Server broker instance. The WebSpeed details page appears.

4. Click Delete, and then click OK to confirm the deletion.

The OpenEdge Management Resources page appears. A set of next-step options that are related to resources are available for your use.

Additional WebSpeed information

You can find additional information about using OpenEdge Management and/or OpenEdge Explorer with WebSpeed in OpenEdge Management: Servers, DataServers, Messengers, and Adapters. See that document for details, including specifics about the following:

- Working with the WebSpeed log file monitor and viewer
- Killing a WebSpeed process
Configuring SonicMQ Adapters

You can work with SonicMQ Adapter instances in OpenEdge Management and OpenEdge Explorer. For details, see the following topics:

- SonicMQ Adapter Broker configuration and administration
- Working with the SonicMQ Adapter
- Configuring a SonicMQ Adapter
- Editing a SonicMQ Adapter configuration
- Creating a SonicMQ Adapter broker
- Deleting a SonicMQ Adapter broker
- Starting or Stopping a SonicMQ Adapter
- Viewing the status of a SonicMQ Adapter broker
- Viewing the SonicMQ Adapter log file

SonicMQ Adapter Broker configuration and administration

The OpenEdge SonicMQ® Adapter allows OpenEdge applications to communicate via JMS Messaging through SonicMQ. The SonicMQ Adapter consists of three connection modes:
• **OpenEdge Adapter for SonicMQ ClientConnect (ClientConnect)** — ClientConnect is for OpenEdge clients and runs transparently as a background process in conjunction with an ABL client or OpenEdge Application Server agent process, with a single adapter process per client process. The application running on the ABL client handles messaging control. ClientConnect takes little or no configuration.

• **OpenEdge Adapter for SonicMQ ServerConnect (ServerConnect)** — ServerConnect is for OpenEdge Application Servers (WebSpeed and AppServer). With this configuration there is a single adapter process per AppServer process, allowing multiple Application Server agents to connect to this single adapter process. ServerConnect is configured at the server.

• **OpenEdge Adapter for SonicMQ BrokerConnect (BrokerConnect)** — BrokerConnect is for ABL client applications. It runs as a separate server process to handle OpenEdge client requests. BrokerConnect is a Unified Broker product, part of the AppServer administration framework. You can use OpenEdge Management or OpenEdge Explorer and the command-line tools adaptconfig and adaptman on all supported platforms, to manage BrokerConnect. You can also edit its properties in the `ubroker.properties` file. No configuration is required within the SonicMQ environment.

**Note:** For BrokerConnect, the OpenEdge installation program creates one instance of the OpenEdge SonicMQ Adapter. In most circumstances, this single adapter instance is sufficient. Although it is possible to create additional instances, there is usually no reason to run multiple OpenEdge SonicMQ Adapter instances on the same host. Each instance of BrokerConnect runs as a broker process. This process is multi-threaded, with one thread for each active client application; it can connect to any SonicMQ Broker. With the SonicMQ Adapter, you can write ABL applications that access JMS messaging using SonicMQ.

You can use OpenEdge Management or OpenEdge Explorer to configure a new instance of BrokerConnect and administer the configuration of existing instances.

**Working with the SonicMQ Adapter**

To work with a SonicMQ Adapter, select the instance. For information about the tasks you can perform, see the following sections:

• **Configuring a SonicMQ Adapter** on page 210
• **Starting or Stopping a SonicMQ Adapter** on page 220
• **Creating a SonicMQ Adapter broker** on page 219
• **Starting or Stopping a SonicMQ Adapter** on page 220
• **Viewing the status of a SonicMQ Adapter broker** on page 221
• **Deleting a SonicMQ Adapter broker** on page 220
• **Viewing the SonicMQ Adapter log file** on page 222

**Configuring a SonicMQ Adapter**

From the management console, you can view or edit the following default configuration properties of a SonicMQ Adapter:

• **Broker**
You can also create a new SonicMQ Adapter instance, set environment variables, view an instance's status, delete an instance you no longer need, or view the instance's log file.

Each of these tasks is described in the following sections.

### Viewing or modifying SonicMQ Adapter broker properties

You can view or modify general, owner information, controlling NameServer, appservice name list, logging setting, and advanced features properties for a SonicMQ Adapter instance.

To view or modify SonicMQ Adapter broker properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the SonicMQ Adapter broker instance whose properties you want to view or modify.
3. Click the SonicMQ Adapter broker instance. The **SonicMQ Adapter** details page appears.
4. In the **Command and control** section of the page, click **Configuration**. The **SonicMQ Adapter Configuration** page opens with the **Broker** properties in focus. From this page, you can:
   - View the SonicMQ Adapter instance's broker properties as described in **SonicMQ Adapter broker properties** on page 211.
   - Click **Edit** to modify the properties. For details, see **Starting or Stopping a SonicMQ Adapter** on page 220.

### SonicMQ Adapter broker properties

The following sections describe the SonicMQ Adapter broker properties.

#### SonicMQ Adapter Broker General properties

**Table 97: SonicMQ Adapter Broker General properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working directory</td>
<td>The SonicMQ Adapter working directory.</td>
</tr>
<tr>
<td>Port number</td>
<td>The number of the TCP/IP port that the SonicMQ Adapter broker listens to. Each broker defined in the <strong>ubroker.properties</strong> file must have a unique port number.</td>
</tr>
<tr>
<td>Auto start</td>
<td>A check box that you select if you want the SonicMQ Adapter broker to start automatically when the AdminServer starts.</td>
</tr>
</tbody>
</table>
SonicMQ Adapter Broker Owner Information properties

Table 98: SonicMQ Adapter Broker Owner Information properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group name</td>
<td>(UNIX only) The name of the group in UNIX.</td>
</tr>
<tr>
<td>Username</td>
<td>The ID of a user account that has system-administration rights.</td>
</tr>
<tr>
<td>Password</td>
<td>The user account's password. Note that this field is labeled Group Name if the AdminServer runs on a non-Windows system.</td>
</tr>
</tbody>
</table>

SonicMQ Adapter Broker Controlling NameServer properties

Table 99: SonicMQ Adapter Broker Controlling NameServer properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register with NameServer</td>
<td>A check box that indicates whether or not to register the SonicMQ Adapter with a controlling NameServer.</td>
</tr>
<tr>
<td>Controlling NameServer</td>
<td>The name of the NameServer with which this SonicMQ Adapter registers. (If you did not select to register the broker with a NameServer, the field is dimmed and unavailable.)</td>
</tr>
<tr>
<td>Registration mode</td>
<td>How the broker specifies its host name if it is registering with a controlling NameServer. This host name information is passed onto a client application when it attempts to connect to an application service that the broker supports. The choices are:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-IPAddress</strong> — (Default) This setting is the most efficient mechanism and can be used in most cases. It registers with the IP address of the machine where the broker is located.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-LocalHost</strong> — The broker registers with the host name of the machine that it runs on. Use this setting when the broker runs on a machine with a single host name and more than one IP address.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Register-HostName</strong> — The broker registers with the values specified in the host name property. Use this setting when your clients need a fully qualified host name to connect to a broker in a different DNS domain.</td>
</tr>
<tr>
<td>Use host name</td>
<td>Specifies the host name value to be used if the Registration mode property is set to Register-HostName. In all other cases, this property is ignored.</td>
</tr>
</tbody>
</table>
**SonicMQ Adapter Broker AppServer Name List property**

Table 100: SonicMQ Adapter Broker AppServer Name List property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application service names</td>
<td>The application services that this SonicMQ Adapter broker registers with the controlling NameServer. A SonicMQ Adapter client connecting to the broker must specify one of these service names. You can add or delete service names from the list.</td>
</tr>
</tbody>
</table>

**SonicMQ Adapter Broker Logging Setting properties**

Table 101: SonicMQ Adapter Broker Logging Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker log filename</td>
<td>The broker log file for this SonicMQ Adapter broker instance.</td>
</tr>
<tr>
<td>Broker logging level</td>
<td>The amount of information written to the broker log. The possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong> — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Error Only</strong> — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Basic</strong> — Logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Verbose</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Basic</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Extended</strong> — Logging entry types determine the logged information, but it is typically more information than <strong>Verbose</strong>.</td>
</tr>
<tr>
<td>Append to broker log file</td>
<td>A check box that you select if you want the SonicMQ Adapter broker to look for the log file each time it starts up. If the log file is present, the new information is added to the end of the existing file. If there is no log file, a new file is created.</td>
</tr>
<tr>
<td>Broker logging entry types</td>
<td>A single entry or comma delimited list of logging entry types.</td>
</tr>
</tbody>
</table>
A limit on how large the log file can get (in bytes). When the log is full, the broker creates a new log file with a sequence number (1 to 999,999).
Possible values: 500000 - 2147483647.

A limit on how many broker log files are kept on the system when there is a threshold limit. The value represents the number of log files including the current one. Includes any rolled over log files that already exist when process started.

### SonicMQ Adapter Broker Advanced Features properties

**Table 102: SonicMQ Adapter Broker Advanced Features properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum client instances</strong></td>
<td>The maximum number of client connections the SonicMQ Adapter broker can support concurrently. The default value 512 is high enough to ensure that the number of client connections is unlimited.</td>
</tr>
<tr>
<td></td>
<td>Realistically, however, the system-level resources required to support an unlimited number of clients might be exhausted before this limit is reached. If you see broker log entries that contain Exception Messages for out of space or out of memory, you might need to reduce the maximum number of client connections to a more reasonable value. Note that lowering this value can cause some client requests to be rejected (Exceeded Max Clients). Configuring and starting multiple brokers to handle higher client loads will alleviate the problem.</td>
</tr>
<tr>
<td><strong>Priority weight (dynamic property)</strong></td>
<td>Defines a load-balancing priority for this broker relative to other brokers.</td>
</tr>
<tr>
<td></td>
<td>You can update this property dynamically. Any changes will affect all current and new brokers.</td>
</tr>
<tr>
<td><strong>Registration retry</strong></td>
<td>The number of seconds that pass between broker keep-alive messages. After registering with its controlling NameServer, the SonicMQ Adapter broker periodically sends keep-alive messages to the NameServer to let the NameServer know that the broker is still active. (The default is 30.)</td>
</tr>
<tr>
<td><strong>TCP/IP version</strong></td>
<td>The TCP/IP version.</td>
</tr>
</tbody>
</table>
### Viewing or modifying SonicMQ Adapter server properties

You can view or modify general, logging setting, or pool range properties for a SonicMQ Adapter server instance.

To view or modify SonicMQ Adapter server properties:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the SonicMQ Adapter broker instance whose server properties you want to view or modify.

3. Click the SonicMQ Adapter broker instance. The **SonicMQ Adapter** details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The **SonicMQ Adapter Configuration** page opens with the **Broker** properties in focus.

5. Click the **Server** tab. From this page, you can:
   - View a read-only display of the SonicMQ Adapter instance’s server properties as described in **SonicMQ Adapter server properties** on page 216. These tables follow this procedure.
   - Click **Edit** to modify the broker properties. For details, see **Starting or Stopping a SonicMQ Adapter** on page 220.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable dynamic property updates</td>
<td>Allows dynamic changes to occur to several of the instance’s properties, without requiring you to shut down the instance. This property is disabled by default.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Any property that is dynamic is labeled as such in this document.</td>
</tr>
<tr>
<td>AppServer Keepalive</td>
<td>Allows the AppServer to recognize that a client bound to it is no longer connected.</td>
</tr>
<tr>
<td>serverASK Activity Timeout</td>
<td>Specifies the number of seconds with no activity that the AppServer waits before sending a message to the client requesting that the connection be maintained. The property is used only if the AppServer Keepalive property is set to <strong>allowServerASK</strong>.</td>
</tr>
<tr>
<td>serverASK Response Timeout</td>
<td>Specifies the number of seconds with no activity after the AppServer sends the client a message (requesting that the connection be maintained) before the client is disconnected. This property is used only if the AppServer Keepalive property is set to <strong>allowServerASK</strong>.</td>
</tr>
</tbody>
</table>
SonicMQ Adapter server properties

The following sections describe the SonicMQ Adapter server properties.

SonicMQ Adapter Server General property

Table 103: SonicMQ Adapter Server General property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server startup parameters</td>
<td>The startup parameters that you want to specify for the SonicMQ server</td>
</tr>
</tbody>
</table>

SonicMQ Adapter Server Logging Settings properties

Table 104: SonicMQ Adapter Server Logging Settings properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The server log file for this SonicMQ Adapter broker instance.</td>
</tr>
<tr>
<td>Server logging level</td>
<td>The amount of information written to the server log. The possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• None — Log no entries. This is equivalent to turning logging off.</td>
</tr>
<tr>
<td></td>
<td>• Error Only — Log OpenEdge error messages. This includes all error messages and is unrelated to the entry types specified. Errors continue to be logged at all higher levels.</td>
</tr>
<tr>
<td></td>
<td>• Basic — Logging entry types determine the logged information. Each entry type generates at least some output. This is the default.</td>
</tr>
<tr>
<td></td>
<td>• Verbose — Logging entry types determine the logged information, but it is typically more information than Basic.</td>
</tr>
<tr>
<td></td>
<td>• Extended — Logging entry types determine the logged information, but it is typically more information than Verbose.</td>
</tr>
<tr>
<td>Append to server log file</td>
<td>A check box that you select if you want the SonicMQ Adapter broker to look for the log file each time it starts up. If the log file is present, the new information is added to the end of the existing file. If there is no log file, a new file is created.</td>
</tr>
</tbody>
</table>
SonicMQ Adapter Server Pool Range properties

Table 105: SonicMQ Adapter Server Pool Range properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial number of threads to start</td>
<td>The number of threads you want to start initially within a SonicMQ Adapter broker process. The default is 20.</td>
</tr>
<tr>
<td>Minimum threads</td>
<td>The minimum number of threads to start. The default is 10.</td>
</tr>
<tr>
<td>Maximum threads</td>
<td>The maximum number of threads to start. The default is 30.</td>
</tr>
</tbody>
</table>

Viewing or modifying SonicMQ Adapter SSL properties

You can view or modify general and advanced features SSL properties for a SonicMQ Adapter instance.

To view or modify SonicMQ Adapter SSL properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the SonicMQ Adapter broker instance whose SSL properties you want to view or modify.
3. Click the SonicMQ Adapter broker instance. The SonicMQ Adapter details page appears.
4. In the Command and control section of the page, click Configuration. The SonicMQ Adapter Configuration page opens with the Broker properties in focus.
5. Click the SSL tab. From this page, you can:
   - View a read-only display of the SonicMQ Adapter instance's SSL properties as described in SonicMQ Adapter SSL properties on page 217. These tables follow this procedure.
   - Click Edit to modify the broker properties, as described in Editing a SonicMQ Adapter configuration on page 219. For details, see Editing a SonicMQ Adapter configuration on page 219.

SonicMQ Adapter SSL properties

The following sections describe the SonicMQ Adapter SSL properties.

SonicMQ Adapter SSL General properties

Table 106: SonicMQ Adapter SSL General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSL client connections</td>
<td>If this property is selected, all connections to this SonicMQ Adapter must use SSL tunneling.</td>
</tr>
</tbody>
</table>
### SonicMQ Adapter SSL Advanced Features properties

**Table 107: SonicMQ Adapter SSL Advanced Features properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable SSL session caching</td>
<td>If this property is checked, caching for the SSL client session is disabled.</td>
</tr>
<tr>
<td>SSL session cache timeout</td>
<td>The length of time, in seconds, that an SSL client session is held in the session cache, during which an SSL client can resume its session. The default is 180.</td>
</tr>
</tbody>
</table>

### Setting or deleting SonicMQ Adapter environment variables

You can set or delete SonicMQ Adapter environment variables.

To set or delete SonicMQ Adapter environment variables:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the SonicMQ Adapter broker instance whose environment variable you want to set.
3. Click the SonicMQ Adapter broker instance. The SonicMQ Adapter details page appears.
4. In the Command and control section of the page, click Configuration. The broker’s SonicMQ Adapter Configuration page appears.
5. Click Edit.
6. Click Environment Variables.
7. Enter each variable name and then provide its value in the format name=value.
8. Click Save when you finish adding (or deleting) the variables and their values.

Enabling or disabling a SonicMQ Adapter broker

You can enable or disable a SonicMQ Adapter broker.
To enable or disable a SonicMQ Adapter broker:
1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the SonicMQ Adapter broker instance you want to enable or disable.
3. Click the SonicMQ Adapter broker instance. The SonicMQ Adapter details page appears.
4. In the Command and control section of the page, click Control. The broker’s Control page appears.
5. Click Edit, and either select the Enabled check box to enable the broker or clear the check box to disable the broker.
6. Click Save.

Editing a SonicMQ Adapter configuration

You can edit a SonicMQ Adapter configuration in the management console.
To edit a SonicMQ Adapter configuration:
1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the SonicMQ Adapter broker instance whose configuration you want to edit.
3. Click the SonicMQ Adapter broker instance. The SonicMQ Adapter details page appears.
4. In the Command and control section of the page, click Configuration. The Configuration page appears.
5. Click Edit. (For details about the properties, see SonicMQ Adapter broker properties on page 211, SonicMQ Adapter server properties on page 216, and SonicMQ Adapter SSL properties on page 217.)
6. Make the configuration changes, and then click Save.

Creating a SonicMQ Adapter broker

OpenEdge Management and OpenEdge Explorer provide a sample SonicMQ Adapter broker instance (SonicMQ1) and use it as a default. A new instance you create also uses the sample configuration, which you can modify.
To create a SonicMQ Adapter broker:

1. From the drop-down for Resources on the management console menu, click New OpenEdge Resource > SonicMQ Adapter. The SonicMQ Adapter Configuration page appears.

2. Type the name of the new broker instance in the field provided.

   **Note:** The SonicMQ Adapter broker instance name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured SonicMQ Adapter broker instance names.

3. Click Save. The SonicMQ Adapter Configuration page appears, allowing you to configure the instance's properties.

You can now set the configuration properties for this instance. When a new instance is created, it has a copy of the default configuration. However, several properties, such as port numbers, must have unique values for each SonicMQ Adapter broker in order for the instance to operate properly.

**Deleting a SonicMQ Adapter broker**

You can delete an inactive SonicMQ Adapter broker instance from the management console. If the instance is running, you cannot delete it. You must stop the instance before you can delete it.

To delete a SonicMQ Adapter broker:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter, or search for, the SonicMQ Adapter broker instance you want to delete.

3. Click the SonicMQ Adapter broker instance. The SonicMQ Adapter details page appears.

4. Click Delete, and then click OK to confirm the deletion.

   The OpenEdge Management Resources page appears. A set of next-step options that are related to resources are available for your use.

**Starting or Stopping a SonicMQ Adapter**

From the management console, you can start or stop a SonicMQ Adapter instance. You can also choose to start the instance automatically when the AdminServer starts.

To start or stop a SonicMQ Adapter instance, refer to the Starting or Stopping OpenEdge resources section in OpenEdge Management: Servers, DataServers, Messengers, and Adapters.

**Starting a SonicMQ Adapter automatically**

You can set a SonicMQ Adapter to start automatically when the AdminServer starts.

To start a SonicMQ Adapter automatically:
1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the SonicMQ Adapter broker instance you want to start automatically.

3. Click the SonicMQ Adapter broker instance. The **SonicMQ Adapter** details page appears.

4. In the **Command and control** section of the page, click **Configuration**. The **SonicMQ Adapter Configuration** page appears.

5. Click **Edit**.

6. On the **Broker** tab, select the **Auto start** option by clicking in the check box.

7. Click **Save**.

---

### Viewing the status of a SonicMQ Adapter broker

You can view the status of an active SonicMQ Adapter broker in the management console.

To view the status of a SonicMQ Adapter broker:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the SonicMQ Adapter broker instance you want to start automatically.

3. Click the SonicMQ Adapter broker instance. The **SonicMQ Adapter** details page appears.

4. In the **Operations Views** section of the page, click **Status**. A Status summary appears and provides the following information:

   - **Host** — The name of the host machine.
   - **Broker Name** — The name of the broker whose status you are viewing.
   - **Operating Mode** — For SonicMQ Adapter brokers, the operating mode is stateless.
   - **Broker Status** — The current state of the broker.
   - **Broker Port** — The TCP/IP port number that the broker listens to.
   - **Broker PID** — The process ID number of the broker.
   - **Active Servers** — The number of active SonicMQ Adapter brokers.
   - **Busy Servers** — The number of SonicMQ Adapter brokers that are currently processing requests.
   - **Locked Servers** — The number of SonicMQ Adapter brokers that have been locked.
   - **Available Servers** — The number of available SonicMQ Adapter brokers.
   - **Active Clients (now, peak)** — The number of client connections at the current point, and the highest number of connections since the broker started.
   - **Client Queue Depth (cur, max)** — The number of clients waiting for agents to become available to service their request. The current value (cur) represents the number of waiting clients at the moment the status is displayed, and the maximum value (max) represents the largest number of clients waiting concurrently since the broker was started.
   - **Total Requests** — The total number of requests from clients since the broker started.
   - **Rq Wait (max, avg)** — The maximum and average time the client had to wait for an available broker.
   - **Rq Duration (max, avg)** — The maximum and average time the request required.

For each server, the following details are also provided:
• Svris — The process ID number of the broker process
• State — The current state of the broker process
• Port — The TCP/IP port number used by the broker
• nRq — The total number of messages sent to the broker since it started
• nRcvd — The total number of messages received by the broker since it started
• nSent — The total number of requests sent by the broker
• Started — The time stamp of when the broker process started
• Last Change — The time stamp of when the server process last changed state

Viewing the SonicMQ Adapter log file

You can view the log file for a SonicMQ Adapter by using the log file viewer. The log file viewer allows you to examine the log file through an HTML interface.

To access the log file viewer:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the SonicMQ Adapter broker instance whose log file viewer you want to access.
3. In the Command and control section of the page, click Log File Viewer of Broker or Log File Viewer of Servers.

You can use the Log File Viewer in the following ways:

• Use the Show field to control how many database log file entries display at one time. The number entered into the Show field cannot be less than 10.
• Use the Overlap field to control how many entries are repeated from screen to screen.

Note: The value in the Overlap field cannot be more than the number in the Show field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

• Click Reload after changing the values in either the Show field or the Overlap field. If you do not reload, the viewer continues to display the previous values.

• Click Go To to control which numbered entry in the log file the viewer begins its display with. For example, a value of 10 entered into the Go To field will begin the display from the tenth log file entry.

Note: You must click Go To after entering a value in the Go To field or the viewer will not update its display.

The default display of entries is in ascending order; choose Descending to change the display. Note that the Show field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

• Click First to display the first x entries, where x is the value in the Show field.
• Click Previous to display the previous x entries, where x is the value in the Show field.
• Click Next to display the next x entries, where x is the value in the Show field.
• Click **Last** to display the last \( x \) entries, where \( x \) is the value in the **Show** field.

• To view additional log file entries without changing your current starting log file entry, leave the **Go To** field blank, change the value in the **Show** field, and click **Reload**.

• If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the **Log file status** field.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for more than four hours stale and at that point release ninety-five percent of any memory being held. If you try to use a stale viewer, OpenEdge Management and OpenEdge Explorer automatically reload the file. Because additional resource activity might have occurred during the viewer's inactivity, the reloaded log file view might not match the previous log file view of that resource.

• OpenEdge Management and OpenEdge Explorer consider a viewer that has been inactive for forty-eight hours dead. Once a viewer dies, OpenEdge Management and OpenEdge Explorer release all of its memory. To return to the log file displayed in a dead view, you must renavigate to it, even if you pinned up the view or saved a link to it before the viewer died.
Configuring OE Web Servers

You can work with OE Web Server instances in OpenEdge Management and OpenEdge Explorer, as described in the following sections:

For details, see the following topics:

- Configuring and managing OE Web Servers
- Managing OE Web Servers
- Working with REST Web Applications

Configuring and managing OE Web Servers

You can configure and manage OE Web Servers by:

- REST Management Agent on page 226
- OE Web Server on page 226
- REST Web Applications on page 226
- Apache Tomcat on page 226
- Installing a REST Management Agent into a Java AdminServer on page 226
- Managing REST Web Applications with the OE Web Server on page 227

Each of these definitions and tasks are described in the following sections.
REST Management Agent

A REST Management Agent is a component of Progress OpenEdge that you use to deploy, configure, and manage a Java AdminServer's OpenEdge REST Web Applications. The REST Management Agent is a Java Web application that acts as an intermediary for you to communicate with the deployed REST Web Applications. It helps you enable/disable the application, collect statistics, and provide a run-time configuration for the REST Web Applications.

OpenEdge ships with a default REST Management Agent (oerm.war).

OE Web Server

An OE Web Server is a remote Web Server application that manages the deployment and management of REST Web Applications.

OpenEdge provides three clients—OpenEdge Management, OpenEdge Explorer, and RESTMAN utility—that use an OE Web Server to manage REST Web Applications.

REST Web Applications

A REST Web Application is a Web application archive file (WAR file) that includes one or more REST services. A REST service is a service defined to expose your REST service annotated ABL applications to a Web Server as a REST Web application.

A REST Web application typically includes java classes, web pages (HTML), images, and other files that together make up a web application.

Apache Tomcat

The default Web Server shipped with OpenEdge is where you can deploy the REST Web Applications that you want to make available to clients. The Tomcat server is available in your OpenEdge installation directory, 

OpenEdge-install-dir/servers/tomcat.

Installing a REST Management Agent into a Java AdminServer

As part of the OpenEdge installation, REST Management Agent is installed in the OpenEdge installation directory, OpenEdge-install-dir/servlets/rest.

Note: You need not perform the following steps if you are using the Java AdminServer installed with the Apache Tomcat shipped with OpenEdge installation as Tomcat already has the OE Web Server installed.

To start working with OE Web Server and REST Web Applications, you must configure your Web Server to install the REST Management Agent in the Java AdminServer associated with the Web Server. To configure your Web Server:

- Copy the REST Management Agent, oerm.war, from the directory, OpenEdge-install-dir/servlets/rest, to the Web Server deployment directory.
- For example, if you are using a Tomcat Web Server, you copy oerm.war to the $CATALINA_HOME/webapps directory.
Managing OE Web Servers

You can manage OE Web Servers by:

- Starting and Stopping Tomcat server on page 228
- Creating an OE Web Server instance on page 229
Starting and Stopping Tomcat server

OpenEdge is shipped with a Tomcat server. The installed and configured Tomcat server provides a Java AdminServer and a web server for the REST Web application development and testing. The Tomcat server is available in your OpenEdge installation directory, $DLC/servers/tomcat.

Before you define/publish an OE Web Server or REST Service, you must start the Apache Tomcat server. You can start Apache Tomcat server installed with OpenEdge either using the Resources panel in the grid frame, or using Proenv.

1. To start the Tomcat server using the Resources panel:
   a) Click **Resources** in the management console menu bar. The resource types appear in the list frame.
   b) Expand the **Web Server** category, and click on the Tomcat instance. The **Details** page for that instance appears in the detail frame.
   c) In the **Command and control** section of the page, click **Control**. The Apache Tomcat installed with OpenEdge page opens.
   d) Click on the **Start Web Server** button to start Tomcat server.

   **Note:** The button is a Toggle button depending on the status of the Tomcat server instance. If the Web server is already started, the button displays the Stop option. To stop the Web server that is already running, click the **Stop Web Server** button.

2. To start the Tomcat server using Proenv:
   a) On the Windows Start > Programs menu, select Progress > OpenEdge > Proenv. Proenv opens a command prompt and configures it for your OpenEdge installation, displaying the configuration:

      proenv>

   b) Enter the following command:

      protc start

      A Tomcat shell window opens displaying the status of the server as running.
Note: To stop the Tomcat Server, enter `protc stop` at the `proenv>` command prompt.

Creating an OE Web Server instance

You can create new OE Web Server instances from the Resources tab.

Note: Make sure that the REST Management Agent is deployed before you create the OE Web Server instance. You must copy the oerm.war file to the deployment server.

To create a new OE Web Server instance:

1. From the management console bar, click the Resources tab's dropdown arrow. Choose New OpenEdge Resource > OE Web Server. The New OE Web Servername page appears.

2. Type the name of the new OE Web Server in the New OE Web Servername field.

Note: The OE Web Server name is case sensitive and can include any character except a period (.) or square brackets ([ ]). The name must be unique among all configured OE Web Server names.

3. Type the URL in the URL field. You can now configure the properties of the new instance. Initially, the instance is enabled only for administration, allowing you to modify its properties and deploy REST services to it without allowing premature user access to an incomplete configuration. Form the root URL according to the following format:

   http://hostmachine:port

Where:

- `hostmachine` — The name of the machine where the OEM is installed (Default: localhost).
- `port` — The HTTP port number on the machine (Default: 8089).

4. Under Deploy OE Web Server section, you can choose to deploy the OE Web Server instance on the Tomcat Manager (Apache Tomcat installed along with the Progress Developer Studio for OpenEdge).

Note: Ensure that you have Tomcat Manager API enabled and a Tomcat user setup to enable this functionality.

- **Deploy OE Web Server** — Select this check box to deploy the OE Web Server instance. Selecting this option requires you to enter the Tomcat Manager Details fields: Username, Password, and URL.
  - `username` — A username required to access the Tomcat Server.
  - `password` — The password for the specified username.
  - `URL` — The URL where you have the Tomcat Server installed. For example, if you are using the default ports in the Tomcat server, you can enter the URL as `http://localhost:8980/manager/text/deploy`. 
Note: You can choose to create an OE Web Server instance without deploying the OE Web Server.

5. Click Save. The new OE Web Server instance appears under the OE Web Server folder.

Configuring an OE Web Server instance

Before deploying REST services, you must configure an OE Web Server instance to host the services that you want to deploy or import. To configure an OE Web Server instance, you set properties for it. After the OE Web Server instance is configured, you can deploy a service and configure additional properties that affect execution of the service.

To configure an OE Web Server instance, you can view or edit properties in the following categories:

- Location (read)
- Proxy Server Setting (read and edit)
- Security (read and edit)

Setting OE Web Server instance properties

You can set location and proxy server setting for an OE Web Server instance.

To set OE Web Server instance properties:

1. Click Resources in the management console menu bar. The resource types appear in the list frame.
2. Expand the OE Web Server category, and click on an instance. The Details page for that instance appears in the detail frame.
3. In the Command and control section of the page, click Configuration. The OE Web Server Configuration page opens. From this page, you can:
   - View and display of the OE Web Server properties.
   - Click Edit to modify the properties. For details, see Editing an OE Web Server instance configuration on page 231.

OE Web Server Location properties

You set the properties, shown in the following table, in this category when you first create the OE Web Server instance. For more information see Creating an OE Web Server instance on page 229.

Table 108: OE Web Server Location properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>Indicates that the machine on which the OE Web Server is installed is remote.</td>
</tr>
<tr>
<td>URL</td>
<td>Specifies the URL for the OE Web Server. For example, <a href="http://localhost:8080/oerm">http://localhost:8080/oerm</a></td>
</tr>
</tbody>
</table>
OE Web Server Proxy Server Setting properties

The properties in this category, as shown in the following table, apply to both local and remote OE Web Server instances. They specify the information required to access any proxy server for the network on which the OE Web Server resides.

Table 109: OE Web Server Proxy Server Setting properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy host</td>
<td>The TCP host name for the proxy server</td>
</tr>
<tr>
<td></td>
<td>Installed Default: None</td>
</tr>
<tr>
<td>Proxy port</td>
<td>The TCP port number of the proxy service</td>
</tr>
<tr>
<td></td>
<td>Installed Default: 0</td>
</tr>
<tr>
<td>Proxy username</td>
<td>A valid username required to access the proxy server</td>
</tr>
<tr>
<td></td>
<td>Installed Default: None</td>
</tr>
<tr>
<td>Proxy password</td>
<td>The password for the specified username</td>
</tr>
<tr>
<td></td>
<td>Installed Default: None</td>
</tr>
</tbody>
</table>

OE Web Server Security properties

The properties in this category, as shown in the following table, apply to both local and remote OE Web Server instances. They specify the information required to authenticate the type of user who will login to OE Web Server.

Table 110: OE Web Server Security properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No user authentication</td>
<td>Access the OE Web Server without any user authentication (no username and password).</td>
</tr>
<tr>
<td>Use default authentication</td>
<td>Access the OE Web Server using the default username and password based on the Web Server authorization setting. This option is selected by default.</td>
</tr>
<tr>
<td>Login to Web Server with username and password</td>
<td>Access the OE Web Server using the new username and password provided by you. Selecting this option requires you to enter the username and password details in the Login page.</td>
</tr>
</tbody>
</table>

Editing an OE Web Server instance configuration

You can edit the configuration properties of an OE Web Server instance from the management console.
To edit OE Web Server properties:
Chapter 9: Configuring OE Web Servers

1. Click Resources in the management console menu bar. The resource types appear in the list frame.

2. Expand the OE Web Server category, and click on an instance. The Details page for that instance appears in the detail frame.

3. In the Command and control section of the page, click Configuration. The OE Web Server Configuration page opens.

4. Click Edit, and make the changes you want to the properties.

5. Click Save.

Setting OE Web Server application defaults

You can view and set the default values used to initialize properties for REST Web Applications that you deploy. Any changes to these defaults affect only REST Web Applications that you deploy after confirming the change. To set the values of properties for deployed REST Web Applications, set the property values individually for each REST Web application.

To view and set OE Web Server application defaults:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance whose default values you want to view.

3. Click the OE Web Server instance. The OE Web Server details page appears.

4. In the Command and control section of the page, click Defaults. The OE Web Servers Defaults page appears.

5. Click Edit.

6. Select the Runtime property whose default value you want to change. For details on the available properties, see OpenEdge Application Server: Administration.

7. Type the new value in the available field. For logical values, typing any value other than true results in a setting of false.

8. Click Submit to change the default value.

To return to the default values, you can click Reset from the Defaults page.

Deploying a REST Web application (.war file)

You can deploy a REST Web application (.war file) to the Servlet AdminServer where the OpenEdge Manager Servlet instance is installed.

A deployed REST Web application receives its initial property values from the default.props file for the REST instance. You can change these default property values after you deploy the .war file. A REST Web application always deploys with a disabled status to prevent premature or unintended client access.

To deploy a REST Web application:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance where you want to deploy the REST Web application.

3. Click the OE Web Server instance. The REST Applications page appears.

4. In the REST Applications section of the page, click the Deploy button.
5. In the Deploy REST Application .war dialog that comes up, browse for the path and name of the WAR (.war) file for the REST Web application that you want to deploy.

6. You can optionally provide the REST Web application name in the Application name field.

7. Click **Deploy** to deploy the REST Web application with the default settings. A confirmation message appears, and the WAR file name is identified in the **REST Applications** section of the REST Applications page.

Deployment generates files in the OE Web Server instance directory, one of which is `applicationName.props`, an XML file containing the current REST Web application property settings (initially set from `default.props`).

**Caution:** You must disable the REST Web application before you try to modify any of the properties. Two of the properties, `serviceLogEntryLevel` and `serviceLoggingLevel` can be modified even when the application is **Enabled**.

After deployment, to make the same REST Web application available using different information, deploy a new REST Web application with the new information, using a different REST Web application name in the same OE Web Server context. You could also deploy the REST Web application to a different OE Web Server context. However, you can change the run-time properties of a deployed REST Web application at any time.

**Viewing the status of an OE Web Server instance**

You can view the status of a selected OE Web Server instance to see if the instance is running. If the instance is running, the status indicates that it is enabled.

To view the status of an OE Web Server instance:

1. Click **Resources** in the management console menu bar. The resource types appear in the list frame.

2. Expand the **OE Web Server** category, and click on an instance. The **Details** page for that instance appears in the detail frame.

3. Under **Operations views**, click **Status**. The following status information appears:
   
   • Whether the OE Web Server instance is running
   • Whether access to administrative functions is enabled

**Note:** You can also view the status of the OE Web Server instance on the top right corner of the **Details** page.

**Viewing OE Web Server instance statistics**

You can view and reset run-time statistics for a selected OE Web Server instance. Each OE Web Server instance creates and begins accumulating these statistics, Start Time, when it is first started by the Java Servlet Engine and Creation Time. You can also reset the statistics at any point during OE Web Server execution, so that the statistics begin accumulating again from a new Start Time.

To view the statistics of an OE Web Server instance:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance whose statistics you want to view.
3. Click the OE Web Server instance. The OE Web Server details page appears.

4. Under Operational Views, click Statistics. The following information appears in the Statistics page:
   - **Active Requests** — Requests currently queued and being acted upon by the OE Web Server
   - **Authentication Errors** — Total number of authentication errors
   - **HTTP Request Errors** — Requests received from the HTTP listener, including administrative, and OE Web Server requests
   - **HTTP Requests** — Total requests received from the HTTP listener, including administrative, and REST requests
   - **Method not allowed errors** — Total errors returned by the OE Web Server, with error counts broken out at the bottom of the list for each of several error categories when total errors are greater than zero (0)
   - **REST Disabled errors** — calls while OE Web Server is disabled
   - **Services disabled** — REST Web Applications deployed to this OE Web Server that are disabled from client access
   - **URL not found errors** — calls made to unidentified resources
   - **Creation Time** — Date and time that counting began for the statistics
   - **Start Time** — Start date and time of the OE Web Server instance

5. To reset the statistics and start accumulating all statistics from zero, click Reset if you want to reset the statistics.

**Changing OE Web Server run-time properties**

You can temporarily change some OE Web Server properties at run time without restarting your Java Servlet Engine (JSE). This is most useful for testing and debugging. The next time you restart your JSE, these settings revert to the current configuration settings for these properties stored in the properties file for the OE Web Server. The OE Web Server property file is stored in the WEB-INF directory of the deployed OE Web Server.

To change OE Web Server run-time properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the OE Web Server instance whose run-time properties you want to change.
3. Click the OE Web Server instance. The OE Web Server details page appears.
5. Click Edit to modify any of the following properties:
   - **adminEnabled** — Specifies whether administrator privileges are in effect, either enabled or disabled.
   - **description** — Specifies the description of the OE Web Server instance.
   - **logAppend** — Specifies whether to start a new log file each time the server is restarted, either true or false.
   - **logEntryTypes** — Specifies a list of logging entry types separated by commas.
   - **logFile** — Specifies the location and name of the OE Web Server log file.
   - **loggingLevel** — Specifies the amount of information to be written to the broker log.
   - **managerPropertiesURI** — Specifies the identifier for the OE Web Server resource.
   - **name** — Specifies the name of the OE Web Server.
   - **propertiesFileName** — Specifies the path and name of the oerm.properties file.
- **webAppEnabled** — Controls the OE Web Server's ability to accept and process requests to any of its deployed OE Web Server applications.
- **workDir** — Specifies the path of the OpenEdge working directory.

6. Click **Submit**.

### Deleting an OE Web Server instance

You can delete any OE Web Server instance listed in the management console as long as it has no REST services deployed to it.

To delete an OE Web Server instance:

1. Click **Resources** in the management console menu bar. The resource types appear in the list frame.
2. Filter, or search for, the OE Web Server instance you want to delete.
3. Click the OE Web Server instance. The **OE Web Server** details page appears.
4. Click **Delete**, and click **OK** to confirm the deletion.

The **OpenEdge Management Resources** page appears. A set of next-step options that are related to resources are available for your use.

**Note:** You can choose to stop the OE Web Server before deleting it. However, all the running applications get undeployed when you delete an OE Web Server.

**Note:** Deleting the OE Web Server instance does not undeploy it from the application server. To undeploy the OE Web Server instance from the application server, you must uninstall the OE Web Server instance from the application server after removing the instance from OEM.

### Enabling or Disabling an OE Web Server instance

From the management console, you can enable or disable monitoring of an OE Web Server instance.

To enable or disable monitoring of an OE Web Server instance:

1. Click **Resources** in the management console menu bar. The resource types appear in the list frame.
2. Expand the **OE Web Server** category, and click on an instance. The **Details** page for that instance appears in the detail frame.
3. Click **Control**. That instance's OE Web Server page appears.
4. Click **Edit**.
5. Select the **Enabled** check box to enable the OE Web Server monitoring or clear the check box to disable it.

### Viewing the OE Web Server log file

You can view the log file for an OE Web Server by using the log file viewer. The log file viewer allows you to examine the log file through an HTML interface.
To access the log file viewer:

1. Click **Resources** in the management console menu bar. The resource types appear in the list frame.

2. Expand the **OE Web Server** category, and click on an instance. The **Details** page for that instance appears in the detail frame.

3. Click **Log File Viewer**. The **OE Web Server Log File Contents** page opens. You can use the Log File Viewer in the following ways:

   - Use the **Show** field to control how many database log file entries display at one time. The number entered into the **Show** field cannot be less than 10.
   - Use the **Overlap** field to control how many entries are repeated from screen to screen.

**Note:** The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

   - Choose **Descending** to change the default display that is in ascending order. Note that the **Show** field dictates the number of entries shown, regardless of whether they display in ascending or descending order.

   - Click **Reload** after you have changed any values for the number of lines or lines per page. If you do not reload, the viewer continues to display the previous values.

   - Click **Go To** to control the numbered entry in the log file from where the viewer begins its display. For example, a value of 10 entered into the **Go To** field will begin the display from the tenth log file entry.

   - Click **First** to display the first x entries, where x is the value in the **Show** field.

   - Click **Prior** to display the previous x entries, where x is the value in the **Show** field.

   - Click **Next** to display the next x entries, where x is the value in the **Show** field.

   - Click **Last** to display the last x entries, where x is the value in the **Show** field. If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in the **Log file status** field.

---

**Working with REST Web Applications**

You can perform the following REST Web application administration tasks from the management console:

- **Enabling or disabling a REST Web application for client access** on page 236
- **Viewing REST Web application statistics** on page 237
- **Using the REST Web application log file viewer** on page 238
- **Updating a deployed REST Web application** on page 239
- **SSL support for REST Web Applications** on page 240
- **Undeploying a REST Web application** on page 240

**Enabling or disabling a REST Web application for client access**

You can enable any deployed REST Web application, making it accessible or inaccessible to user requests on the network. A deployed REST Web application is disabled by default.
To enable or disable a REST Web application for client access:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance where the REST Web application you want to enable or disable is deployed.

3. Click the OE Web Server instance. The **OE Web Server** Details page appears.

4. Click the REST Applications link. The REST Applications page appears.

5. Under the **REST Applications** section, select the deployed REST Web application.

6. Click **Enable** in the Actions column to enable the application. The **Enabled column** displays the status of the **Enabled** option.

7. To disable the enabled application, click **Disable** in the **Actions** column.

**Viewing REST Web application statistics**

You can view (and reset) run-time statistics for a selected REST Web application. Each REST Web application creates and begins accumulating these statistics (Start Time) from the time that its parent OE Web Server instance starts up (Creation Time). You can reset the REST Web application statistics at any point while the OE Web Server instance is running. A reset clears and begins accumulating the statistics again with a new Start Time.

To view the run-time statistics for a REST Web application:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance where the REST Web application whose runtime statistics you want to view is deployed.

3. Click the OE Web Server instance. The **OE Web Server** details page appears.

4. Click the REST Applications link. The REST Applications page appears.

5. Under the **REST Applications** section, select the deployed REST Web application.

6. Under **REST Application Details section**, click the Statistics tab to view a read-only statistics presentation with the following information:

   - **Adapter errors** — Total number of OE Web Server failures
   - **Application errors** — Total number of errors related to REST Web application
   - **Average response time** — Average response time of each request response.
   - **Failed admin logins** — Total number of failed administrator logins
   - **Failed admin requests** — Total number of administrator requests that failed due to missing request information
   - **Failed app server connections** — Total number of failed AppServer connections in the connection pool
   - **Failed user logins** — Total number of failed user logins
   - **Failed user requests** — Total number of requests that failed due to missing request information
   - **Max response time** — Maximum response time of each request response
   - **Successful admin logins** — Total number of successful administrator logins
   - **Successful admin requests** — Total number of successful administrator requests
• **Successful app server connections** — Total number of successful AppServer connections in the connection pool for this REST Web application
• **Successful user logins** — Total number of successful user logins
• **Successful user requests** — Total number of successful user requests handled by the selected REST Web application
• **Creation time** — Date and time that counting began for the statistics
• **Start time** — Start date and time of the REST Web application

7. To reset the statistics and start accumulating from zero, click **Reset** if you want to reset the statistics.

### Using the REST Web application log file viewer

To view the contents of a REST Web application log file, access the viewer associated with each individual log file.

The log file viewer allows you to examine the contents of a REST Web application log file in HTML format. You can access a log file viewer from the following location:

- Click the **Log File Viewer** button in the **REST Applications** section of the **REST Applications** page. The log file viewer button appears with every deployed application.

The following information will help you use the REST Web application log file viewer:

- Use the **Show** field to control how many OE Web Server log file entries appear at one time. The number entered into the **Show** field cannot be less than 10.
- Use the **Overlap** field to control how many entries are repeated from screen to screen.

**Note:** The value in the **Overlap** field must not be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or fewer of them.

- Click **Reload** after changing the values in either the **Show** field or **Overlap** field. OpenEdge Management prompts you to click **Reload**. A warning message that reads **changed, reload needed** appears in the **File log status** field in the **log file summary** section of the page.

If you do not reload, the viewer displays the previous values.

- Click **Go To** to control which numbered entry in the log file the viewer begins its display with. For example, a value of 10 entered into the **Go To** field begins the display from the tenth log file entry.

**Note:** You must click **Go To** after entering a value in the **Go To** field, or the viewer will not update its display.

- The default display of entries is in ascending order. Choose **Descending** to change the display. Note that the **Show** field dictates the number of entries shown, whether they appear in ascending or descending order.
- Click **First** to display the first \( x \) entries, where \( x \) is the value in the **Show** field.
- Click **Prior** to display the previous \( x \) entries, where \( x \) is the value in the **Show** field.
- Click **Next** to display the next \( x \) entries, where \( x \) is the value in the **Show** field.
- Click **Last** to display the last \( x \) entries, where \( x \) is the value in the **Show** field.
- To view additional log file entries without changing your current starting log file entry, leave the **Go To** field blank, change the value in the **Show** field, and click **Reload**.
Updating a deployed REST Web application

You can update the definition and deployment information for a deployed OE Web Server at any time.

Caution: You can freely update the definition and deployment information for a deployed REST Web application during development. However, once you deploy and enable a REST Web application for client access, avoid making any changes to this information, as client implementations depend on its stability.

After deployment, to make the same REST Web application available using different information, deploy a new REST Web application with the new information, using a different REST Web application name in the same OE Web Server context. You can also deploy the REST Web application to a different OE Web Server context.

Setting OE Web Server application properties

You can view and set the value of each property for a deployed REST Web application, or you can reset the values of all properties to the current REST Web application defaults (stored in the default.props file for the OE Web Server instance where the REST Web application is deployed).

You must disable a REST Web application before you can set most properties for it. You can verify if the REST Web application is disabled by checking its status. The only properties that you can set for an enabled OE Web Server are:

- serviceLogEntryLevel
- serviceLoggingLevel

If you set other properties while the REST Web application is enabled, the property value changes take effect only after you disable and then enable the REST Web application again.

Note: Unlike OE Web Server instance properties, the values for REST Web application properties are not stored in the ubroker.properties file. They are stored in the applicationname.props XML file located in the Web application Servlet directory for the OE Web Server instance where the REST Web application is deployed.

To view and set REST Web application properties:

1. Click Resources in the management console menu. All resources managed by your console appear in the grid frame.
2. Filter or search for the OE Web Server instance where the REST Web application whose properties you want to view and set is deployed.
3. Click the OE Web Server instance. The OE Web Server details page appears.
4. Click the REST Applications link. The REST Applications page appears.
5. Under the REST Applications section, select the deployed REST Web application.
6. Under REST Application Details section, select the Properties tab to view the application properties of the deployed REST Web application.

For details on the properties that can appear in this list, see OpenEdge Application Server: Administration.

Note: The actual list of properties displayed is a subset of all the properties, and changes depending on whether the session model of the OE Web Server is session managed or session free. To verify the session model of an OE Web Server, check its status. For more information, see Enabling or disabling a REST Web application for client access on page 236.
7. Type the new value in the available field. For logical values, entering any value other than true results in a setting of false.

8. Click **Save** when you finish. The changes are then written to the `applicationname.props` file for the selected REST Web application.

### SSL support for REST Web Applications

To have a secure connection between the REST Web application and the AppServer, the following conditions must be met:

- You must obtain and install public key certificates for the host machine in which the OE Web Server resides.

  **Note:** You can add any additional public certificates to `psccerts.jar` (located in WEB-INF/lib folder and contains all the certificates for the REST Web application required to connect to the OpenEdge AppServer) using the `certutil.bat`. If you do not want to use `psccerts.jar`, you can uncomment the property `certlocation` in the REST Web application's `web.xml` file.

- The REST Web application must send SSL requests to the AppServer to process the client requests. To configure the service to send SSL requests, set the value of the **ApplicationService Protocol** property to `AppServerS` or `AppServerDCS`. For more information on setting properties for REST Web Applications, see Setting OE Web Server application properties on page 239.

- The AppServer must be SSL-enabled, meaning that it accepts SSL requests from the OE Web Server (or other clients). OpenEdge supports Secure Socket Layer (SSL)-enabled communications between the Web server where the REST Management Agent runs and a Web server client. To achieve this, you must enable the REST Web application for HTTPS (SSL) connections. For more information on enabling SSL, see Viewing or modifying AppServer broker properties.

  For details on SSL support in OpenEdge, including configuring and operating a client of an SSL-enabled AppServer, see *OpenEdge Getting Started: Core Business Services - Security and Auditing*.

### Undeploying a REST Web application

You can undeploy a deployed REST Web application, which removes its deployment information, its properties, and other settings from the OE Web Server instance context.

You must disable a REST Web application before you can undeploy it. You can verify if the REST Web application is disabled by checking its status.

To undeploy a REST Web application:

1. Click **Resources** in the management console menu. All resources managed by your console appear in the grid frame.

2. Filter or search for the OE Web Server instance where the REST Web application you want to enable or disable is deployed.

3. Click the OE Web Server instance. The **OE Web Server** details page appears.

4. Click the REST Applications link. The REST Applications details page appears.

5. Under the **REST Applications** section, select the deployed REST Web application.

6. Click **Undeploy** in the OE Web Server's column to undeploy the application.
The application is undeployed. If the REST Web application is already disabled, it is now undeployed from the OE Web Server instance context and removed from the management console. If the REST Web application is still enabled, an error appears indicating that you must disable the REST Web application before undeploying it.

**Note:** You can undeploy multiple REST Web Applications at the same time, by selecting the checkbox against the application name and clicking the Undeploy button on the top left corner of the REST Application section.
Configuring Batch Programs

This section describes how to configure and manage Batch Program resources using OpenEdge Management.

For details, see the following topics:

- Overview
- Creating a Batch Program
- Modifying Batch Program configuration
- Starting a Batch Program
- Stopping a Batch Program
- Deleting a Batch Program
- Viewing and managing Batch Program processes
- Viewing Batch Program log files
- Batch Program properties
- API references

Overview

A Batch Program is an OpenEdge resource designed to run for longer periods of time and supports separate scripts to start, monitor, and stop the Program. You can manage and monitor Batch Program resources using OpenEdge Management.
Each Batch Program resource can have several copies of the program that is running. OpenEdge Management considers each program as an individual resource and allows you to start, stop, and monitor it. OpenEdge Management monitors the Batch Program instances by defining rules for each instance and generating alerts based on the status of the program.

Like any other resource, OpenEdge Management allows you to create, configure, and monitor Batch Program resources. When configuring a Batch Program resource, you can write scripts to monitor the processes supported by OpenEdge Management. Also, you can write your own scripts to monitor the processes that are NOT supported by OpenEdge Management. Once configured, they appear in the OpenEdge Management Resource Grid page.

Note: Creating, managing, and monitoring of Batch Program resources is not supported in OpenEdge Explorer.

Creating a Batch Program

You can create a Batch Program resource using the Create Batch Program page in the OpenEdge Management.

To create a Batch Program:

1. From the OpenEdge Management console menu, click Resources > New > Batch Program.
   
   The Create Batch Program page appears.

2. Specify the configuration properties as described in Batch Program properties on page 248.

3. Click CREATE.
   
   The Batch Program: <instance name> page appears with Command and Control and Processes sections.

Modifying Batch Program configuration

You can modify the configuration of a Batch Program resource using the Edit Batch Program in the OpenEdge Management.

To modify the configuration of a Batch Program resource:

1. From the OpenEdge Management console menu, click Resources > Go to Resources.
   
   All the resources managed by your console appear in a grid frame.

2. Filter or search for, and select the required Batch Program resource.
   
   The Batch Program: <instance name> page appears.

3. In the Command and Control section, click Configuration.
   
   The Edit Batch Program page appears.

4. Click EDIT and then modify the Batch Program resource properties as described in Batch Program properties on page 248.

5. Click SAVE
Starting a Batch Program

You can start a Batch Program resource using the Batch Program: <instance name> in the OpenEdge Management.

To start a Batch Program resource:

1. From the OpenEdge Management console men, click Resources > Go to Resources.
   All the resources managed by your console appear in a grid frame.
2. Filter or search for, and select the required Batch Program resource.
   The Batch Program: <instance name> page appears.
3. Click START PROGRAM.
   The Batch Program process is started and appears in the Processes section.
   The START PROGRAM option is displayed only when:
   • The Startup script is specified as described in Batch Program Startup properties on page 253.
   • The Batch Program is not running and supports only one process.
   • The Batch Program is running already but supports multiple processes.
4. To run multiple processes, click START PROGRAM again.

Stopping a Batch Program

You can stop a Batch Program resource using the Batch Program: <instance name> in the OpenEdge Management.

To stop a Batch Program resource:

1. From the OpenEdge Management console men, click Resources > Go to Resources.
   All the resources managed by your console appear in a grid frame.
2. Filter or search for, and select the required Batch Program resource.
   The Batch Program: <instance name> page appears.
3. Click STOP BATCH PROGRAMS.
4. Click YES in the confirmation dialog box to stop all the copies of the Batch Program resource.
   The Batch Program Control page appears.
   This page displays the number of processes running in the Program summary section. When the processes are stopped, the Process Count is displayed as 0.
   Optionally, you can click START BATCH PROGRAM to start the Batch Program resource again.
5. To stop an individual process, see Viewing and managing Batch Program processes on page 246.
Deleting a Batch Program

You can delete a Batch Program resource using the Batch Program: <instance name> in the OpenEdge Management.

To delete a Batch Program resource:

1. From the OpenEdge Management console men, click Resources > Go to Resources.
   All the resources managed by your console appear in a grid frame.

2. Filter or search for, and select the required Batch Program resource.
   The Batch Program: <instance name> page appears.

3. Click DELETE.

   Note: Ensure that you stop all the Batch Program processes before deleting the Batch Program. Clicking DELETE does not stop the Batch Program processes that are running.

4. Click YES in the confirmation dialog box.

Viewing and managing Batch Program processes

You can view and manage multiple processes that are starting, running, or stopped for a Batch Program resource using the Batch Program: <instance name> page in the OpenEdge Management. This enables you to monitor the status and other properties of multiple Batch Program processes simultaneously, and if required, stop or delete a process.

To view the available processes for a Batch Program resource:

1. From the OpenEdge Management console men, click Resources > Go to Resources.
   All the resources managed by your console appear in a grid frame.

2. Filter or search for, and select the required Batch Program resource.
   The Batch Program: <instance name> page appears.

3. In the Processes section, you can view the following information for each Process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>The Process ID (PID) of the process.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the process</td>
</tr>
<tr>
<td>State</td>
<td>The status of the process – Starting, Running, or Not Running.</td>
</tr>
<tr>
<td>Message</td>
<td>Provides additional information such as Process timed out or PID not alive.</td>
</tr>
<tr>
<td>Start Time</td>
<td>The time when the process is started.</td>
</tr>
</tbody>
</table>
### Viewing Batch Program log files

You can view the log files—stdout and stderr—of a Batch Program resource using the log file viewer in the OpenEdge Management. This allows you to examine the contents of the Batch Program resource log files through an HTML interface.

To view the Batch Program resource log files:

1. From the OpenEdge Management console men, click **Resources > Go to Resources**.
   - All the resources managed by your console appear in a grid frame.
2. Filter or search for, and select the required Batch Program resource.
   - The **Batch Program: <instance name>** page appears.
3. In the **Command and Control** section, you can choose to click one of the following to view the log file viewer:
   - **Log File Viewer (stdout)** — Displays the standard out log file entries.
   - **Log File Viewer (stderr)** — Displays the standard err log file entries.
4. In the **Log File Viewer**, you can:
   - Click **FIRST** to display the first x entries, where x is the value in the **Show** field.
   - Click **PRIOR** to display the previous x entries, where x is the value in the **Show** field.
Chapter 10: Configuring Batch Programs

- Click **NEXT** to display the next \( x \) entries, where \( x \) is the value in the **Show** field.
- Click **LAST** to display the last \( x \) entries, where \( x \) is the value in the **Show** field.
- Click **RELOAD** to reload the values, if changed in either the **Show** field or the **Overlap** field.
- Use the **Go to** field to type the number of the log file entry you want the log file viewer to begin with. For example, if you type a value of 10 in the **Go to** field and press **Enter**, the log file viewer displays from the tenth log file entry.
- Use the **Show** field to type the number of log file entries you want the log file viewer to display at one time. The value in the **Show** field cannot be less than 10.
- Use the **Overlap** field to type the number of log file entries you want to be repeated from screen to screen. The value in the **Overlap** field cannot be more than the number in the **Show** field minus one. For example, if you show 30 entries, you can overlap only 29 or lesser.
- Use the **Display** drop-down to choose the order in which you want the log file entries to be displayed, **Ascending** or **Descending**. The default is **Ascending**.
- If the contents of the log file have changed since you opened the viewer, the log file viewer indicates this in its footer’s **Status** field.

**Batch Program properties**

When creating or configuring a Batch Program resource, you can specify or modify the Batch Program properties as described in the following sections:

- **Batch Program General properties** on page 252
- **Batch Program Startup properties** on page 253
- **Batch Program Monitoring properties** on page 256
- **Batch Program Shutdown properties** on page 259
- **Batch Program Environment variable properties** on page 260
Table 111: Startup properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup script</td>
<td>(Optional) The name of the Batch Program that you want to run. This can be the Batch Program or a script that starts the Batch Program.</td>
</tr>
<tr>
<td>Startup script parameters</td>
<td>(Optional) The parameters that are to be passed to the Startup script.</td>
</tr>
<tr>
<td>Startup script timeout (seconds)</td>
<td>(Optional) The timeout value for the Batch Program to start. The default value is 15 seconds. If the status of the Batch Program is unknown within the timeout value, OpenEdge Management considers it as a failure.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Startup script, if running, after the specified timeout. If this option is not selected and the Startup script is not terminated after the specified timeout, OpenEdge Management tracks the PID of the script.</td>
</tr>
<tr>
<td>Supports multiple processes</td>
<td>(Optional) Supports running multiple process.</td>
</tr>
<tr>
<td>Startup failure behavior</td>
<td>(Optional) The behavior of OpenEdge Management when the startup of the Batch Program fails on the first, second, and subsequent startups.</td>
</tr>
<tr>
<td>• On first startup failure</td>
<td>Select one or more behaviors from the following options:</td>
</tr>
<tr>
<td>• On second startup failure</td>
<td>• <strong>Fire alert</strong> — Fires an alert.</td>
</tr>
<tr>
<td>• On subsequent startup</td>
<td>• <strong>Perform action</strong> — Executes an action.</td>
</tr>
<tr>
<td>failure(s)</td>
<td>• <strong>Restart the program</strong> — Restarts the Batch Program.</td>
</tr>
<tr>
<td>Subsequent startup limit</td>
<td>(Optional) The number of times the Subsequent startup behavior must be executed.</td>
</tr>
<tr>
<td></td>
<td>The minimum value is 1 and there is no maximum value. The program restarts until it reaches the specified limit.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Monitoring method</td>
<td>The method to track the status of the program.</td>
</tr>
<tr>
<td></td>
<td>Select from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Status script</strong> — Tracks the status of the Batch Program. The status can be running or failed. The script can call other programs such as reading log files, reading PID files, or making an HTTP request.</td>
</tr>
<tr>
<td></td>
<td>• <strong>PID file</strong> — The Process ID (PID) of the Batch Program. When you define the path for the PID file, OpenEdge Management looks for the PID file and considers that the Batch Program is running only if the file is valid and the process associated with PID is executing.</td>
</tr>
<tr>
<td></td>
<td>• <strong>PID registration</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>Exit code</strong></td>
</tr>
<tr>
<td>Status script</td>
<td>(Optional) The path to the Status script. OpenEdge Management utilizes this script to determine the status of the Batch Program.</td>
</tr>
<tr>
<td>Status script parameters</td>
<td>(Optional) The parameters that are to be passed to the Status script.</td>
</tr>
<tr>
<td>Status script timeout (seconds)</td>
<td>(Optional) The timeout value for the OpenEdge Management to wait for tracking the Batch Program's status.</td>
</tr>
<tr>
<td></td>
<td>The default value is 15 seconds. The Status script must execute and terminate within the timeout value. If the Status script terminates with an exit code 0, OpenEdge Management considers that the status of the Batch Program as running. If the script terminates with an exit code other than 0, OpenEdge Management considers that the status as failed. If the script is missing or cannot be executed, OpenEdge Management considers that the Program is not running and failed.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Status script, if running, after the specified timeout. If this option is not selected and the Status script is not terminated after the specified timeout, OpenEdge Management tracks the PID of the script.</td>
</tr>
<tr>
<td>PID file</td>
<td>The absolute path to the location where PID is maintained by Startup or Shutdown scripts.</td>
</tr>
<tr>
<td>Status failure behavior</td>
<td>The behavior of OpenEdge Management when the status of the Batch Program is unknown.</td>
</tr>
<tr>
<td></td>
<td>Select one or more behaviors from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Fire alert</strong> — Fires an alert.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Perform action</strong> — Executes an action.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Restart the program</strong> — Restarts the Batch Program.</td>
</tr>
</tbody>
</table>
Table 113: Shutdown properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shutdown method</td>
<td>The method to stop the Batch Program. Select from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Shutdown script</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>Terminate</strong></td>
</tr>
<tr>
<td></td>
<td>If you select <strong>Terminate</strong>, OpenEdge Management looks for the Process ID (PID) of the Batch Program and stops the Batch Program. If the PID is not available, OpenEdge Management fails to stop the Batch Program.</td>
</tr>
<tr>
<td>Shutdown script</td>
<td>(Optional) The path to the script to stop the Batch Program.</td>
</tr>
<tr>
<td>Shutdown script parameters</td>
<td>(Optional) The parameters that are to be passed to the Shutdown script.</td>
</tr>
<tr>
<td>Shutdown script timeout (seconds)</td>
<td>(Optional) The timeout value for the OpenEdge Management to wait for executing the Shutdown script to stop the Batch Program. The default value is 15 seconds. If the Shutdown script is not executed within the timeout value, OpenEdge Management considers it as a failure.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Shutdown script, if running, after the specified timeout. If this option is not selected and the Shutdown script is not terminated after the specified timeout, OpenEdge Management tracks the PID of the script.</td>
</tr>
</tbody>
</table>

Table 114: Environment variable properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>(Optional) The name or ID of the user account on the machine where the process is to be executed. This field allows you to determine the permissions that the process must have when executed.</td>
</tr>
<tr>
<td>Group</td>
<td>(Optional) (UNIX only) The name of the group.</td>
</tr>
<tr>
<td>Password</td>
<td>(Optional) The user account's password for authentication. It is stored in the configuration database.</td>
</tr>
<tr>
<td>Environment variables</td>
<td>(Optional) The environment variables specified by the user. These variables are provided during the execution of the Startup, Status, and Shutdown scripts.</td>
</tr>
<tr>
<td>Debug program</td>
<td>Debugs the Batch Program.</td>
</tr>
</tbody>
</table>
# Batch Program General properties

Table 115: General properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique name for the Batch Program.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The name must not contain spaces or special characters such as an asterisk (*), an ampersand (&amp;), or a period.</td>
</tr>
<tr>
<td>AdminServer</td>
<td>The AdminServer on which the Batch Program is to be executed. The default is the <code>localhost</code>.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Running a Batch Program resource on a remote AdminServer is not supported in OpenEdge 11.7.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) The description for the Batch Program.</td>
</tr>
<tr>
<td>Enabled</td>
<td>(Optional) Enables the Batch Program for monitoring.</td>
</tr>
<tr>
<td></td>
<td>This check box is not selected by default for testing purposes. Selecting this check box enables OpenEdge Management to perform status checks,</td>
</tr>
<tr>
<td></td>
<td>trigger alerts, and execute actions based on the status of the Batch Program.</td>
</tr>
<tr>
<td>Working directory</td>
<td>(Optional) The working directory of the Batch Program.</td>
</tr>
<tr>
<td></td>
<td>Specify a valid pathname. If not specified, the default is the OpenEdge Work directory.</td>
</tr>
<tr>
<td>Output file (stdout)</td>
<td>(Optional) The path where the standard out file from the Startup, Status, and Shutdown scripts is directed to.</td>
</tr>
<tr>
<td></td>
<td>If not specified, the output from the Program will be written to the AdminServer log file.</td>
</tr>
<tr>
<td>Output file (stderr)</td>
<td>(Optional) The path where the standard err file from the Startup, Status, and Shutdown scripts is directed to.</td>
</tr>
<tr>
<td></td>
<td>If not specified, the output from the Program will be written to the AdminServer log file.</td>
</tr>
</tbody>
</table>
Batch Program Startup properties
### Table 116: Startup properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup script</td>
<td>(Optional) The name of the Batch Program that you want to run. This can be the Batch Program or a script that starts the Batch Program.</td>
</tr>
<tr>
<td>Startup script parameters</td>
<td>(Optional) The parameters that are to be passed to the Startup script.</td>
</tr>
<tr>
<td>Startup script timeout (seconds)</td>
<td>(Optional) The timeout value for the Batch Program to start. The default value is 15 seconds. If the Startup script terminates within the specified timeout with a non-zero Exit code, OpenEdge Management considers it as a failure. If the Startup script is not terminated within the specified timeout, OpenEdge Management considers it as running.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Startup script, if running, after the specified timeout. If this option is not selected and the Startup script is not terminated after the specified timeout, OpenEdge Management tracks the PID of the script.</td>
</tr>
<tr>
<td>Supports multiple processes</td>
<td>(Optional) Supports running multiple process. If this option is not selected, you cannot start the multiple processes of the Batch Program neither from the OpenEdge Management console nor the REST API.</td>
</tr>
<tr>
<td>Startup failure behavior</td>
<td>(Optional) The behavior of OpenEdge Management when the startup of the Batch Program fails on the first, second, and subsequent startups. Select one or more behaviors from the following options:</td>
</tr>
<tr>
<td>• On first startup failure</td>
<td>Fire alert — Fires an alert.</td>
</tr>
<tr>
<td>• On second startup failure</td>
<td>Perform action — Executes an action.</td>
</tr>
<tr>
<td>• On subsequent startup failure(s)</td>
<td>Restart the program — Restarts the Batch Program. To enable the behavior options on the next failure, select the Restart the Program option. The default is that the OpenEdge Management does not follow any behavior.</td>
</tr>
<tr>
<td>Subsequent startup limit</td>
<td>(Optional) ) The maximum number of times the Batch Program can be started before the OpenEdge Management considers the Startup as a failure. The program restarts until it reaches the specified limit. The minimum value is 1 and the maximum value is 300. In a case where the Startup program continuously fails, and Restart the Program is selected to avoid busy loops, there will be a delay of few seconds between each attempt to restart.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Exit code   | Indicates the status of the process when it exited. If the Startup script does not terminate within the specified timeout value, OpenEdge Management monitors the script until it exits and uses the exit code of the process to determine the action to perform.  

**Note:** For OpenEdge Management to monitor the exit code, the process should be started and should NOT be restarted before the process terminates.  
In a case where OpenEdge Management is restarted or the exit code is unknown, OpenEdge Management performs the default exit code action. |
| Action      | The action that you want to associate with the failure of the Batch Program. For more information, see [Using Batch Program default actions](#) on page 260.  
Once you specify the exit code and select its associated action, click **ADD/UPDATE** to add them. You can choose to add multiple exit codes and their associated actions. Click **DELETE** to delete your selection. |
Batch Program Monitoring properties
Table 117: Monitoring properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring method</td>
<td>The method to track the status of the program. Select from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Status script</strong> — Tracks the status of the Batch Program. The script can perform any action to determine the Batch Program's return state including reading log files, reading PID files, or making an HTTP request. This script can be Windows batch programs or executables, or UNIX shell scripts or executables.</td>
</tr>
<tr>
<td></td>
<td>• <strong>PID file</strong> — The file that contains a Process ID (PID) of the Batch Program. If available, OpenEdge Management uses this file to determine the state of the process. When you define the path for the PID file, OpenEdge Management looks for the PID file and considers that the Batch Program is running only if the file is valid and the process associated with PID is executing. In a case where the PID file is not available, cannot be read, or does not contain a single line with an integer value, the status is considered as failed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>PID registration</strong> — Exposes a REST API with which a stand-alone process can register a PID with OpenEdge Management. OpenEdge Management monitors the PID periodically to ensure that it is running. For the process to be monitored, it must be running on the same machine where the AdminServer is configured for the Batch Program.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Exit code</strong> — Indicates the status of the process when it exited.</td>
</tr>
<tr>
<td>Status script</td>
<td>(Optional) The path to the Status script. OpenEdge Management utilizes this script to determine the status of the Batch Program.</td>
</tr>
<tr>
<td>Status script parameters</td>
<td>(Optional) The parameters that are to be passed to the Status script.</td>
</tr>
<tr>
<td>Status script timeout (seconds)</td>
<td>(Optional) The timeout value for the OpenEdge Management to wait for tracking the Batch Program's status. The default value is 15 seconds. The Status script must execute and terminate within the timeout value. If the Status script terminates with an exit code 0, OpenEdge Management considers that the status of the Batch Program as passed. If the script terminates with an exit code other than 0, OpenEdge Management considers that the status as failed. If the script is missing or cannot be executed, OpenEdge Management considers that the Program is not running and failed.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Status script, if running, after the specified timeout. If this option is not selected and the Status script is not terminated after the specified timeout, OpenEdge Management writes a message to the Batch Program log files.</td>
</tr>
</tbody>
</table>
The absolute path to the location where PID is maintained by Startup or Shutdown scripts. However, the path can also be a relative path or to OpenEdge work directory ($DLC$\%WRK).

The behavior of OpenEdge Management when the status of the Batch Program is unknown. Select one or more behaviors from the following options:

- **Fire alert** — Fires an alert.
- **Perform action** — Executes an action.
- **Restart the program** — Restarts the Batch Program.

The action that you want to associate with the failure of the Batch Program. For more information, see Using Batch Program default actions on page 260.

Once you specify the exit code and select its associated action, click **ADD/UPDATE** to add them. You can choose to add multiple exit codes and their associated actions. Click **DELETE** to delete your selection.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID file</td>
<td>The absolute path to the location where PID is maintained by Startup or Shutdown scripts. However, the path can also be a relative path or to OpenEdge work directory ($DLC$%WRK).</td>
</tr>
<tr>
<td>Status failure behavior</td>
<td>The behavior of OpenEdge Management when the status of the Batch Program is unknown. Select one or more behaviors from the following options:</td>
</tr>
<tr>
<td>Action</td>
<td>The action that you want to associate with the failure of the Batch Program. For more information, see Using Batch Program default actions on page 260.</td>
</tr>
</tbody>
</table>

Once you specify the exit code and select its associated action, click **ADD/UPDATE** to add them. You can choose to add multiple exit codes and their associated actions. Click **DELETE** to delete your selection.
Batch Program Shutdown properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shutdown method</td>
<td>The method to stop the Batch Program. Select from the following options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Shutdown script</strong></td>
</tr>
<tr>
<td></td>
<td>• <strong>Terminate</strong></td>
</tr>
<tr>
<td></td>
<td>If you select <strong>Terminate</strong>, OpenEdge Management looks for the Process ID (PID) of the Batch Program and stops the Batch Program. If the PID is not available, OpenEdge Management fails to stop the Batch Program.</td>
</tr>
<tr>
<td></td>
<td>The <strong>Terminate</strong> option works only if PID file or PID registration is used for status monitoring, or the startup program is running even after the specified timeout value and exit code is selected for status monitoring.</td>
</tr>
<tr>
<td>Shutdown script</td>
<td>(Optional) The path to the script to stop the Batch Program.</td>
</tr>
<tr>
<td>Shutdown script parameters</td>
<td>(Optional) The parameters that are to be passed to the Shutdown script.</td>
</tr>
<tr>
<td>Shutdown script timeout (seconds)</td>
<td>(Optional) The timeout value for the OpenEdge Management to wait for executing the Shutdown script to stop the Batch Program.</td>
</tr>
<tr>
<td></td>
<td>The default value is 15 seconds. If the Shutdown script is not executed within the timeout value, OpenEdge Management considers it as a failure.</td>
</tr>
<tr>
<td>Kill if still running after timeout</td>
<td>Terminates the Shutdown script, if running, after the specified timeout.</td>
</tr>
<tr>
<td></td>
<td>If this option is not selected and the Shutdown script is not terminated after the specified timeout, OpenEdge Management considers the shutdown as a failure and leaves the resource in its previous state.</td>
</tr>
</tbody>
</table>
### Batch Program Environment variable properties

#### Table 119: Environment variable properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>(Optional) The name or ID of the user account on the machine where the process is to be executed. This field allows you to determine the permissions that the process must have when executed.</td>
</tr>
<tr>
<td>Group</td>
<td>(Optional) (UNIX only) The name of the group.</td>
</tr>
<tr>
<td>Password</td>
<td>(Optional) The user account's password for authentication. It is stored in the configuration database.</td>
</tr>
<tr>
<td>Environment variables</td>
<td>(Optional) The environment variables specified by the user. These variables are provided during the execution of the Startup, Status, and Shutdown scripts.</td>
</tr>
<tr>
<td>Debug program</td>
<td>Debugs the Batch Program resource and includes debugging information in its output. This enables the jvmstart tool to write a trace log, and has no effect on the Startup, Status, or Shutdown scripts.</td>
</tr>
</tbody>
</table>

### Using Batch Program default actions

OpenEdge Management supports log and email actions for Batch Program resource. You can create your own log and email actions, or use the sample email template for `Batch_Program_Stopped` action.

When a log file action is triggered, OpenEdge Management writes the start time, stop time, and status of the current Batch Program process to the configured log file.
When an email action is triggered, OpenEdge Management invokes the email action with a set of values appropriate for the process. The following table describes the properties used in the email message template:

**Table 120: Email message template properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOSTLINK</td>
<td>The URL to OpenEdge management.</td>
</tr>
<tr>
<td>PIDFILE</td>
<td>The PID file configured for the Batch Program process.</td>
</tr>
<tr>
<td>PROCESSCONTAINER</td>
<td>The container name of the Batch Program resource.</td>
</tr>
<tr>
<td>PROCESSENVVARS</td>
<td>The environment variables configured for the Batch Program.</td>
</tr>
<tr>
<td>PROCESSPID</td>
<td>The PID of the last known process.</td>
</tr>
<tr>
<td>PROCESSSTARTTIME</td>
<td>The start time of the process, if available. This requires a PID that OpenEdge Management can query.</td>
</tr>
<tr>
<td>PROCESSSTATE</td>
<td>The reported state of the process such as initializing, starting, running, stopping, not running.</td>
</tr>
<tr>
<td>PROCESSSTATUSMSG</td>
<td>The Status message of the current process when the email action is triggered.</td>
</tr>
<tr>
<td>PROCESSSTOPTIME</td>
<td>The stop time of the process, if OpenEdge Management is able to determine it. Otherwise, the stop time of the end of the shutdown script.</td>
</tr>
<tr>
<td>PROGRAMNAME</td>
<td>The name of the Batch Program resource.</td>
</tr>
<tr>
<td>SHUTDOWNSCRIPT</td>
<td>The Shutdown script, if configured.</td>
</tr>
<tr>
<td>SHUTDOWNSCRIPTARGS</td>
<td>The Shutdown script arguments, if configured.</td>
</tr>
<tr>
<td>STARTUPSCRIPT</td>
<td>The Startup program, if configured.</td>
</tr>
<tr>
<td>STARTUPSCRIPTARGS</td>
<td>The arguments for the Startup program.</td>
</tr>
<tr>
<td>STATUSSCRIPT</td>
<td>The Status script, if configured.</td>
</tr>
<tr>
<td>STATUSSCRIPTARGS</td>
<td>The Status script arguments, if configured.</td>
</tr>
<tr>
<td>STDERRFILE</td>
<td>The stderr file configured for the Batch Program process.</td>
</tr>
<tr>
<td>STDOUTFILE</td>
<td>The stdout file configured for the Batch Program process.</td>
</tr>
</tbody>
</table>

Few properties mentioned in the table are not applicable when an email action is fired for a status failure without a process involved in it. For example, the properties PROCESSSTARTTIME, PROCESSSTOPTIME, PROCESSSTATE, PROCESSPID are not applicable without a process object.
Using PID registration

To register a Batch Program for tracking using PID registration:

1. From the OpenEdge Management console menu, click Resources > Go to Resources.
   All the resources managed by your console appear in a grid frame.
2. Filter or search for, and select the required Batch Program resource.
   The Batch Program: <instance name> page appears.
3. In the Command and Control section, click Configuration.
   The Edit Batch Program page appears.
4. Click EDIT.
5. In the Monitoring section, select the Monitoring method as PID registration.
6. Select Status failure behavior as Fire alert.
7. In the following curl example, replace the Process ID in the URL and the PID in the JSON body as required
   and run it.
   ```
   curl -u admin:admin -H "Content-Type: application/json" -d '{"pid": 5380 }' -X POST
   http://localhost:9090/oem/containers/localhost/batch/localhost%3Aresource.batch.apoint/processes/5380
   ```
   The registered process appears in the Processes section of the Batch Program: <instance name> page.
8. In the Shutdown section, select Shutdown method as Terminate or Shutdown script to stop the process.
9. Click SAVE.

OpenEdge Management tracks a Batch Program's PID periodically based on the monitoring plan and notifies
you when the process is no longer running by firing alerts, and runs the action associated with the default
action.

**Note:** OpenEdge Management is unable to detect the exit code for a process managed through PID registration.
In such case, if the process is stopped outside the control of OpenEdge Management, OpenEdge Management
 can only use the action associated with the default exit code. If you need OpenEdge Management to respond
to specific exit codes, you must use a startup script and start the process with OpenEdge Management.
API references

The following API references enable you to manage a Batch program process:

- GET( ) method (Process) on page 263
- POST( ) method (Process) on page 264
- POST( ) method on page 265
- PUT( ) method (Process) on page 265
- DELETE( ) method (Process) on page 266
- GET( ) method (Processes) on page 267

GET( ) method (Process)

GET /containers/{container}/batch/{key}/processes/{processid}

Retrieves details of a specific process associated with the Batch Program.

A Batch Program configured for monitoring using a PID file, PID registration, or status script may have a process that has never started directly from OpenEdge Management. This method is used to track the instance when not using a startup script from within OpenEdge Management.

Endpoint URL

/containers/{container}/batch/{key}/processes/{processid}

- {container} refers to the name of the AdminServer on which the Batch Program process is running.
- {key} refers to the OpenEdge Management resource key for the Batch program process.
- {processid} refers to the Process ID assigned for tracking the Batch program process through the REST API.

HTTP method

GET

Parameters

None

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>The URL to access the details of the Batch Program process.</td>
</tr>
</tbody>
</table>

Sample request

None
Sample response

```json
{
    "name": "prowin2",
    "pid": 19280,
    "startTime": 1488325450000,
    "processId": "BCKZKSpvSMiZOy5BztrcqQ",
    "state": "Not Running",
    "statusMessage": "Not Running",
    "icon16": "/images/batch16.png",
    "icon32": "/images/batch32.png",
    "url": "/oem/containers/localhost/batch/localhost%3Aresource.batch.prowin/processes/BCKZKSpvSMiZOy5BztrcqQ"
}
```

POST( ) method (Process)

**POST /containers/{container}/batch/{key}/processes/{processid}**

Registers a process that is to be tracked by the Batch Program using the specified ID.

For a Batch Program resource that is configured for PID registration, the PID must be available on the same machine as the AdminServer where the Batch Program resource is configured. You can provide your own specified identifier that is used for the PID.

**Endpoint URL**

```
containers/{container}/batch/{key}/processes/{processid}
```

- `{container}` refers to the name of the AdminServer on which the Batch Program process is running.
- `{key}` refers to the OpenEdge Management resource key for the Batch program process.
- `{processid}` refers to the Process ID assigned for tracking the Batch program process through the REST API.

**HTTP method**

POST

**Parameters**

None

**Sample request**

```json
{
    "pid": 19280
}
```

**Sample response**

```json
{
    "name": "prowin2",
    "pid": 19280,
```
Code example
The following is a curl example to register a Batch Program for tracking using PID registration:

```bash
```

**POST( ) method**

**POST /containers/{container}/batch/{key}/processes**

This method is for the Batch Program process that is not assigned with a Process ID from the client in the URL. OpenEdge Management assigns a Process ID for this process.

**Endpoint URL**

```
/containers/{container}/batch/{key}/processes
```

- `{container}` refers to the name of the AdminServer on which the Batch Program process is running.
- `{key}` refers to the OpenEdge Management resource key for the Batch program process.

**HTTP method**

POST

**Parameters**

None

**Sample request**

None

**Sample response**

None

**PUT( ) method (Process)**

**PUT /containers/{container}/batch/{key}/processes/{processid}**

Updates the process to use a different PID.
Endpoint URL

/containers/{container}/batch/{key}/processes/{processid}

- `{container}` refers to the name of the AdminServer on which the Batch Program process is running.
- `{key}` refers to the OpenEdge Management resource key for the Batch program process.
- `{processid}` refers to the Process ID assigned for tracking the Batch program process through the REST API.

HTTP method

PUT

Parameters

None

Sample request

```
{
    pid: 12334
}
```

Sample response

None

DELETE( ) method (Process)

DELETE /containers/{container}/batch/{key}/processes/{processid}

Removes the tracking of the specified Batch Program process and stops the process before removing it.

**Note:** Stopping the process before removing the process is optional.

The **Shutdown method** to stop the process can be either **Terminate** or **Shutdown script**. If this method stops the process before removing it, the process is no longer tracked. If this method is not specified, the tracking of the process is removed without stopping the process.

Endpoint URL

/containers/{container}/batch/{key}/processes/{processid}

- `{container}` refers to the name of the AdminServer on which the Batch Program process is running.
- `{key}` refers to the OpenEdge Management resource key for the Batch program process.
- `{processid}` refers to the Process ID assigned for tracking the Batch program process through the REST API.

HTTP method

DELETE
Parameters
This method uses stop as a parameter in the endpoint URL.

/containers/{container}/batch/{key}/processes/{processid}?stop=true

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stop</td>
<td>boolean</td>
<td>Yes</td>
<td>Indicates if the process must be stopped before removing it. The default is set to false. If the shutdown of the process fails, the process tracking is not removed.</td>
</tr>
</tbody>
</table>

Sample request
None

Sample response
None

GET( ) method (Processes)

GET /containers/{container}/batch/{key}/processes

Retrieves a list of processes tracked by OpenEdge Management for a Batch Program.

Endpoint URL

/containers/{container}/batch/{key}/processes

• {container} refers to the name of the AdminServer on which the Batch Program process is running.
• {key} refers to the OpenEdge Management resource key for the Batch program process.

HTTP method
GET

Parameters
None

Sample request
None

Sample response
None
Chapter 10: Configuring Batch Programs
11

Configuring third-party Web applications

This section describes how OpenEdge Management enables you to host and manage custom third-party Web applications available in your production environment.

For details, see the following topics:

• Overview
• Requirements for hosting a third-party Web application
• Hosting a third-party Web application
• Page contributions for a third-party Web application
• OpenEdge Management Extension framework
• Third-party Web application layout
• OpenEdge Management REST API framework
• Sample Web application in OpenEdge Management

Overview

OpenEdge Management helps you monitor the availability and performance of OpenEdge resources. To accommodate management capabilities for Web applications beyond the OpenEdge environment, OpenEdge Management supports custom third-party Web applications.
This allows you to manage Web applications that are available in your own production environment through the OpenEdge Management console. In other words, you can now manage both OpenEdge resources as well as non-OpenEdge Web applications using a single management console.

With support for third-party Web applications, you can host your own Java Web applications in the OpenEdge Management Web server, integrate your own web pages, and extend existing OpenEdge Management functionality. Third-party Web applications can take advantage of OpenEdge Management's Web server security and styling, and specific extension points provided in OpenEdge Management.

**Extension mechanism in OpenEdge Management**

The extension mechanism in OpenEdge Management to support third-party Web applications allows you to:

- Store and retrieve resource properties.
- Host third-party standard Java Web applications within the Jetty Web server.
- Add your own web pages and access them through the management console main menu.
- Implement an extension point to contribute to your own landing page.
- Use the sample Web application shipped with OpenEdge Management as a template to take advantage of OpenEdge Management security and styling.
- Use the REST API framework implementation to design your own REST APIs within your Web application.

To begin with, the following are provided:

- A Web application template as a sample for resource implementation.
- Limited access to OpenEdge Management internals to perform the above mentioned tasks through restricted API.
- Documentation to access a custom Web application with custom resources and web pages in OpenEdge Management.

The following are not provided:

- Direct access to cache database or to create of new charts.
- Direct access to the OpenEdge Management configuration database.
- Direct access to OpenEdge Management class internals beyond what is available in the documentation.
- Direct access to remote AdminServers or to create new remote AdminServer plugins.

**OpenEdge Management security**

As you manage your Web applications using OpenEdge Management, one of the primary concerns is the security of the data transferred over the Internet. Attackers can exploit the vulnerabilities in Web applications to maliciously gain access to the management console and additional privileges to cause service disruption.

For security reasons, OpenEdge Management discourages the use of inline JavaScript. OpenEdge Management is configured to disable inline JavaScript when a web page request is made. And to address Cross-site scripting (XSS) attacks, OpenEdge Management recommends the use of penetration tools such as OWASP Zap to perform security testing for third party Web applications.

To simplify the security configuration, OpenEdge Management provides a SSO process for all Web applications running in its Web server. For more information, see Implementing single sign-on (SSO) on page 271.
Requirements for hosting a third-party Web application

To host a third-party Web application in OpenEdge Management Web server and make the integration seamless, the following are required:

- **Single sign-on (SSO)** — Creating a single sign-on session between OpenEdge Management and your Web application to provide a single point of entry to users.

- **OpenEdge Management JSP tags** — Accessing OpenEdge Management’s standard JSP tags allows you to share and create a consistent look and feel across the management console.

- **Web application support** — Hosting a standard Java Web application that consists of a web.xml file, JSP pages, images, style sheets, and JAR files.

Hosting a third-party Web application

OpenEdge Management is configured to use the Jetty Web server that is capable of loading third-party Web applications. In OpenEdge Management, the Web user interface is a Web application that consists of JSP files, CSS files, and image files, and the associated Java code is hosted in the fathom.jar file. The OpenEdge Management Web application is considered the ROOT Web application. Third-party Web applications can have their own names without conflicting with the directory structure of the ROOT Web application.

You can host standard Web applications, for example web.xml and its related configuration items, that are supported by the Jetty Web server. Third-party Web applications are hosted in the OEM/webapps directory. You can edit the fathomWebApplicationDir system property in the fathom.init.params file to change the directory and its default value.

---

**Note:** To begin with, you can use the sample Web application WAR file provided with the OpenEdge Management installation.

Though, by default, third-party applications have access to fathom internals, direct access to all internal methods is restricted. However, a set of public objects that contain a subset of OpenEdge Management internals are provided so that a third-party Web application can compile against these public objects.

**Note:** Progress reserves the right to modify any internal methods unless the methods are explicitly mentioned in OpenEdge documentation that they are available for third-party Web application use.

---

Implementing single sign-on (SSO)

In general, Web applications hosted in a container such as Jetty are independent of other Web applications, and each Web application is configured with its own security implementation. To simplify the security configuration for third-party Web applications and provide a single point of entry to users, OpenEdge Management provides an SSO process—sharing a session cache implementation—for all Web applications running in its Web server.
To share a session cache implementation, you can use the sample Web application shipped with OpenEdge Management as a template. A custom session cache configuration shares session login information with the third-party Web application. You can use this configuration as an SSO solution for your third-party Web applications or provide your own security implementation.

**Note:** OpenEdge Management login configuration is available in the `web.xml` file and the `shiro.in` file in the sample Web application template that is shipped with OpenEdge Management.

### Accessing JSP Standard Tag Libraries

To share and create a consistent look and feel between OpenEdge Management and third-party Web applications, you must access OpenEdge Management’s standard JSP tags. These include:

- Toolbars, web page sections such as headers and footers, common style sheets, and common JavaScript libraries that are shipped with OpenEdge Management.

  You can see the web pages available in the sample Web application shipped with OpenEdge Management as an example. The common fathom style sheet is available in `/stylesheets/fathom.css`.


  Standard tag libraries are available in the sample Web application. However, if required, you can add other standard tag libraries for your application.

**Note:** OpenEdge Management ships with a full implementation of ExtJS and Kendo UI Professional frameworks. To use Kendo or third-party libraries, ensure that you have correct development licenses. Progress is not responsible for renewing the licenses purchased for these libraries that are used for third-party Web applications. Alternatively, you can use your own JavaScript libraries, but ensure that necessary library resources are included as part of your Web application.

### Loading Java classes

A class loader provides libraries and JSP pages, which are compiled at runtime into Java classes, to a Web application. The class loader of your Web application requires access to JAR files, a few internal classes, and any API classes of OpenEdge Management that must be exposed to third-party Web applications.

OpenEdge Management must ensure that the class loader is configured properly so that the class loader is created when the Web server is being initialized at the startup of OpenEdge Management. Each class loader of a third-party Web application is considered a child of OpenEdge class loader and remains independent of other Web applications. Each third-party Web application is defined as an extension to OpenEdge Management and the search path of the Web application class loader is allowed to access OpenEdge Management classes.

OpenEdge Management also provides shared libraries that allow third-party Web applications to host their own JAR files. The `$OEM/jars` directory acts as a shared library for all JAR files that are shipped with OpenEdge Management. The third-party Web applications can host their own JAR files in the `WEB-INF/lib` directory of the Web application.

**Note:** Each Web application must have its own class loader in order to work with the Jetty Web server.
Page contributions for a third-party Web application

OpenEdge Management provides hooks, located in predefined locations, which allow a third-party Web application to add user interface elements to every page. In this release, only a limited set of hooks are provided, for example, main menu and footers in the OpenEdge Management console.

To register these hooks and to inject a custom `PageContributor` class into OpenEdge Management, a registration mechanism is provided for third-party Web applications. During execution of every page, the methods in the page contributor instance are invoked at specific points.

**Note:** A third-party Web application must implement its own `PageContributor` class in order to work with OpenEdge Management.

OpenEdge Management implements specific callbacks for the page contributor to the following locations in the management console:

- **Page header** — To inject script and style.
- **Main menu** — To add a functional menu to the main menu bar.
- **Footer menu** — To add new entries to the footer.
- **Options page** — To add new entries to the **Options** page.

The extension mechanism provides the following configuration entry points to register extensions in OpenEdge Management:

- Extension name — The name of the extension.
- Extension class path — Extends the default behavior of OpenEdge Management.
- Extension initializer class — For startup, state, and shutdown of the extension.
- Standard file layout — The location of the configuration file of the extension.
- Standard Java interfaces — Supports the above entry points.
- Dashboard entry point class — Adds top-level pages.
- Configuration entry point class — Adds entries to the **Options** page in the management console.
- User login entry point page — Allow user logins to open the Dashboard page or any other page defined by the plugin.

**Behavior of OpenEdge Management**

OpenEdge Management implements callbacks to the exact contributor whenever necessary, and recognizes the specific locations in the management console as unique identifiers. These identifiers are hard-coded and can be used by menu contributors to determine the context. OpenEdge Management determines when to call the Contributor class, and may invoke the methods of a contributor more than once per page or may not invoke them at all.

For example, a callback for contributors to the OpenEdge Management console main menu receives a parent `main MENU-ITEM` object and a page context. The Contributor class adds child objects. In cases where the same callback is implemented to many locations, OpenEdge Management provides sufficient context information to the contributors to generate the required child objects.
OpenEdge Management ensures that each page element is uniquely identified. For example, for a parent MENU-ITEM that has been passed to the Contributor instance, OpenEdge Management sets a unique "id" property to the MENU-ITEM so that it can be uniquely identified.

Registering a contributor

Third-party Web application menu contributor objects must be registered with OpenEdge Management. A contributor is registered when the third-party Web application is initialized. The third-party Web application provides a configuration property that defines a properties file that has the name of the Contributor class.

OpenEdge Management creates and implements the ServletContextListener class to ensure that the Contributor class is instantiated in the required Web application context and registered with OpenEdge Management. When registering the contributor class is no longer required, the ServletContextListener implementation ensures that the contributor is removed from memory.

The context listener loads the properties from the WEB-INF/fathomextension.properties file. This file contains details such as name, version, and properties that specify the contributor class. For more information, see the sample Web application.

OpenEdge Management Extension framework

OpenEdge Management allows extensions to contribute to the user interface to third-party Web applications. Predefined points are provided where the extensions can be added to the user interface. The primary points are main menu, page footers, and the Options page of the management console.

To accomplish this, a framework has been built to allow simple extensions through contributor class objects. Each individual point, where extensions are added to the user interface, requests necessary data using the fathom extension contributor classes. You can add other contributor classes to the framework, but only without altering the extension code of a previous version of the framework.

The extension framework allows hosting of third-party Web applications on the same Web server as that of OpenEdge Management. This enables OpenEdge Management to define the contributor classes as needed. The lifecycle of the extension points can be controlled using the lifecycle of the third-party Web application.

Accessing OpenEdge Management internals

Typically, OpenEdge Management is a Web application designed to be isolated from other Web applications. The extension framework restricts direct access to OpenEdge Management internals.

To allow third-party Web applications to access OpenEdge Management internals, a standard Java Web application servlet listener is written. This servlet listener searches for and loads necessary configuration files that have required contributor classes. This avoids hosting of third-party Web application JAR files in the OpenEdge Management Web application without restricting extensions to OpenEdge Management.
Third-party Web application layout

OpenEdge Management uses the Jetty Web server, a Java servlet container, which can host multiple Web applications. This allows the Web applications to load their own context listener. OpenEdge Management has a single servlet context listener to apply extensions to the user interface. This context listener looks for a standard Java properties file named WEB-INF/fathomextension.properties that has the configuration information for extensions to OpenEdge Management.

You can define extensions to OpenEdge Management by implementing specific classes and referencing known property names in the fathomextension.properties file to define which classes implement which interfaces.

An example of a servlet context listener specified in a web.xml file is provided in Table 1:

Table 121: WEB-INF/web.xml

```
<listener>
  <listener-class>com.progress.fathom.extension.ExtensionLoaderListener</listener-class>
</listener>
```

A sample fathomextension.properties file with a fathom extension initializer implementation and menu contributor implementation is provided in Table 2:

Table 122: WEB-INF/fathomextension.properties

```
# this is a sample fathom extension definition file. This file contains
an example set of contributor classes that add UI elements to OpenEdge Management
# this is optional internal name of web extension
name=samplewebapp
# this is optional internal version of web extension
version=1.0
# this is an optional property that references a class that is an instanceof
IExtensionInitializer
initializer=com.progress.fathom.sample.webapp.SampleApplicationInitializer
# this is an optional property that references a class that is an instance of
IMenuContributor
menuContributor=com.progress.fathom.sample.webapp.SampleMenuContributor
# sample landing page contributor. The default fathom implementation does not do anything
landingPageContributor=com.progress.fathom.extension.preferences.FathomLandingPageContributor
```

When initialized, the servlet context listener instantiates the classes defined for known properties in the fathomextension.properties file. Though other properties in the file are ignored, they can be accessed from the initializer class implementation when required.

Once the extensions are loaded, they can be seen in the extension diagnostic page:
Currently there are limited set of known properties that the context listener looks for, and they are described below:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>(Optional) Name of the fathom extension. This can be any string.</td>
</tr>
<tr>
<td>version</td>
<td>(Optional) Version of the fathom extension, displayed only in the extension diagnostic page. This can be any string.</td>
</tr>
<tr>
<td>initializer</td>
<td>(Optional) Fully qualified name of a class that implements IExtensionInitializer. A single instance of this class is stored in the memory until the Web application is terminated. This class can initialize any information required by the fathom extension.</td>
</tr>
<tr>
<td>menuContributor</td>
<td>(Optional) Fully qualified name of a class that implements IMenuContributor. A single instance of this class is stored in the memory until the Web application is terminated. This class is used by the fathom extension to add items to the main menu and footer menu of the console.</td>
</tr>
<tr>
<td>landingPageContributor</td>
<td>(Optional) Fully qualified name of a class that implements ILandingPageContributor. A single instance of this class is stored in the memory until the Web application is terminated. This class is used by the fathom extension to add landing pages that the user can select to use immediately after logging into the console.</td>
</tr>
</tbody>
</table>

OpenEdge Management REST API framework

The REST API in OpenEdge Management is built on an annotation-based framework to define a REST interface. HTTP clients use the REST interface to interact with OpenEdge Management. Third-party Web applications looking to extend OpenEdge Management functionality in their own Web application can use the OpenEdge Management REST API framework.

The REST API framework provides permission checks, parameterized URL matching, bi-directional object conversion of JSON-mapped entities, and handling upload of multi-part encoded files. The framework supports GET, PUT, POST, and DELETE HTTP methods. The sample Web application shipped with OpenEdge Management is an example of a simple REST API with sample GET and POST operations.

Note: The REST API framework in OpenEdge Management relies on the Jackson third-party library, an open source library available for use within your Web application.
Configuring a third-party Web application

The REST API framework in OpenEdge is built on the standard JAVA servlet framework. The code for the REST API framework is built into the fathom.jar file and is available to third-party Web applications. The servlet receives its configuration from an XML file which specifies the classes that are to be scanned for annotations.

Note: The servlet must be defined in the Java Web application web.xml file.

An example of REST servlet definition and configuration in an XML file is provided in Table 3:

Table 123: WEB-INF/web.xml

```xml
<servlet>
  <servlet-name>RestServlet</servlet-name>
  <servlet-class>com.progress.fathom.FathomAPIServlet</servlet-class>
  <init-param>
    <param-name>configuration</param-name>
    <param-value>/WEB-INF/restconfig.xml</param-value>
  </init-param>
  <init-param>
    <param-name>urlprefix</param-name>
    <param-value>/rest</param-value>
  </init-param>
</multipart-config>
</servlet>
<servlet-mapping>
  <servlet-name>RestServlet</servlet-name>
  <url-pattern>/rest/*</url-pattern>
</servlet-mapping>
```

The file format of the configuration XML file is simple. The file can define any number of implementations. Each <fathomapi/> node defines the fully qualified name of a class that extends the FathomAPIServletHandler class. To handle URL mapping and request access to a context, the derived class must inherit from the FathomAPIServletHandler class.

The XML file specifies the classes that are to be scanned for annotations to map them to URLs. Servlet mapping is defined in the web.xml file and all the URLs in the classes are made available for mapping by the Java servlet beneath this servlet mapping. In the above example, all the URLs mapped from the classes are made available beneath /<webcontextpath/rest/>. 
An example of a complete `restconfig.xml` file is provided in Table 4:

**Table 124: WEB-INF/restconfig.xml**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<fathomapis>
  <fathomapi>
    <!-- this is optional. This provides standard /api human readable content -->
    <class>com.progress.fathom.api.OEMAPI</class>
  </fathomapi>
  <fathomapi>
    <!-- simple example that shows how to consume and generate JSON content via
    REST API -->
    <class>com.progress.fathom.sample.webapp.RestExample</class>
  </fathomapi>
</fathomapis>
```
REST API code example

An example of the code used in the samplwebapp.war file shipped with OpenEdge Management is provided in Table 5:

Table 125: RestExample.java

```java
package com.progress.fathom.sample.webapp;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.node.ObjectNode;
import com.progress.fathom.Body;
import com.progress.fathom.Description;
import com.progress.fathom.FathomAPIServletHandler;
import com.progress.fathom.PathParam;
import com.progress.fathom.QueryParam;
import com.progress.fathom.RequestException;
import com.progress.fathom.RequestMapping;
import com.progress.fathom.RequestMethod;
import com.progress.fathom.RequiredResourcePermission;
import com.progress.fathom.Since;
import com.progress.isq.ipqos.resources.security.SecurityBean;
/**
 * this is sample REST API handler based on the OpenEdge Management REST
 * annotation mapping
 * *
 */
public class RestExample extends FathomAPIServletHandler {

    @Description("Example REST get request")
    @RequestMapping(path = "/get/{id}", method = RequestMethod.GET)
    @Since("11.7")
    @RequiredResourcePermission(grantName = SecurityBean.SECURITY_ROOT, action = SecurityBean.ACTION_READ)
    public Object getSample(@PathParam("id") String id, @QueryParam("q") String queryString) throws RequestException {
        ObjectMapper mapper = new ObjectMapper();
        // this creates a simple JSON response using the values from the URL
        // mapped back into the object as JSON object properties
        ObjectNode response = mapper.createObjectNode();
        response.put("id", id);
        return response;
    }

    @Description("Example REST POST request")
    @RequestMapping(path = "/post/{id}", method = RequestMethod.POST)
    @Since("11.7")
    @RequiredResourcePermission(grantName = SecurityBean.SECURITY_ROOT, action = SecurityBean.ACTION_READ)
    public Object postSample(@PathParam("id") String id, @Body ObjectNode body) throws RequestException {
        ObjectMapper mapper = new ObjectMapper();
        // this creates a simple JSON response using the values from the URL
        // mapped back into the object as JSON object properties, along with the
        // content of the request body
        // from the POST
        ObjectNode response = mapper.createObjectNode();
        response.put("id", id);
        response.put("request", body);
        return response;
    }
}
```
The REST API framework scans each class for methods that are public and have a `RestMapping` annotation defined. The annotation specifies the URI pattern and HTTP method that are to be handled. The `RequestMapping` annotation with a specified path parameter is the least required annotation. If the method parameter is not specified, the annotation considers HTTP GET as the default method parameter.

**Note:** Each class can contain as many annotated methods as required. Each method must handle only one combination of URI pattern and HTTP method.

**URL mapping**

The REST API framework performs URL mapping based on the path specified in the `RequestMapping` annotation. The URL mapping accepts parameters defined within the URL string, and then the mappings are made available to parameters for a method through annotations.

The REST API code example on page 279 shows both URL and query-parameter mapping. The mapping is based on a simple pattern matching of the incoming URL, and each parameter is treated as a wildcard string for its position within the URL. Only the URL part of the request is matched against the annotated REST handler methods, and the query string of the URL is ignored.

Consider an example where the path: `/some/{container}/url/{key}` has two parameters defined in it: `container` and `key`. Their values from the URL can be retrieved either from annotated method parameters or from within the body of the method. In the REST API code example on page 279, the parameters to the method are annotated with `PathParam("<url parameter>")`, where the `url parameter` is the name matching the value in the path for the method.

**Exception handling**

The REST API framework expects the REST handler methods to throw exceptions of `RequestException` type. The `RequestException` requires an HTTP response code and a description text that is used to return an error to the client. However, an inner exception may also be passed to the constructor.

If a `RequestException` is thrown from a REST handler method, the framework considers it as a known exception and returns a JSON formatted response body with error text to the client. In some cases, the REST handler method might throw any subclass of `RequestException`.

The framework considers the `RequestException` as a normal termination of the REST handler method and does not log the exception. On other hand, an exception that does not extend `RequestException` is considered as an application error and recorded in the fathom error log file. The client then receives a response body with error text `HTTP 500 Internal Server Error` from the exception.
An example of `RequestException`:

```java
@RequestMapping(path = "\get", method = RequestMethod.GET)
public Object getSample(@QueryParam("id") String idFromQuery) {
    if ("0".equals(id)){
        throw new RequestException("Invalid id", HttpServletResponse.SC_BAD_REQUEST);
    }
    ...
}
```

The corresponding response:

```json
{
    error: {
        stack: [],
        number: "403",
        text: "Invalid id",
        source: "OpenEdge Management"
    }
}
```

**REST handler method response**

Each REST handler method must send a response or throw an exception. The method may handle the response or allow the framework to convert the method return value to a JSON string. All responses are can be Content-Type specified as application/JSON. Though this is not mandatory, the handler method must handle the HTTP Content-Type header and response body.

The response from a method can be one of the following types:

- **String** — If the type of object returned from the method is String, it is returned as-is with no validity checks. The method must ensure that the string is a properly formatted JSON encoded string.

- **ObjectNode** — If the type of object returned from the method is ObjectNode, a part of Jackson JSON library, the framework serializes it and writes it back to the client.

- **Plain old Java object** — If the type of object returned from the method is a Java object, the framework attempts to convert it to a JSON encoded string using the built-in Jackson library. If the object cannot be converted to a JSON encoded string, an **HTTP 500 Internal Server Error** is returned to the client, and the exception is recorded in the fathom error log file.

**Note:** Returning a string or an ObjectNode is quicker than returning a Java object because the latter avoids using reflection to serialize the object.

For details about Jackson capabilities, see https://github.com/FasterXML/jackson.

An example of a method returning String:

```java
@RequestMapping(path = "/get", method = RequestMethod.GET)
public Object getSample(@QueryParam("id") String idFromQuery) {
    return "{"id" : "someid\""}"
}
```
An example of a method returning ObjectNode:

```java
@RequestMapping(path = "/get", method = RequestMethod.GET)
public Object getSample(@QueryParam("id") String idFromQuery) {
    ObjectMapper mapper = new ObjectMapper();
    ObjectNode resp = mapper.createObjectNode();
    resp.put("id", "someid");
    return resp;
}
```

An example of a method returning Plain old Java object:

```java
@RequestMapping(path = "/get", method = RequestMethod.GET)
public Object getSample(@QueryParam("id") String idFromQuery) {
    MyThing thing = new MyThing();
    thing.setId("someid");
    return thing;
}
```

Method annotations

Several annotations are available at method level that perform method-related tasks. A few are provided here:

Description

Accepts a text string as its value. The text appears in the API method description displayed from "/api" request handler.

**Note:** This annotation is for documentation purposes only.

If the com.progress.fathom.api.OEMAPI class is listed in the restconfig.xml file, the "/api" URL is mapped and provides a human-readable description of the URLs that are mapped for the Web application. The description text is included in the response.

Refer to the REST API for the sample Web application on page 287 for an example of an API for a sample Web application containing description text.

Since

Accepts a text string as its value. Though this text does not appear in any example, you can use it as you see fit.

**Note:** This annotation is for documentation purposes only.

RequiredResourcePermission

Adds an authorization check performed by the fathom REST servlet implementation. Authorization checks in OpenEdge Management are based on a security object identified (a string), and an action.

If the user making the REST request exists and is authenticated, the authentication check must succeed to invoke the method. If the user is not authenticated, the REST request fails with a 403 Forbidden response. If the user exists and the RequiredResourcePermission annotation is not available, the user is authorized to access OpenEdge Management to perform the API call.
At the least, the following annotation must be added to prevent users without normal resource permissions, such as the PSCTrend role, from accessing the method. However, users with PSCOper and PSCAdmin roles will still be able to access the method.

```java
@RequiredResourcePermission(grantName = SecurityBean.SECURITY_ROOT, action = SecurityBean.ACTION_READ)
```

For a list of predefined permission objects, see the SecurityBean class. At the least, your Web application should check `SecurityBean.SECURITY_RESOURCE, action=SecurityBean.ACTION_READ`.

**Note:** Three predefined user roles—PSCOper, PSCAdmin, and PSCTrend—are available in OpenEdge Management, and permissions are granted individually based on the user role. PSCOper and PSCAdmin user roles have full read permissions on security_resource. The PSCAdmin user role has full read/write permissions on all security permission tokens within OpenEdge Management. The PSCTrend user role has limited permissions to a few APIs that are required for remote trending.

### Parameter annotations

Several parameter annotations are available that perform type conversion, resource lookup, URL parameter mapping, and query parameter mapping from the HTTP request. The FathomAPIServletHandler has methods to access the URL, query parameters, HTTP requests and response objects.

You can define any number of parameters for a method. If any parameter cannot be resolved based on its annotation, the parameter value is set to null for Object parameters and default value for Java scalar types (int, float).

A few annotations are provided for your convenience; using them is not mandatory:

- **QueryParam**
  Add this to a method parameter to retrieve a value from the query to the URL.
  For example:

  ```java
  @RequestParam(path = "/get", method = RequestMethod.GET)
  public Object getSample(@QueryParam("id") String idFromQuery) {
      ...
  }
  ```

- **PathParam**
  Add this to a method parameter to retrieve a value from a parameterized URL.
  For example:

  ```java
  @RequestMapping(path = "/get/{id}", method = RequestMethod.GET)
  public Object getSample(@PathParam("id") String id) {
      ...
  }
  ```

- **RequestPathParam**
  Add this to a method parameter to assign the parameter to a source object based on a fully qualified resource key. The value of the key is derived from a URL parameter.
If the resource defined from the value of the key does not exist, the method is not called and a 404 Resource Not Found response is returned to the caller. If the resource exists, but cannot be cast to the data type defined by the method parameter, the caller receives a 403 Bad Request with a JSON response body. The error response indicates that incorrect resource type was discovered by the resource key in the URL.

For example:

```java
@RequestMapping(path = "/get/{key}", method = RequestMethod.GET)
public Object getResource(@ResourcePathParam("key") IResource resource) {
    ...
}
```

Files

Handles an HTTP multi-form encoded request that contains files. The files are saved to disk and passed as a list of UploadedFile objects with details from the POST method.

For example:

```java
@RequestMapping(path = "/files", method=RequestMethod.POST)
public Object postFiles(@Files List<UploadedFile> files) {
}
```

Body

Maps the body of the resource to the method parameter. The framework assigns the value based on the data type of the parameter. This is only for use with PUT, POST, and DELETE methods. HTTP requests that use GET method do not have body content.

**Note:** To ensure that the string conversion behaves properly, the client must send the character set information with the Content-Type header when posting the data using the POST method. If the charset field of the Content-Type header is not available, OpenEdge Management assumes that the character set is IS8859-1. If the Content-Type is specified as application/JSON, the header must be UTF-8.

An example with String as the data type is provided in Table 6:

**Table 126: String REST API method**

```java
@RequestMapping(path = "/content", method=RequestMethod.POST)
public Object postFiles(@Body String requestBody) {
}
```

OpenEdge Management uses the Jackson JSON parser. If the request Content-Type is specified as application/JSON, OpenEdge Management attempts to read the request content and convert it to a Jackson ObjectNode implementation.
An example with Jackson ObjectNode as the data type in Table 7:

**Table 127: Jackson ObjectNode REST API method**

```java
@RequestMapping(path = "/post/{id}", method = RequestMethod.POST)
public Object postSample(@PathParam("id") String id, @Body ObjectNode body) {
}
```

Jackson can convert a JSON structure parsed from a string to a Plain Old Java Object based on standard getter/setter and few annotations. It also supports some annotations on the entity class methods to determine how deserialization is to be performed. The OpenEdge Management REST API framework built this ability to REST handler methods via annotations.

The following example defines a class `MyThing` and uses `Body` annotation to import a request body from the POST method to a `MyThing` object. Jackson maps properties from the JSON request body.

**Table 128: Sample JSON POST content**

```
{
   "id" : "someid",
   "description" : "some description"
}
```

**Table 129: MyThing.class and REST API method**

```java
@RequestMapping(path = "/post/{id}", method = RequestMethod.POST)
public Object postThing(@PathParam("id") String id, @Body MyThing thing) {
}

public class MyThing {
   private String id;
   private String description:

   public String getId() {
      return id;
   }

   public String getDescription() {
      return description;
   }

   public void setId(String id) {
      this.id = id;
   }

   public void setDescription(String description) {
      this.description = description;
   }
}
```
Sample Web application in OpenEdge Management

A typical install of OpenEdge Management comes with a sample Web application WAR file and its original source code bundled in a ZIP file included in the $OEM/samples directory. The sample Web application is a single page application with a sample REST API. To access the REST API sample, see Accessing the REST API extension on page 287. To begin with, you can use this sample Web application as a template for third-party Web applications.

The sample Web application provides the following:

- Java Web application definition including the web.xml file and the required file structure.
- Configuration for Web application security using Shiro. OpenEdge Management uses Shiro for its security configuration.
- A sample REST API implementation based on REST annotated classes in OpenEdge Management.
- Hooks to add menu entries to menus in OpenEdge Management.
- A sample web page with OpenEdge Management styling that can be used as a template for custom web pages.

Using the sample Web application

The sample Web application shipped with OpenEdge Management is available in the $OEM/samples directory.

To use the sample Web application:

1. Stop the AdminServer (using the command fathom -stop or proadsv -stop).
2. Navigate to OpenEdge Management install directory $OEM on your machine. Typically, this is the C:\Progress\oemgmt directory on your Windows machine.
3. Open the Samples folder to find:
   - A sample Web application samplewebapp.war file, which is a compiled and deployable Web application archive (WAR) file readily available for use.
   - The source file for the sample Web application samplewebapp.src.zip bundled in a ZIP file.
4. Copy the sample Web application samplewebapp.war file to the webapps folder in the OpenEdge Management install directory.
   If this folder does not exist, you must create a new folder named webapps in the OpenEdge Management install directory and then copy the WAR file into the folder.
5. Restart the AdminServer (using the command fathom -start or proadsv -start).
6. Open and log into the OpenEdge Management console using http://localhost:9090.
   After logging in, the OpenEdge Management console provides the following extensions as part of the sample Web application:
   - A functional menu, Sample, in the main menu bar for resource implementation.
   - A link in the Options page, Sample Config, to add your own preferences to the third-party Web application.
• A link in the footer, Sample, to provide access to your own web pages or online help.

The sample Web application defines a menu contributor that adds a new menu item, Sample, to the main menu and the footer in the management console. The new menu item is named as Sample; it has few example links beneath it.

A landing page contributor is also defined in the sample Web application. It includes a landing page item for the OpenEdge Management jobs home page. To choose a landing page example, select Options > User Preferences in the management console.

Accessing the REST API extension

To access the REST API extension for the sample Web application shipped with OpenEdge Management:

1. Enter the URL http://localhost:9090/samplewebapp in your web browser.

   The Sample Web Application Example Page appears in the OpenEdge Management console.

2. In the GET example section:
   1. Click rest/get/id1 to see a sample API for GET.
   2. Click rest/api to see a list of REST APIs available for the sample Web application.

3. In the POST example section:
   1. Provide a value as input in the Enter some value field.
   2. Click POST.
   3. The value returned using the API for POST appears in the Output shows up here field.

REST API for the sample Web application

An example of a sample Web application REST API available in OpenEdge Management is provided in Table 10:

Table 130: /rest/api

<table>
<thead>
<tr>
<th>Service URLs marked with a + are generic APIs where the URL handles several APIs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>/rest/api</td>
</tr>
<tr>
<td>GET:</td>
</tr>
<tr>
<td>Description: returns a human readable list of API methods provided by fathom REST interface</td>
</tr>
<tr>
<td>/rest/get/{id}</td>
</tr>
<tr>
<td>GET:</td>
</tr>
<tr>
<td>Description: Example REST get request</td>
</tr>
<tr>
<td>/rest/post/{id}</td>
</tr>
<tr>
<td>POST:</td>
</tr>
<tr>
<td>Description: Example REST POST request</td>
</tr>
</tbody>
</table>
Accessing resources

You can access both OpenEdge and non-OpenEdge resources using the Sample Web Application Example Page in OpenEdge Management.

To access the resources:

1. Enter the URL http://localhost:9090/samplewebapp in your web browser.

   The Sample Web Application Example Page appears in the OpenEdge Management console.

2. In the Search for resource field of the Resources search example section, type some text relevant to the name or description of the resource that you want to access.

   The name and description of first few relevant resources appear in a list.

3. From the list, select the required resource to access its home page.

Compiling sample Web application using Ant

You can compile the sample Web application using an example build.xml file.

To compile the sample Web application:

1. Extract the samplewebapp.src.zip file available in the $OEM/samples directory.
2. Create a build.xml file in the top-level directory using the example file provided below.
4. Change the directory to the directory that contains the sample Web application code.
5. Run proant.

   The following is the example build.xml file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project name="samplewebapp" default="build" basedir="."/>
<!-- make environment variables available as properties -->
<property environment="env"/>
<!-- Compiler settings. -->
<property name="javacFailOnError" value="true"/>
<property name="javacDebugInfo" value="on"/>
<property name="javacVerbose" value="false"/>
<property name="logExtension" value=".log"/>
<property name="compilerArg" value=""/>
<property name="ant.build.javac.source" value="1.8"/>
<property name="ant.build.javac.target" value="1.8"/>

<!-- directory names for convenience -->
<property name="DLC" value="${env.DLC}" />
<property name="OEM" value="${env.OEM}" />
<property name="build.src.dir" value="${basedir}/src" />
<property name="build.out.dir" value="${basedir}/build" />
<property name="build.class.dir" location="${build.out.dir}/classes" />
<property name="samplewebapp.webcontent.dir" value="${basedir}/WebContent"/>
<property name="samplewebapp.jar" location="${build.out.dir}/samplewebapp.jar" />
<property name="samplewebapp.war" location="${build.out.dir}/samplewebapp.war"/>
<property name="samplewebapp.src.zip" location="${build.out.dir}/samplewebapp.src.zip" />
```
<!-- classpath for building samplewebapp.jar -->
<path id="compile.classpath">
  <!-- include progress.jar -->
  <pathelement path="${DLC}/java/progress.jar"/>
  <pathelement path="${DLC}/java/messages.jar"/>
  <!-- include fathom.jar -->
  <pathelement path="${OEM}/jars/fathom.jar"/>
  <!-- java servlet specific stuff needed for most web pages -->
  <pathelement path="${OEM}/jars/servlet-api-*.jar"/>
  <!-- include fathom.jar -->
  <pathelement path="${OEM}/jars/jackson-core-*.jar"/>
  <pathelement path="${OEM}/jars/jackson-annotations-*.jar"/>
  <pathelement path="${OEM}/jars/jackson-databind-*.jar"/>
  <!-- shiro for security -->
  <pathelement path="${OEM}/jars/shiro-core-*.jar"/>
  <pathelement path="${OEM}/jars/shiro-web-*.jar"/>
</path>
</target>
<target name="compile" depends="generic.java.compile"
description="Compile all classes for samplewebapp.jar.">
</target>
<target name="generic.java.compile"
description="Compile all classes in ${build.class.dir} as javasrc
${ant.build.javac.source}"
<mkdir dir="${build.out.dir}"/>
<mkdir dir="${build.class.dir}"/>
<javac
  srcdir="${build.src.dir}"
  destdir="${build.class.dir}"
  includes="**/*.java"
  failonerror="${javacFailOnError}"
  verbose="${javacVerbose}"
  debug="${javacDebugInfo}"
  includeAntRuntime="no"
  <!-- <compilerarg value="-Xlint"/> -->
  <classpath refid="compile.classpath"/>
</javac>
</target>
<tstamp>
</tstamp>
<target name="jar" description="create samplewebapp.jar">
  <tstamp/>
  <jar destfile="${samplewebapp.jar}"
       manifest="${basedir}/META-INF/MANIFEST.MF">
    <manifest>
      <attribute name="url" value="http://www.progress.com"/>
      <attribute name="Vendor" value="Progress Software"/>
      <attribute name="Bundle-Name" value="OpenEdge Management Sample Web Application"/>
      <attribute name="Built-Date" value="${TODAY}"/>
    </manifest>
    <fileset dir="${DLC}"/>
    <fileset dir="${build.class.dir}"
      include="**/*.class"/>
    <fileset>
    </fileset>
  </jar>
</target>
OpenEdge release 11.7 ships with a copy of Ant build tool. A script called proant which sets up and runs the Ant is available in the $DLC/bin directory. Running proant uses the build.xml file to compile and generate the samplewebapp.war file into a directory named build. The example build.xml script can be used as it is or adjusted to your own environment. The build.xml file requires some environment variables that are normally set by Proenv.

The sample Web application also contains .project and .setting files for eclipse. You can import the sample Web application source code as an eclipse project and modify it as required. You can this project to create your own Web application.
Index

4GL debugger enabled property
  AppServer 129
  WebSpeed Transaction Server 201

A

Activate property 130
Admin role
  Web Services Adapter 164
  AdminServer 21
After image buffers property 35
After image process property 35
After image stall property 35
After-image archival directory list property 35
After-image archival interval property 35
After-image Writer 49
Agent maximum port property 54
Agent minimum port property 54
Agent name property 54
Agent Shutdown Action property 55
AIW 49
Alternate buffer pool property 32
Apache Tomcat 226
Append to broker log file property
  AppServer broker 119
  AppServer messaging 133
  MS SQL Server DataServer 89
  ODBC DataServer 65
  Oracle DataServer 77
  WebSpeed broker 189
  WebSpeed messaging 203
Append to log file property 162
Append to NameServer log file property 102
Append to server log file property
  AppServer messaging 134
  WebSpeed agent 197
  WebSpeed messaging 204
Application service names property
  AppServer 118
  WebSpeed 188
Application URL property 200
AppServer Internet Adapters (continued)
  properties (continued)
    general 144
    logging setting 145
    SSL 148
  viewing log files 150
AppServer Keepalive property 149, 215
AppServers
  actions allowed on 114
  additional management 142
  agent properties
    advanced features 129
    general 125–126, 129
    logging setting 126
    pool range 129
  agents
    adding or trimming 136
  broker properties
    advanced features 121
    AppService Name List 118
    controlling NameServer 117–118
    general 116–117
    logging setting 118
    owner information 117
  configuration overview 113
  creating instances 135
  deleting instances 141
  editing a configuration 135
  environment variables 134
  features 114
  messaging properties 132–133
  property storage 115
  SSL properties
    advanced features 132
    general 131
  starting
    automatically 136
    manually 135
  status 137
  viewing log files 142
APW 49
Assigning database default configurations 43
Asynchronous Page Writer 49
Asynchronous page writers property 34
Authorized IP list property 184
Auto start property
  AppServer 116
  MS SQL Server DataServer 87
  NameServer 101
  ODBC DataServer 63
  OpenEdge database 30
Auto start property (continued)
   Oracle DataServer 75
   WebSpeed Transaction Server 187
Auto trim timeout property
   AppServer 122
   WebSpeed Transaction Server 192
Automatically add After imaging extents property 57

B

Background writers 49
Backup method property 57
Base index number property 40
Base table number property 41
Batch Program home 245
Before image buffers property 34
Before image cluster age property 35
Before image delay writers property 34
Before image process property 34
Before image stall property 34
Before image threshold property 34
Before-image Writer 49
Begin After imaging after transition property 57
Binary upload max size property 200
BIW 49
Blocks in database buffers property 31
Broker keep alive timeout property 101
Broker log file threshold size property
   AppServer 121
   MS SQL Server DataServer 89
   ODBC DataServer 65
   Oracle DataServer 77
   WebSpeed Transaction Server 191
Broker log filename property
   AppServer 118
   MS SQL Server DataServer 88
   ODBC DataServer 65
   Oracle DataServer 77
   WebSpeed Transaction Server 188
Broker logging entry types property
   AppServer 119
   MS SQL Server DataServer 89
   ODBC DataServer 65
   Oracle DataServer 77
   WebSpeed Transaction Server 189
Broker logging level property
   AppServer broker 118
   AppServer messaging 133
   MS SQL Server DataServer 89
   ODBC DataServer 65
   Oracle DataServer 77
   WebSpeed broker 189
   WebSpeed messaging 203

C

Case table name property 36
Changing
   AppServer
      agent properties 125
      broker properties 116
      environment variables 134
      messaging properties 132
      SSL properties 131
   AppServer Internet Adapter properties 143
   MS SQL Server DataServer
      broker properties 86
      environment variables 92
      server properties 90
   NameServer
      environment variables 106
      properties 99
   ODBC DataServer
      broker properties 62
      environment variables 68
      server properties 66
   OpenEdge database
      basic properties 30
      default properties 31
      server group properties 44
   Oracle DataServer
      broker properties 74
      environment variables 80
      server properties 78
   SonicMQ Adapter
      broker properties 211
      environment variables 218
      server properties 215
      SSL properties 217
   Web Services Adapter
      advanced features properties 165
      location properties 159, 230
      logging setting properties 161
      properties 158, 230
      proxy server setting properties 160, 231
      run-time properties 172, 234
      security properties 163
      WSDL properties 160
   WebSpeed Messenger
      properties 180
   WebSpeed Transaction Server
      agent properties 195
      broker properties 186
      environment variables 204
      messaging properties 202
      SSL properties 201
Changing OE Web Server run-time properties 234
   Client Type property 46
   Cluster enabled property 30
   Collation table property 36
OpenEdge Management and Explorer: Configuration

Index

Configuration group 28
Configuring
  AppServer components 113
  AppServer Internet Adapters 142
  AppServers 113
  Batch Program 243
  DataServers 61
  MS SQL Server DataServers 85
  NameServers 97
  ODBC DataServers 61
  OpenEdge databases 27
  OpenEdge Web Services 153
  Oracle DataServers 73
  SonicMQ Adapters 209, 225
  Web service instances 174, 236
  Web Services Adapter instances 156
  WebSpeed components 179
  WebSpeed Messengers 179
  WebSpeed Transaction Servers 185

Configuring Batch Program 244
  conmgr.properties file 28
  Connect property 130

Connection Timeout property
  replication
    agent 54
    control agent 56

Controlling NameServer property
  AppServer 117
  MS SQL Server DataServer 87
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Messenger 181
  WebSpeed Transaction Server 188

Conversion map property 36

CPU statistic 51, 140

Create after-image archival directories property 35
Create Batch Program 244

Creating
  AppServer instances 135
  AppServer Internet Adapter instances 149
  MS SQL Server DataServer instances 92
  NameServer instances 107
  ODBC DataServer instances 69
  OpenEdge database configurations 43
  server groups 47
  Oracle DataServer instances 81
  SonicMQ Adapter instances 219
  Web Services Adapter instances 156
  WebSpeed Transaction Server instances 205

Critical agent property 56

Data service name list property (continued)
  ODBC DataServer 64
  Oracle DataServer 76

Database group 28
Database name property 30

Database property
  replication
    agent 54
    control agent 56
    server 55

Database Role property 57
Database service communication area size property 37

DataServers 61
Deactivate property 130

Debug clients property 166

Debug mode property 200

Default configuration property 30

Default cookie domain property 200

Default cookie path property 200
default.props file 167

Defer Agent Startup property 55

Deleting
  AppServer instances 141
  AppServer Internet Adapter instances 151
  MS SQL Server DataServer instances 93
  NameServer instances 107
  ODBC DataServer instances 69
  OpenEdge database configurations 52
  references 51
  server groups 52
  Oracle DataServer instances 81
  SonicMQ Adapter instances 220
  Web Services Adapter instances 173, 235
  WebSpeed Transaction Server instances 207

Deploying a REST application (.war file) 232

Direct I/O property 33

Disable client verification of SSL host name property 183

Disable SSL session cache property 42

Disable SSL session caching property 132, 183, 202

Disable SSL session reuse property 183

Disconnect property 130

E

Editing
  AppServer Internet Adapter properties 149
  AppServer properties 135
  MS SQL Server DataServer properties 92
  NameServer properties 106
  ODBC DataServer properties 69
  OpenEdge database default properties 42
  server group properties 47
  Oracle DataServer properties 81
  SonicMQ Adapter properties 219
Index

Editing (continued)
  Web Services Adapter properties 166
  WebSpeed Messenger properties 184
  WebSpeed Transaction Server properties 205
Editing Batch Program 244
Enable dynamic property updates property
  WebSpeed Transaction Server 193
Enable SSL client connections property
  AppServer 131
  WebSpeed Messenger 183
  WebSpeed Transaction Server 201
Enable SSL for remote connections property 41
Enable Web Services property 164
Enable WSA administration property 163
Enable WSDL listing retrieval property 161
Enable WSDL retrieval property 160
Environment property 105
Execution Time Limit property 201

F
Fault tolerance
  connection-level 108
  server-level 109
File upload directory property 200
Full on-line and off-line backup arguments property 57

G
Getting started 23
Group name property
  AppServer 117
  MS SQL Server DataServer 87
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Transaction Server 187

H
Hash table entries property 32
Host name property 100
Host property
  database server group 47
  replication control agent 56
HTTP error page property 166

I
Incremental backup arguments property 57
Index range size property 41
Initial number of servers to start property
  AppServer 129
  WebSpeed Transaction Server 199
Internal administration command property 184

J
Java classpath property 41
jvmargs property
  AppServer 123
  WebSpeed Transaction Server 193

K
Keep Alive property 55
Kernel time statistic 51, 141
Key alias name property 41
Key alias password property 42

L
Load balancing
  priority weight 109
Location property
  NameServer 100
  Web Services Adapter 159, 230
Location transparency 109
Lock table entries property 32
Lock table hash table size property 37
Log code page property 36
Log file threshold size property
  NameServer 103
  Web Services Adapter 163
  WebSpeed Messenger 183
Log file viewer
  Batch Program 247
Log File Viewer (stderr) 247
Log File Viewer (stdout) 247
Log filename property
  WebSpeed Messenger 182
Log files
  AppServer 142
  AppServer Internet Adapter 150
  MS SQL Server DataServer 95
  NameServer 112
  ODBC DataServer 72
  OpenEdge database 29
  Oracle DataServer 84
  SonicMQ Adapter 222
  WebSpeed Messenger 184
Log message threshold size property 162
Logging entry types property
  NameServer 103
  Web Services Adapter 163
  WebSpeed Messenger 182
Logging level property
  Web Services Adapter 162
  WebSpeed Messenger 182
Login
  Web server 158
Managed databases 28
Managing Batch Program 243
Managing database instances 29
Managing Web Services Adapter instances 225
Max servers property 33
Max users property 32
Maximum client instances property
  AppServer 121
  MS SQL Server DataServer 89
  ODBC DataServer 66
  Oracle DataServer 78
  WebSpeed Transaction Server 191
Maximum clients per server property 46
Maximum dynamic port property 46
Maximum message property 56
Maximum NameServer client port property 181
Maximum number of broker log files property
  AppServer 121
  MS SQL Server DataServer 89
  ODBC DataServer 65
  Oracle DataServer 77
  WebSpeed Transaction Server 191
Maximum number of database areas property 37
Maximum number of JTA transactions property 41
Maximum number of log files property
  NameServer 103
  Web Services Adapter 163
  WebSpeed Messenger 183
Maximum number of server log files property
  AppServer 129
  WebSpeed Transaction Server 199
Maximum port number property
  AppServer 126
  MS SQL Server DataServer 91
  ODBC DataServer 68
  Oracle DataServer 80
  WebSpeed Transaction Server 196
Maximum private database buffers per user property 33
Maximum servers property
  AppServer 129
  WebSpeed Transaction Server 199
Maximum shared-memory segment size property 39
Message buffer size property 45
Messenger executable filename property 181
Minimum clients per server property 46
Minimum dynamic port property 47
Minimum NameServer client port property 181
Minimum port number property
  AppServer 126
  MS SQL Server DataServer 91
  ODBC DataServer 68
  Oracle DataServer 80
  WebSpeed Transaction Server 196
Minimum servers property
  AppServer 129
  WebSpeed Transaction Server 199
Modifying, See Changing
Modifying configuration 244
Monitored property 42
MS SQL Server DataServers
  actions allowed on 85
  broker properties 86
  configuration overview 85
  creating instances 92
  deleting instances 93
  editing a configuration 92
  environment variables 92
  server properties 90
  starting
    automatically 93
    manually 93
  status 94
  viewing log files 95
NameServer logging level property 102
NameServers
  actions allowed on 98
  configuration overview 98
  configuring local instances 99
  configuring remote instances 105
  creating instances 107
  deleting instances 107
  editing a configuration 106
  environment variables
    setting and deleting 106
  features
    fault tolerance 108
    load balancing 108
    location transparency 109
  firewalls and UDP 111
  local and remote 99
  starting
    automatically 109
    manually 109
  status 110
  viewing log files 112
Nap interval property 37
Nap max interval property 38
Nap time increment property 38
Nap time steps between nap time property 38
Neighboring NameServers property 104
Normal startup arguments property 57
Number of open SQL cursors property 41
Number of servers property 45
ODBC DataServers
  actions allowed on 62
  broker properties 62
  configuration overview 61
  creating instances 69
  deleting instances 69
  editing a configuration 69
  environment variables 68
  server properties 66
starting
  automatically 70
  manually 70
status 70
viewing log files 72
OE Web Server
  Configuring and managing 225
OE Web Servers
  Managing 227
OpenEdge databases
  actions allowed on 28
  assigning a default configuration 43
  background writers 49
  basic properties 30
  configuration overview 28
  configuring replication-enabled 52
  conmgr.properties file 28
  creating a configuration 43
  creating a server group 47
  default configuration
    editing 42
  default properties 31
  deleting references 51
  managed 28
  managing instances 29
  process details 50, 140
  property settings
    awareness 29
    basic 30
    default 31
scripted 28
server group properties
  editing 47
shared memory systems 49
starting
  automatically 48
  manually 48
status 49
stopping 48
viewing log files 29
watchdog process 49
OpenEdge Explorer
  comparing OpenEdge Management 23
introduction 21
overview 22
OpenEdge Management
  comparing OpenEdge Explorer 23
  introduction 21
  overview 22
OpenEdge Web Services
  application server 154
  configuration overview 153, 226
  default property settings 167, 232
  default.props file 167
deploying to a WSA 168, 232
  enabling or disabling for access 174, 236
exporting from a WSA 169
importing to a WSA 170
listing deployed to a WSA 171
login 158
managing 174, 236
managing with the WSA 154, 227
run-time statistics 175, 237
session models 155
status of deployed Web service 174
undeploying 177, 240
updating
  definition and deployment information 176
  property settings 177, 239
Operating mode property
  AppServer 117
  MS SQL Server DataServer 86
  ODBC DataServer 63
  Oracle DataServer 74
  WebSpeed Transaction Server 187
Operator role
  Web Services Adapter 164
Oracle DataServers
  actions allowed on 73
  broker properties 74
  configuration overview 73
  creating instances 81
  deleting instances 81
  editing a configuration 81
  environment variables 80
  server properties 78
starting
  automatically 82
  manually 82
status 82
viewing log files 84
Other arguments property
  replication
    agent 54
    control agent 56
    server 55
Page writer max buffers property 38
Page writer queue delay property 38
Page writer queue minimum property 38
Page writer scan delay property 38
Page writer scan property 38
Password confirm property
  AppServer broker 117
  AppServer SSL 132
  WebSpeed broker 187
Password property
  AppServer 117
  MS SQL Server DataServer 87
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Transaction Server 187
Password to access the key/certificate Confirm property
  WebSpeed SSL 202
Password to access the key/certificate property
  AppServer 132
  WebSpeed Transaction Server 202
Pending connection timeout property 38
Pin shared memory segments property 39
Polling delay maximum property
  replication
    agent 54
    server 55
Polling delay minimum property
  replication
    agent 54
    server 55
Port number property
  AppServer 117
  MS SQL Server DataServer 87
  NameServer 100
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Transaction Server 187
Port property 56
Priority weight property
  AppServer 121
  MS SQL Server DataServer 90
  ODBC DataServer 66
  Oracle DataServer 78
  WebSpeed Transaction Server 191
Private key/Digital certificate alias name property
  AppServer 132
  WebSpeed Transaction Server 202
Process details
  OpenEdge database 50, 140
Process time statistic 51, 141
Program Properties 244
PROPATH property
  AppServer 126
  MS SQL Server DataServer 91
  ODBC DataServer 67
  Oracle DataServer 79
  WebSpeed Transaction Server 195
Properties
  AppServer
    agent 125
    broker 116
    messaging 132
    SSL 131
  AppServer Internet Adapter 143
  MS SQL Server DataServer
    broker 86
    server 90
  ODBC DataServer
    broker 62
    server 66
  OpenEdge database
    basic 30
    conmgr.properties file 28
    default 31
    server group 44
    setting awareness 29
  Oracle DataServer
    broker 74
    server 78
  SonicMQ Adapter
    broker 211
    server 215
    SSL 217
  Web Services Adapter 158, 230
  WebSpeed Messenger 180
  WebSpeed Transaction Server
    agent 195
    broker 186
    messaging 202
    SSL 201
Proxy host property 160, 231
Proxy password Confirm property 160
Proxy password property 160, 231
Proxy port property 160, 231
Proxy username property 160, 231
R
Register with NameServer property
  AppServer 117
  MS SQL Server DataServer 87
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Messenger 181
  WebSpeed Transaction Server 188
Registration host name property
  AppServer 118
  MS SQL Server DataServer 88
  ODBC DataServer 64
  Oracle DataServer 76
  WebSpeed Transaction Server 188
Registration mode property
  AppServer 118
Registration mode property (continued)
  MS SQL Server DataServer 88
  ODBC DataServer 64
  Oracle DataServer 76
  WebSpeed Transaction Server 188
Registration retry property
  AppServer 122
  MS SQL Server DataServer 90
  ODBC DataServer 66
  Oracle DataServer 78
  WebSpeed Transaction Server 192
Replication method property 56
Replication-enabled databases
  property settings
    agent 54
    categories 53
    control agent 56
    rules 53
    server 55
    transition 57
Reporting interval property 45
Request timeout property
  AppServer 122
  WebSpeed Transaction Server 192
Require Web Services authorization property 164
Require WSA administration authorization property 164
Require WSDL Retrieval authorization property 164
Resident size statistic 50, 140
Responsibility property 57
REST application
  Undeploying 240
  Updating a deployed REST application 239
REST applications
  SSL support 240
Restart after transition property 57
Roles
  Web Services Adapter 164

S

Schema Lock Action property 55
Scripted databases 28
Semaphore sets property 39
Server application mode property 196
Server code page property 36
Server executable file property
  AppServer 125
  MS SQL Server DataServer 91
  ODBC DataServer 67
  Oracle DataServer 79
  WebSpeed Transaction Server 195
Server log filename property
  AppServer 126
  MS SQL Server DataServer 91
  NameServer 101
  Oracle DataServer 80
  WebSpeed Transaction Server 196
Server logging entry types property
  AppServer 127
  WebSpeed Transaction Server 197
Server logging level property
  AppServer agent 127
  AppServer messaging 134
  WebSpeed agent 197
  WebSpeed messaging 204
Server port maximum property 55
Server port minimum property 55
Server startup parameters property
  AppServer 126
  MS SQL Server DataServer 91
  ODBC DataServer 67
  Oracle DataServer 79
  WebSpeed Transaction Server 195
Server startup timeout property
  AppServer 122
  WebSpeed Transaction Server 192
Server Transition property 55
Server Transition Timeout property 55
Server watch dog interval property
  AppServer 129
  WebSpeed Transaction Server 199
serverASK Activity Timeout property 215
serverASK Response Timeout property 215
ServerGroup group 28
Service name or port property 45
Session connection ID property 184
Session-Free
  AppServers 114
  Web services 155
Session-Managed
  AppServers 114
  Web services 155
Shared memory overflow size property 39
Shared memory systems 49
Shutdown property 130
SOAP-AVM gateway 154
SonicMQ Adapters
  actions allowed on 210
  broker properties
    advanced features 214
    AppService Name List 213
    controlling NameServer 212
    general 211
    logging setting 213
    owner information 212
  configuration overview 209
  creating instances 219
SonicMQ Adapters (continued)
deleting instances 220
editing a configuration 219
enabling or disabling for access 219
server properties
general 216
logging setting 216
pool range 217
SSL properties
advanced features 218
general 217
starting
manually 220
status 221
viewing log files 222
SonicMQ ServerConnect broker log filename property
AppServer 133
WebSpeed Transaction Server 203
SonicMQ ServerConnect enabled property
AppServer 133
WebSpeed Transaction Server 203
SonicMQ ServerConnect server log filename property
AppServer 133
WebSpeed Transaction Server 203
Source startup arguments property 57
source.repl.properties file 52
Spin lock retries property 40
SQL sorting memory property 41
SQL sorting on disk 41
SQL stack size property 41
SQL statement cache size property 41
SQL temporary data page size property 41
SSL session cache timeout property
AppServer 132
WebSpeed Transaction Server 202
Starting
AppServers
automatically 136
manually 135
MS SQL Server DataServers
automatically 93
manually 93
NameServers
automatically 109
manually 109
ODBC DataServers
automatically 70
manually 70
OpenEdge databases
automatically 48
manually 48
Oracle DataServers
automatically 82
manually 82
Starting (continued)
SonicMQ Adapters
manually 220
Web Services Adapters 167
WebSpeed Transaction Servers
automatically 206
manually 205
Starting Batch Program 245
Startup parameters property
AppServer 131
Startup property 130
Status
AppServer 137
deployed Web Services 174
MS SQL Server DataServer 94
NameServer 110
ODBC DataServer 70
OpenEdge database 49
Oracle DataServer 82
SonicMQ Adapter 221
Web Services Adapter
run-time 171, 233
Stopping
OpenEdge databases 48
Web Services Adapters 173
Storage object cache size property 40
Structure file containing After image area definitions property 57
Supports default service property
AppServer 118
MS SQL Server DataServer 88
ODBC DataServer 64
Oracle DataServer 76
WebSpeed Transaction Server 188
Synchronous or Asynchronous property 56
T
Table range size property 41
Target startup arguments property 57
target.repl.properties file 52
TCP/IP version property
AppServer 123
NameServer 104
OpenEdge database 46
WebSpeed Transaction Server 193
To enable administration roles property 164
Transition to Agents property 57
TXE commit lock skip limit property 40
U
ubroker.properties file 23
Unified Broker products 98
Updating, See Changing
URL property 159, 230
User MUX latches property 40
User time statistic 51, 140
Username property
  AppServer 117
  MS SQL Server DataServer 87
  ODBC DataServer 63
  Oracle DataServer 75
  WebSpeed Transaction Server 187

V

Viewing
  AppServer
    agent properties 125
    broker properties 116
    environment variables 134
    log files 142
    messaging properties 132
    SSL properties 131
    status 137
  AppServer Internet Adapter
    log files 150
  AppServer Internet Adapter properties 143
  deployed Web service
    run-time statistics 175, 237
    status 174
  MS SQL Server DataServer
    broker properties 86
    environment variables 92
    log files 95
    server properties 90
    status 94
  NameServer
    environment variables 106
    log files 112
    properties 99
    status 110
  ODBC DataServer
    broker properties 62
    environment variables 68
    log files 72
    server properties 66
    status 70
  OpenEdge database
    basic properties 30
    default properties 31
    log files 29
    process details 50, 140
    server group properties 44
    status 49
  Oracle DataServer
    broker properties 74
    environment variables 80
    log files 84
    server properties 78

Viewing (continued)
  Oracle DataServer (continued)
    status 82
  SonicMQ Adapter
    broker properties 211
    environment variables 218
    log files 222
    server properties 215
    SSL properties 217
    status 221
  Web Services Adapter
    advanced features properties 165
    location properties 159, 230
    logging setting properties 161
    properties 158, 230
    proxy server setting properties 160, 231
    security properties 163
    status run-time 171, 233
    WSDL properties 160
  WebSpeed Messenger
    log files 184
    properties 180
  WebSpeed Transaction Server
    agent properties 195
    broker properties 186
    environment variables 204
    messaging properties 202
    SSL properties 201
  Viewing OE Web Server instance statistics 233
  Viewing the status of an OE Web Server instance 233
  Virtual size statistic 50, 140

W

Watchdog process 49
Watchdog process property 33
Web services, See OpenEdge Web Services
Web Services Adapter
  Admin or Operator role 164
Web Services Adapters
  actions allowed on 156
  changing run-time properties 172, 234
  configuration overview 157
  creating instances 156
  deleting instances 173, 235
  deploying Web services 168, 232
  description 153, 226
  editing a configuration 166
  enabling or disabling Web services 174, 236
  exporting Web services 169
  importing Web services 170
  listing deployed Web services 171
  properties
    advanced features 165
    location 159, 230
Web Services Adapters (continued) properties (continued)
- logging setting 161
- proxy server setting 160, 231
- security 163
- WSDL 160
- setting defaults for Web services 167, 232
- starting 167
- status
  - run-time 171, 233
  - stopping 173
- undeploying Web services 177, 240
- updating Web services 175, 239
- viewing Web service statistics 172
- Web service run-time statistics 175, 237

Web Services Protocol Value property 159

WebSpeed
- additional management 208

WebSpeed Messengers
- configuration overview 179
- editing a configuration 184
- properties
  - advanced features 184
  - controlling NameServer 181
  - general 181
  - logging setting 182
  - SSL 183
- viewing log files 184

WebSpeed transaction server host property 182
WebSpeed transaction server port property 182

WebSpeed Transaction Servers
- actions allowed on 186
- additional management 208
- agent properties
  - advanced features 200
  - general 195
  - logging setting 196
  - pool range 199

WebSpeed Transaction Servers (continued)
- agents
  - adding or trimming 206
- broker properties
  - advanced features 191
  - AppService Name List 188
  - controlling NameServer 188
  - general 187
  - owner information 187
- configuration overview 185
- creating instances 205
- deleting instances 207
- editing a configuration 205
- environment variables 204
- messaging properties 202–203
- SSL properties
  - advanced features 202
  - general 201
  - starting
    - automatically 206
    - manually 205

Weighted CPU statistic 51, 140
Windows Event Level property 37

Working directory property
- AppServer 117
- MS SQL Server DataServer 86
- NameServer 101
- ODBC DataServer 63
- Oracle DataServer 74
- Web Services Adapter 161
- WebSpeed Messenger 181
- WebSpeed Transaction Server 187

WSA, See Web Services Adapter

WSA administration authentication property 164
WSA administration SOAP action property 165
WSDL inaccessible page property 160
WSDL listing page property 161