Progress®



Upgrade to OpenEdge 12

OpenEdge®

Table of Contents

Copyright....5

Preface....7

Using ABL documentation....7

References to ABL compiler and run-time features....7

References to ABL data types....8

Typographical conventions....8

Examples of syntax descriptions....9

Long syntax descriptions split across lines....11

Complex syntax descriptions with both required and optional elements....11

OpenEdge messages....11

Obtaining more information about OpenEdge messages....12

Chapter 1: Overview of Upgrading to OpenEdge 12.0....13

Chapter 2: Converting an OpenEdge Release 11 Database to OpenEdge Release 12....15

Chapter 3: Migrate OpenEdge Management from Jetty to Progress Application Server....17

Changes to OpenEdge Management....18

Windows service....18

File system layout....19

Configuration changes....23

Log configuration on PASOE....29

Chapter 4: Upgrade ABL Service Projects in a PAS for OpenEdge Instance....31

Upgrade ABL Service projects....31

Upgrade the PAS for OpenEdge instance....32

Chapter 5: Progress Developer Studio for OpenEdge Upgrade Tasks...35

Upgrade an ABL WebApp Project....35

Package an ABL WebApp ANT project....36

Chapter 6: Upgrade Security to OpenEdge 12.....37
Updates to password encryption with GENSPRINGPWD.....37
Updates to the cryptography used to secure Client-Principal objects.....38
PAS for OpenEdge and OpenEdge Authentication Gateway: Domain configuration.....38
Secure communications in OpenEdge Management.....39
Migrate OpenEdge Authentication Gateway configuration.....39

Chapter 7: Upgrade ABL Applications.....41

Chapter 8: Migration notes.....43

Copyright

© **2019 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.

Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Deliver More Than Expected, Icenium, Kendo UI, Kinvey, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, SequeLink, Sitefinity (and Design), Sitefinity, 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. Analytics360, AppServer, BusinessEdge, DataDirect Autonomous REST Connector, DataDirect Spy, SupportLink, DevCraft, Fiddler, JustAssembly, JustDecompile, JustMock, Kinvey Chat, NativeChat, NativeScript Sidekick, OpenAccess, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries 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.

February 2019

Last updated with new content: Release 12.0

Preface

For details, see the following topics:

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

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.

For the latest documentation updates go to https://docs.progress.com.

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:

Convention	Description
Bold	Bold typeface indicates commands or characters the user types, provides emphasis, or the names of user interface elements.
Italic	Italic typeface indicates the title of a document, or signifies new terms.
SMALL, BOLD CAPITAL LETTERS	Small, bold capital letters indicate OpenEdge key functions and generic keyboard keys; for example, **GET** and **CTRL**.
KEY1+KEY2	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**.
KEY1 KEY2	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**.
Syntax:	
`Fixed width`	A fixed-width font is used in syntax, code examples, system output, and file names.

Convention	Description
Fixed-width italics	Fixed-width italics indicate variables in syntax.
Fixed-width bold	Fixed-width bold italic indicates variables in syntax with special emphasis.
UPPERCASE fixed width	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.
Period (.) or colon (:)	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.
[ ]	Large brackets indicate the items within them are optional.
[ ]	Small brackets are part of ABL.
{ }	Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.
{ }	Small braces are part of ABL. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.
|	A vertical bar indicates a choice.
. . .	Ellipses indicate repetition: you can choose one or more of the preceding items.

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

```
ASSIGN   { [ FRAME frame ] { field [ = expression ] }
           [ WHEN expression ] } ...
       | { record [ EXCEPT field ... ] }
```

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**.

1

Overview of Upgrading to OpenEdge 12.0

When you upgrade an OpenEdge Release 11 environment to OpenEdge 12.0, there are tasks that you must perform to ensure that your applications and operating environment will function properly. The tasks that are required depend on the specific OpenEdge components that you are upgrading.

The general tasks that you must perform are:

1. Export the source installation configuration of your existing OpenEdge 11 software. See configutil in *Administer Progress Application Server for OpenEdge*.
2. Note the deprecated and de-supported features and functionality listed in *OpenEdge 12 Platform & Product Availability Guide*, available on https://docs.progress.com.
3. Be sure to read Before You Install in *Install OpenEdge* before you install OpenEdge 12, and also *OpenEdge 12 Release Notes*.
4. If your database is enabled for Transparent Database Encryption (TDE) or uses the RC4 encryption algorithm for any of its tables or indexes, you must change the TDE policy for each RC4-encrypted table or index to use an encryption algorithm that is supported in OpenEdge 12. You must perform this update **before** you convert your database to version 12, as described in Converting an OpenEdge Release 11 Database to OpenEdge Release 12 on page 15.
5. Convert your database to version 12, as described in Converting an OpenEdge Release 11 Database to OpenEdge Release 12 on page 15.
6. If you use OpenEdge Management, note the changes described in Migrate OpenEdge Management from Jetty to Progress Application Server on page 17.
7. If you have an ABL WebApp developed using Progress Developer Studio for OpenEdge, complete the tasks described in Upgrade ABL Service Projects in a PAS for OpenEdge Instance on page 31 and Progress Developer Studio for OpenEdge Upgrade Tasks on page 35.
8. Complete the tasks described in Upgrade Security to OpenEdge 12 on page 37 to upgrade your application environment security to Release 12.0.

9. If you have ABL applications, you may need to perform the upgrade tasks described in Upgrade ABL Applications on page 41.
10. Be sure to read the information described in Migration notes on page 43.

2

Converting an OpenEdge Release 11 Database to OpenEdge Release 12

The PROUTIL CONV1112 utility converts the schema of an OpenEdge Release 11 database to OpenEdge Release 12.

To convert the schema of an OpenEdge Release 11 database:

1. Back up the database using OpenEdge Release 11.

   Caution: There is always a chance that your schema could become corrupt during conversion. If the conversion fails, your database cannot be recovered. If this happens, you must restore the backup copy of your database and begin the conversion again.

2. Verify that the database is not in use.
3. If your OpenEdge Release 11 database is enabled for Transparent Database Encryption (TDE) or has area, table, or index data objects currently encrypted with RC4, you must complete the following steps before running the PROUTIL CONV1112 utility. If your database is not TDE-enabled, or is not using RC4 for any of its areas, tables, or indexes, you may skip to step 4.

a) Modify your OpenEdge Release 11 database to change the TDE policy for each RC4-encrypted table or index to use an encryption algorithm that is supported in OpenEdge 12, such as AES-128. For details, see Changing the cipher of an encryption policy with PROUTIL EPOLICY in *Introduction to Security and Auditing*.

b) Update the encrypted data that is stored in the database. For an example, see EPOLICY MANAGE UPDATE example in *Introduction to Security and Auditing* (an example of updating is described in steps 4 and 5).

c) Make sure that the updated tables and indexes show that all of the stored data is encrypted with the new algorithm.

4. Truncate your before-image file with Release 11 PROUTIL TRUNCATE BI. You cannot truncate the before-image file of your Release 11 database with Release 12.
5. Disable after-imaging, if enabled, with Release 11 RFUTIL AIMAGE END.

   If you have OpenEdge Replication enabled, you must disable it before you can disable after-imaging. For information on Replication, see *Use Database Replication*.

6. Disable two-phase commit, if enabled, with Release 11 PROUTIL 2PHASE END.
7. Verify your Release 11 backup, and then install OpenEdge Release 12, if you have not already done so, following the instructions in *Install OpenEdge*.
8. Run PROUTIL CONV1112 to convert your database.

   Use the PROUTIL CONV1112 command, as shown:

   ```
   proutil db-name -C conv1112
   ```

   CONV1112 performs extensive checks of your database during a pre-scan process. If the pre-scan encounters any issues that would prevent a successful conversion, the issues are logged and the convert utility exits without making any changes to the database (it is still a valid Release 11 database).

9. After you have successfully converted your database, back up your OpenEdge Release 12 database using Release 12 PROBKUP.

   - You should back up your Release 12 database in case it is damaged due to some failure. If you have only a Release 11 backup of your database, then you will need to go through the conversion process again.
   - Changes to large file support in OpenEdge Release 12.0 will impact databases created using CONV1112. See Database creation and large file support for more information.

3

Migrate OpenEdge Management from Jetty to Progress Application Server

Before OpenEdge 12.0, the OpenEdge Management web application was hosted on the Jetty Web server. The Jetty Web server was, in turn, hosted as a plugin to the AdminServer.

In OpenEdge 12.0, OpenEdge Management is hosted on a PAS for OpenEdge instance. PAS for OpenEdge is a Tomcat Web server with Progress-specific configuration settings.

Benefits

Hosting OpenEdge Management on PAS for OpenEdge provides the following benefits:

- For enhanced stability, security, and extensibility, OpenEdge Management is now a self-hosted web application.
- OpenEdge Management now runs independently of the AdminServer.
- You can manage multiple local AdminServer instances in addition to the remote AdminServer instances.

Key changes

For Release 12.0, the following key changes have been made to host OpenEdge Management on PAS for OpenEdge:

- This is an OpenEdge 12.0 release only change
- Jetty is no longer relevant
- OpenEdge Management and OpenEdge Explorer run on a PAS for OpenEdge instance
- OpenEdge Management is a separate process from the AdminServer process
- Changes to the configuration file system layout are introduced
- TCMAN works as expected.
- The Fathom command line tool is revised to wrap TCMAN for many operations
- OpenEdge Management shares some third-party libraries with PAS for OpenEdge, and many are replaced
- All Tomcat configuration is located in the `<OEMgmt>/conf` directory.

For details, see the following topics:

- Changes to OpenEdge Management

Changes to OpenEdge Management

The following is a summary of the changes made in OpenEdge Management in Release 12.0:

- Changes to the Windows service
- Configuration changes that include:
  - Consolidation of configuration
  - File system layout
  - Fathom command line
  - AdminServer
- Logging configuration changes that include:
  - Exception handling
  - Fathom log file changes
  - Logging configuration changes for Tomcat and OpenEdge Management

Windows service

On Windows systems, OpenEdge Management is registered as a Windows service and is set to autostart by default by the installation program.

Note the following:

- Use the Windows Services Manager (`services.msc`) to determine whether OpenEdge Management is registered as a Windows service.
- Use `tcman service <name> unregister` to unregister fathom as Windows service, and make changes to system properties or environment variables in the `catalina.properties` and `appserver.properties` files as needed.
- After making changes to the system properties or environment variables, re-register fathom as a Windows service.
- If you unregister a Windows service, then group permissions apply for start and stop operations.
- The fathom CLI can identify whether the Tomcat Web server is a Windows service.

Note: If OpenEdge Management is registered as a Windows service, then make sure that you have administrator privileges to start and stop OpenEdge Management.

File system layout

In Release 12.0, the OpenEdge Management installation directory by default is colocated with the PAS for OpenEdge installation directory, and user-created files such as reports and logs continue to be a part of the OpenEdge Management working directory.

The following table lists and summarizes the files in the OpenEdge Management installation directory, and identifies any change in their new location in OpenEdge 12.0. Note that some of these files are created at run time.

Table 1:

Old file location	New file location	Description
<OEMgmt>/db/100/db.zip	No change	The fathom database backup.
<OEMgmt>/db/fathom.db	No change	The fathom trend database. This file is configured on first use. You can place it anywhere, but the default location is inside the instance.
<OEMgmt>/etc/fathomRealm.properties	<OEMgmt>/conf/fathomRealm.properties	
<OEMgmt>/etc/fathomRealm.properties.orig	<OEMgmt>/conf/fathomRealm.properties.orig	
• <OEMgmt>/config/fathom.delete • <OEMgmt>/config/fathom.update • <OEMgmt>/config/fathom.init.params	Removed	
<OEMgmt>/config/Ports.xml	<OEMgmt>/conf/Ports.xml	
<OEMgmt>/config/PSC-FM-MIB.txt	<OEMgmt>/ conf/PSC-FM-MIB.txt	Used with the SNMP adapter.
<OEMgmt>/config/tailor.xml	<OEMgmt>/ conf/tailor.xml	
<OEMgmt>/config/configdb	<OEMgmt>/work/configdb	The OpenEdge Management configuration database.
<OEMgmt>/config/fathomweb.xml	Removed	The Jetty Web server configuration file, which is not available in Release 12.
<OEMgmt>/etc/	<OEMgmt>/conf	Configuration files previously in the /etc and /config directories are now merged with the configuration files in /conf directory.
<OEMgmt>/etc/fathomRealm.policy	OEMgmt>/conf/fathomRealm.policy	Contains roles and permission definitions.
<OEMgmt>/etc/fathomRealm.policy.orig	<OEMgmt>/conf/fathomRealm.policy.orig	The baseline set of permission definitions. It is copied at run time if the fathomRealm.policy file does not exist, but is never modified.
<OEMgmt>/etc/demokeystore	Removed	The Tomcat keystore configuration. This file is no longer used in Release 12.0.

Old file location	New file location	Description
<OEMgmt>/jars	<OEMgmt>/jars	Any .jar files needed by the fathom web application at run time is placed in this directory instead of the WEB-INF/lib directory. OpenEdge Management supports third-party plugins, and these will need access to shared functionality that would not be accessible if all the .jar files are in the web applications' WEB-INF/lib directory.
<OEMgmt>/jspwork	Removed	Tomcat compiles .jsp files on-the-fly to the <OEMgmt>/temp directory.
<OEMgmt>/samples/samplewebapp.war	No change	A sample web application.
<OEMgmt>/src	No change	
<OEMgmt>/templates	No change	Contains bootstrap configuration for rules, reports, schedules, and jobs. These are imported to the configuration database on first startup.
<OEMgmt>/web	<OEMgmt>/webapps/ROOT.war	The fathom Web application that is hosted in Jetty in Release 11. In Release 12.0 this file is changed to fathom.war and is located in the instance as the ROOT.war web application.
<OEMWORK>/logs	<OEMgmt>/logs	The version 11 log file written to the OpenEdge working directory. In Release 12 PAS for OpenEdge and fathom logs are stored in the new location.
<OEMWORK>/reports	No change	
<OEMWORK>/files.dir	<OEMgmt>/files.dir	Temp files used by OpenEdge Management to track jobs.
<OEMWORK>/cachedata	<OEMgmt>/work/cachedb	The graph cache database. It is now be created inside the instance working directory by default.

Old file location	New file location	Description
<OEMWORK>/snmp	<OEMgmt>/work/snmp	The SNMP adapter bootstrap configuration directory. This directory exists only after the SNMP adapter has been started at least once. It stores startup counter information that is required by SNMP implementations. This file is safe to delete at any point.
	<OEMgmt>/webapps	The Tomcat default webapps directory for hosting the fathom.war file and other third-party libraries.
<OEMgmt>/config	<OEMgmt>/conf	The default Tomcat configuration directory. Any modifiable configuration files are stored here. This is part of the default Tomcat instance. Anything previously within the existing <OEMgmt>/config directory is moved to this location.
	<OEMgmt>/conf/adminserver.properties <OEMgmt>/conf/default.properties	Files that contain a list of system properties that are set at bootstrap time.
	<OEMgmt>/conf/jvm.properties	
	<OEMgmt>/temp	Holds temp files, specifically the PAS for OpenEdge .pid file.
<DLC>/properties/fathom.properties	<OEMgmt>/conf/fathom.properties	
<WRKDIR>/admserv.log <WRKDIR>/ads0.exp	No change	OpenEdge Management log file messages are no longer stored in these files. Instead, the fathom web application is configured to write logs in a separate configuration file that is stored in the <oemgmt>/logs directory.
	<OEMgmt>/logs/fathom.log	The fathom log file name includes a date in the format fathom.*date*.log.
	<OEMgmt>/bin	Directory that includes the PAS for OpenEdge files such as startup, shutdown, tcman and other scripts.

Configuration changes

This topic describes the changes made to configuration files after you upgrade to OpenEdge Release 12.0, and also identifies changes that you are required to make.

System properties

Each PAS for OpenEdge instance has a `conf/appserver.properties` file in which you make updates to system properties. Several of the properties that were previously defined in the `fathom.init.params` file are moved into this `conf/appserver.properties` file as-is.

Example `appserver.properties` file

Some properties are new in OpenEdge 12.0 and replace values that were previously stored in the `fathom.init.params` file, as follows:

```
#Fri, 11 Jan 2019 18:03:26 +0530
# The appserver.properties file contains name-value Java properites used by
# the Progress Application Server (PAS) and web applications.
# Each non blank and non comment line will be defined as a java system property
# by prepending a '-D' and passed on the command line to the JVM via the
# JAVA_OPTS environment variable.
#
# Comment lines may be used by placing a pound signe (#) in column zero.  Trailing
# comments are not supported.
# You may use shell environment variable substitution using the form ${env-var-name}

# The release version of the Progress Application Server version
psc.as.version=4.0.0

# The type of PAS installation {standard|instance}
psc.as.type=instance

# The PAS installation OS type {windows|unix}
psc.as.os=windows

# The PAS installation personality type {appserver|tokenserver|fathom}
psc.as.personality=fathom


# windows service name used in windows registry for the service name
psc.as.winsvcname=cat


# Optional PAS instance name for tcman admin tools.  If not specified
# the instance name is the CATALINA_BASE's directory name "home"
psc.as.alias=ho

# Optional directory path of a PAS instance's parent
psc.as.parent=C:/oemgmt

# The PAS server & instances security policy {default|developer|production}
psc.as.security.model=production

# Indicator {true|false} of Windows OS service registration
psc.as.service=true

#OpenEdge installation directory needed for classic Adminserver libraries
Install.Dir=C:/dlc

#OpenEdge installation directory needed for PAS libraries
psc.as.oe.dlc=C:/dlc

#Fathom configuration file location
psc.oem.fathom.properties=${CATALINA_BASE}/conf/fathom.properties

#Fathom log file directory
psc.oem.logs=${CATALINA_BASE}/logs

#Fathom SNMP adapter state directory
psc.oem.snmp=${CATALINA_BASE}/work/snmp

#Fathom installation directory
psc.oem.install=${CATALINA_BASE}

#Fathom log file
psc.log.default=${CATALINA_BASE}/logs/fathom.log

#Fathom exception log file
psc.log.exceptions=${CATALINA_BASE}/logs/exceptions.log

#logging level for Fathom messages
LogLevel=3

#Local directory for Fathom configuration database
```

```
# configuration database will be created in a subdirectory of the specified directory
psc.oem.configdb.dir=${CATALINA_BASE}/work/

#Local directory for Fathom cache database
# graph cache database will be created in a subdirectory of the specified directory
psc.oem.activitydb.dir=${CATALINA_BASE}/work/

#User file location for Fathom logins
psc.oem.realm.properties=${CATALINA_BASE}/conf/fathomRealm.properties

#Permission policy file for Fathom user roles
psc.oem.realm.policy=${CATALINA_BASE}/conf/fathomRealm.policy

#Default location for Fathom trend database
psc.oem.fathom.trend=${CATALINA_BASE}/work/fathomdb/fathom.db

#OrientDB tuning parameters for cache and configuration database
storage.record.lockTimeout=75000

# max segment size for write ahead log (BI file equivalent) for cache and configuration
 databases
storage.wal.maxSegmentSize=32

# max size in MB of write ahead log for cache and configuration databases (value applies
 separately to each)
storage.wal.maxSize=256

# amount of memory in MB to allocate for cache and configuration databases
storage.diskCache.bufferSize=384

# controls memory allocation and pooling for lock manager pool
environment.lockManager.concurrency.level=8

# controls the number of files created per cluster for cache and configuration databases
class.minimumClusters=1

# Disable directIO access.  OrientDB use of Direct IO on Linux fails if the file system
 is on an NFS mount. This disables
# direct IO which is a bit sloewr, but prevents the file open problems.  This can safely
 be enabled
# on Windows, and non-NFS mounted drives on *nix.
storage.wal.allowDirectIO=false

#OpenEdge Management report generation directory. If not provided this uses
$OEMWRKDIR/reports directory
#You may override the report directory by specifying an alternate value here
#psc.oem.reports=

# ensure snmp4j sends logging information to log4j
snmp4j.LogFactory=org.snmp4j.log.Log4jLogFactory

# Configure SLF4J compatible logger for OEM
psc.log.factory=com.progress.fathom.logging.LogSystemBridgeToSlf4JFactory

# Logback configuration file
logback.configurationFile=${CATALINA_BASE}/conf/logging.xml
```

jvm.properties

The following is a sample `jvm.properties` file:

```
# This file contains a list of JVM command line options.
# You may insert any command line option, including -D.
# Each line should contain a single option to facilitate
# easier automated administration tools.
#
-Dfile.encoding=UTF-8
-server
-Xms512m
-Xmx1024m
-XX:NewSize=64m
-XX:MaxNewSize=64m
-XX:+DisableExplicitGC
-XX:MaxDirectMemorySize=2g
-Dorg.apache.coyote.USE_CUSTOM_STATUS_MSG_IN_HEADER=false
-Dorg.apache.catalina.connector.CoyoteAdapter.ALLOW_BACKSLASH=false
-Dorg.apache.tomcat.util.buf.UDecoder.ALLOW_ENCODED_SLASH=false
-Dorg.apache.catalina.connector.RECYCLE_FACADES=false
-Dorg.apache.catalina.STRICT_SERVLET_COMPLIANCE=true
-Dorg.apache.tomcat.util.http.ServerCookie.FWD_SLASH_IS_SEPARATOR=false
-Djava.awt.headless=true

-d64
```

Note: Adjust the JVM maximum direct memory size available to the JVM. Set this value to the maximum system available RAM.

```
# Adjust JVM maximum direct memory size available to the JVM. Should be set to maximum
system available RAM

-XX:MaxDirectMemorySize=4G
```

catalina.properties

The following is an example of the updated `catalina.properties` file:

```
# Allow for changes to HTTP request validation
# WARNING: Using this option will expose the server to CVE-2016-6816
#tomcat.util.http.parser.HttpParser.requestTargetAllow=|

# port #s used by server.xml
psc.as.http.port=9090
psc.as.https.port=9443
psc.as.ajp13.port=9009
psc.as.shut.port=-1
psc.as.shut.pwd=SHUTDOWN

# Optional places used by server.xml
psc.as.webappdir=C:/oemgmt/webapps
# Security settings used by server.xml
psc.as.autodeploy=false
psc.as.unpackwars=true
```

To support OpenEdge Management upgrade from 11.7.x to 12.0, the HTTP and HTTPS ports are now enabled by default. To start OpenEdge Management, the default HTTP port is 9090 and the default HTTPS port is 9443. You can configure the HTTPS connector for the OpenEdge Management Web server from the Tomcat configuration files.

Fathom command line

The `fathom-restart`, `fathom-start`, `fathom-stop`, and `fathom-query` command-line options now use TCMAN commands. The new `fathom-restart` command-line option restarts the underlying PAS for OpenEdge instance.

Note: The `proadsv -start` and `-stop` commands may no longer be used to start and stop fathom.

---

glue and unglue scripts

The `glue` and `unglue` scripts are removed in OpenEdge Management Release 12.0.

fathom.xml

The `fathom.xml` file imported from a previous OpenEdge version is not supported in OpenEdge 12.0. Therefore, Progress recommends that you remove resources such as Appserver, ODBC Dataserver, Webspeed, WSA, AIA and REST from the `fathom.xml` file prior to upgrading to OpenEdge 12. If you do not remove these resources prior to upgrading OpenEdge to Release 12.0, you must delete these resources after you import the `fathom.xml` file.

Note: The `fathomcli.jar` file is new in OpenEdge 12.0 and is used to run the fathom command line operations. This file is located in the `<OEMgmt>/jars` directory.

---

fathom.properties

In OpenEdge 12.0, the `fathom.properties` file is moved from the `<DLC>/properties` directory to the `<OEMgmt>/conf` directory. In addition, some of the properties defined in this file are not used in Release 12.0. Some properties may still be viewable, but you must update the location in which they are stored if you want to retain them.

The following table lists the properties that are affected by the transition to Tomcat. Any properties not listed in this table remain unchanged in Release 12.0.

Table 2: Property changes

Property Name	Description
WebServer.HttpEnabled	This property remains in the user interface, and you can set it by using TCMAN to enable and disable the HTTP connector.
WebServer.HttpPort	This property remains in the user interface and is stored in the `<instance>/conf/catalina.properties` file as the*psc.as.http.port* property.
WebServer.HttpsEnabled	This property remains in the user interface, and you can set it using TCMAN to enable and disable the HTTP connector.
WebServer.HttpsPort	This property remains in the user interface and is stored in the `<instance>/conf/catalina.properties` file as the *psc.as.https.port* property.
SSLIdentityAlias	This property remains in the user interface and is stored in the `<instance>/conf/catalina.properties` file as the *psc.as.https.keyalias* property.
SSLIdentityAliasPassphrase	This property remains in the user interface and is stored in the `<instance>/conf/catalina.properties` file as the *psc.as.https.keypass* property.
SSLIdentityKeystorePathname	

Remote trending

Remote trending is no longer supported in OpenEdge 12.0. If you have been using remote trending in an earlier release of OpenEdge, then you must configure a local trend database and avoid storing trending data in a separate OpenEdge Management installation.

Log configuration on PASOE

Logging in PASOE

In Release 12.0, OpenEdge Management is a web application that runs in a PAS for OpenEdge instance. As a result, OpenEdge Management adopts the logging mechanism that is already used by existing Web applications such as ROOT. The logging provider for OpenEdge Web apps is logback, which is a direct implementation of slf4j without the overhead of a JUL implementation.

OpenEdge Management logging in 12.0

In previous releases of OpenEdge, OpenEdge Management and AdminServer logging used the ProLog implementation, whereby OpenEdge Management appended its logs to the AdminServer log file, `<DLC_WRK>\admserv.log`. Any exceptions were logged separately in the `<DLC_WRK>\ads0.exp` file. The log level was configured through JVM arguments.

In Release 12.0, OpenEdge Management runs in a PAS for OpenEdge instance as a web application. Consequently, the logging configuration is stored in the `<OEMgmt>/conf/logging.xml` file.

Note: Logs generated by Reports and Jobs are unchanged in Release 12.0.

The following is an example logback configuration, which is stored in the `logging.xml` file.

Note: This example is based upon a simple logback configuration for OpenEdge Management. PAS for OpenEdge uses a more complex logger configuration that provides logging separation.

The following are common settings and applicable to all logging approaches:

```
<configuration debug="true" scan="true" scanPeriod="5 seconds">
  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <append>true</append>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
     <fileNamePattern>${catalina.base}/logs/fathom.%d{yyyy-MM-dd}.log</fileNamePattern>

    </rollingPolicy>
    <encoder>
      <pattern>%date [%thread] %-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>

  <root level="debug">
    <appender-ref ref="FILE" />
  </root>
</configuration>
```

Similar to ProLog, logback supports changes to the logging configuration dynamically. This behavior can be configured by setting the scan attribute to true and specifying the scanPeriod, as follows:

```
<configuration debug="true" scan="true" scanPeriod="5 seconds">
```

Unlike ProLog, logback creates and writes logs to multiple files by appending the date to each file name. As with other PAS for OpenEdge Web apps, OpenEdge Management creates the `fathom.date.log` file in the `<OEM_PAS_INSTANCE>\logs` directory, as follows:

```
<fileNamePattern>${catalina.base}/logs/fathom.%d{yyyy-MM-dd}.log</fileNamePattern>
```

By default, OpenEdge Management sets the daily rollover for its log files (as does PAS for OpenEdge).

The log level of the fathom log file can be set at root level (unless multiple loggers are specified). Unlike the AdminServer log, which used numbers to indicate severity, the severity is set to one of the readable formats ERROR, WARN, INFO, DEBUG, TRACE. For example:

```
<root level="debug">
```

4

Upgrade ABL Service Projects in a PAS for OpenEdge Instance

Before you upgrade a PAS for OpenEdge instance and its deployed applications from OpenEdge Release 11.7 to 12.0, be sure to create a backup your existing PAS for OpenEdge instance and its deployed instances. You can do this by creating a zip archive of the root directory of the OpenEdge 11.7 instance. And if you are not using a source code control system, then Progress also recommends that you also create a backup of the entire project space.

After your backup is complete, upgrade all of your ABL Service projects that are created in Progress Developer Studio, and then create a new PAS for OpenEdge Release 12.0 instance. For details, complete the steps described in the following topics:

- Upgrade ABL Service projects on page 31
- Upgrade the PAS for OpenEdge instance on page 32

For details, see the following topics:

- Upgrade ABL Service projects
- Upgrade the PAS for OpenEdge instance

Upgrade ABL Service projects

To upgrade to your ABL Service projects to OpenEdge Release 12.0, use the newly-installed Progress Developer Studio for OpenEdge 12.0 to open the workspaces that contain those projects. For each project, complete the following steps:

1. Copy the `PASOEContent/WEB-INF/logging.xml` file into the `PASOEContent/WEB-INF/backup` folder.
2. Copy the `%dlc%/servlets/oeabl/WEB-INF/logging.xml` file into `PASOEContent/WEB-INF` folder. Note that this step overwrites the older `logging.xml` file.
3. Copy the `PASOEContent/WEB-INF/oeablSecurity.properties` file into the `PASOEContent/WEB-INF/backup` folder.
4. Copy the `%dlc%/servlets/oeabl/WEB-INF/oeablSecurity.properties` file into the `PASOEContent/WEB-INF` folder.
5. Merge any changes from the `PASOEContent/WEB-INF/backup/oeablSecurity.properties` file into the `PASOEContent/WEB-INF/oeablSecurity.properties` file.
6. Copy the `PASOEContent/WEB-INF/oauth2ResSvcClients.cfg` file into the `PASOEContent/WEB-INF/backup` folder.
7. Copy the `%dlc%/servlets/oeabl/WEB-INF/oauth2ResSvcClients.cfg` file into `PASOEContent/WEB-INF` folder. Note that this step overwrites the older `oauth2ResSvcClients.cfg` file.
8. Merge any changes from the `PASOEContent/WEB-INF/backup/oeablSecurity.properties` file into the `PASOEContent/WEB-INF/oauth2ResSvcClients.cfg` file.
9. Copy the `%dlc%/servlets/oeabl/WEB-INF/oeablSecurity.xml` file into the `PASOEContent/WEB-INF` folder.
10. Copy the `%dlc%/servlets/oeabl/WEB-INF/spring` folder and its contents into the `PASOEContent/WEB-INF/spring` folder.
11. Copy the `%dlc%/servlets/oeabl/WEB-INF/web.xml` file into the `PASOEContent/WEB-INF` folder.
12. Copy the `%dlc%/servlets/oeabl/WEB-INF/home` folder and its contents into the `PASOEContent/WEB-INF` folder.
13. Copy the `%dlc%/servlets/oeabl/static/common*` files into the `PASOEContent/static` folder.
14. Copy the contents of the `%dlc%/servlets/oeabl/static/images` folder into the `PASOEContent/static/images` folder.
15. Create new PAS for OpenEdge instances that are based on OpenEdge Release 12.0, as described in Upgrade the PAS for OpenEdge instance on page 32.
16. Deploy the ABL Service projects as PAS for OpenEdge Web Apps in the newly-created PAS for OpenEdge instances.

Upgrade the PAS for OpenEdge instance

The best practice for upgrading a PAS for OpenEdge instance is to create a new instance that is based on OpenEdge Release 12.0, and to merge changed files into that new instance. You then take the old instance out of service. Note that if the new instance is configured with the same port number as the old instance, then you must stop the old instance before you are able to start the new one.

To upgrade a PAS for OpenEdge instance, complete the following steps:

1. Create a new PAS for OpenEdge instance by entering the following command:

```
pasman create
```

This command creates a default ABL application that uses the ROOT Web App.

2. Deploy upgraded Web Apps by entering the following command. If you are not using the PASOE instance name as the default ABL application name, then make sure that you specify the ABL application name correctly.

   ```
   tcman deploy
   ```

3. Copy files from the old PAS for OpenEdge instance to the new instance, as follows:

   a. Copy any files and folders in all of the `ablapps/`*`abl-app-name`* directories, excluding the `conf/oeablSecurity.properties` file.

   b. Merge properties that are specified in the *`/path-to-old-instance`*`/ablapps/`*`abl-app-name`*`/conf/oeablSecurity.properties` file into the *`/path-to-new-instance`*`/ablapps/`*`abl-app-name`*`/conf/oeablSecurity.properties` file.

   c. Copy ABL application lifecycle scripts from the *`/path-to-old-instance`*`/bin` directory to the *`/path-to-the-new-instance`*`/bin` directory. This includes the following:

   - *`abl-app-name`*`_setup.{bat|sh}`
   - *`abl-app-name`*`_startup.{bat|sh}`
   - *`abl-app-name`*`_started.{bat|sh}`
   - *`abl-app-name`*`_shutdown.{bat|sh}`

   d. Copy product OS environment variable setup scripts from the *`/path-to-old-instance`*`/bin` directory to the *`/path-to-the-new-instance`*`/bin` directory. This includes the *`your-name`*`_setenv.{bat|sh}` script.

   e. Copy or merge any other product shell scripts in the *`/path-to-old-instance`*`/bin` directory to the *`/path-to-new-instance`*`/bin` directory.

   f. Copy any source or rcode file from the *`/path-to-old-instance`* directory to the *`/path-to-new-instance`* directory.

   g. Copy any other installation- or application-specific folders from the *`/path-to-old-instance`* directory to the *`/path-to-new-instance`* directory.

4. Merge properties from the older instance into the new, as follows:

   a. For OpenEdge-specific properties, use the following command:

   ```
   /path-to-new-instance/bin/oeprop -f
   /path-to-old-instance/conf/openedge.properties
   ```

   b. Merge any other properties file using a text merge tool.

5. Merge PAS for OpenEdge features from the old instance to the new, as follows:

   a. Obtain the feature configuration by entering the following command:

   ```
   /path-to-old-instance/bin/tcman.{bat|sh} feature
   ```

   b. Make any changes by entering the following command:

   ```
   /path-to-new-instance/bin/tcman.{bat|sh} feature feature-name={on|off}
   ```

6. Make sure the ROOT Web App is not in use, and then remove it by entering the following command:

```
tcman undeploy ROOT
```

A best practice is to replace the ROOT Web App with an empty or blank Java Web App that satisfies Tomcat's default web application context lookup.

7. Test the PAS for OpenEdge instance before adding it to your production environment.

5

Progress Developer Studio for OpenEdge Upgrade Tasks

This section contains the steps to upgrade the ABL Web App projects and Spring Security files to OpenEdge 12.0.

For details, see the following topics:

- Upgrade an ABL WebApp Project
- Package an ABL WebApp ANT project

Upgrade an ABL WebApp Project

If you have imported an ABL WebApp project from OpenEdge 11.6 and 11.7 to the current release in the Progress Developer Studio for OpenEdge, you must upgrade the Spring Security files.

To upgrade the Spring Security files, follow the below steps:

1. Right-click the ABL WebApp project and select **Progress OpenEdge** > **Migrate Spring Security Files**. The **Migrate Spring Security Files** dialog box also opens.

   Note: The dialog box also opens when you open an old workspace or import an old project to 12.0 workspace that contains ABL WebApp projects.

2. Click **OK**.

Note: If you click **Cancel**, you must later follow the previous step to upgrade.

The Spring Security files in the current `WEB-INF` folder in the project directory are moved to the newly created backup folder. The migrated Spring Security files are placed in the `WEB-INF` folder.

Package an ABL WebApp ANT project

To package an ABL WebApp ANT project in release 12.0, you need to update the existing `build.xml` file. Add the following code in `build.xml`.

```
<pathelement location="${dlc.java.dir}/ext/jettison-1.2.jar"/>
<pathelement location="${dlc.java.dir}/ext/commons-logging-1.2.jar"/>
<pathelement location="${dlc.java.dir}/ext/xmlschema-core-2.2.1.jar"/>
```

For more details, see Package an ABL WebApp ANT project in *Progress Developer Studio for OpenEdge online help*.

6

Upgrade Security to OpenEdge 12

The following topics describe important notes regarding security updates made in OpenEdge Release 12.0, and also describe tasks you may need to complete to upgrade your application environment security to Release 12.0.

For details, see the following topics:

- Updates to password encryption with GENSPRINGPWD
- Updates to the cryptography used to secure Client-Principal objects
- PAS for OpenEdge and OpenEdge Authentication Gateway: Domain configuration
- Secure communications in OpenEdge Management
- Migrate OpenEdge Authentication Gateway configuration

Updates to password encryption with GENSPRINGPWD

PAS for OpenEdge supports the extended local (`extlocal`) user authentication manager. `extlocal` allows you to store encrypted user passwords in the `users.properties` file, which is local to your instance's web application. OpenEdge 11.7 supports several hashing algorithms—`bcrypt`, `sha256`, `sha512`, and `sha1`—for password encryption. In OpenEdge 12.0, however, the only supported algorithm is `bcrypt`, which is set by default—the other algorithms have been dropped because of their known vulnerability.

If you encrypted passwords with `sha256`, `sha512`, or `sha1` in your OpenEdge 11.7 applications, then perform the following actions to ensure a successful migration:

1. Obtain the original clear-text (unencrypted) password that you encrypted with `GENSPRINGPWD`.
2. Encrypt the password with `GENSPRINGPWD` in OpenEdge 12.0. This generates the encrypted password using `bcrypt`, which is the only supported algorithm.
3. Copy the encrypted password and add it to the corresponding entry in the instance's `users.properties` file.

The following is the syntax for storing passwords in the users.properties file:

```
username=password, role1, [role2, ..., roleN, ] { enabled | disabled }
```

Updates to the cryptography used to secure Client-Principal objects

Any static Client-Principal objects created by an ABL application and used to access an ABL application code or an OpenEdge database must be created again in OpenEdge 12.0. We provide this capability for the ABL server application code to verify a client before the client executes something.

PAS for OpenEdge and OpenEdge Authentication Gateway: Domain configuration

In OpenEdge, the authentication system used to authenticate a given user identity is identified by an OpenEdge security domain, or simply OpenEdge domain. To authenticate a user identity, an authentication system must:

- Be enabled for user authentication
- Have access to a valid set of user accounts
- Be part of an OpenEdge domain configuration that is enabled for run time

This configuration set is necessary for a domain to be enabled for authentication. For an OpenEdge authentication operation, such a domain must be appended to the username to obtain the fully qualified `user ID`.

For users who do not specify a domain, OpenEdge provides a built-in blank domain and its blank access code. This is to cater to the needs of single OpenEdge domain customer architectures. This default behavior extends to PAS for OpenEdge and OpenEdge Authentication Gateway configurations.

Because of vulnerabilities, the ability to configure a domain access code in the PAS for OpenEdge and Authentication Gateway configuration properties files has been removed in OpenEdge 12.0. Now, you must explicitly define the domains and their access codes, which are used by the PAS for OpenEdge and Authentication Gateway servers to create Client-Principal objects. You cannot configure an OpenEdge domain access code in the PAS for OpenEdge and OpenEdge Authentication Gateway configuration properties files. If you use the properties file to configure the domain access code, the servers will not start.

Explictly defining domains and their access codes is the default functionality in 12.0 OpenEdge Authentication Gateway, in which the access code 'key' property is removed from the sts.properties file for clarity and for the server to start.

PAS for OpenEdge configuration ships with a built-in OpenEdge domain registry file, which supplies the same default blank domain and its blank access code. If your 11.7.x PAS for OpenEdge application security configuration:

- Uses only the default settings, then no migration is required
- Uses a domain registry file (with a list of domains and their access codes defined), then generate the registry file again using the GENDOMREG utility
- Allows multiple domains, but uses the default blank domain access code to seal Client-Principal objects, then remove the 'key' property and use the OpenEdge domain registry configuration

Secure communications in OpenEdge Management

If you migrate your resource configuration from OpenEdge 11.7.x to OpenEdge 12.0, then if you had the `HTTPS` port turned off in your 11.7.x version, it will be turned on automatically in OpenEdge 12.0. This is because both `HTTP` and `HTTPS` are turned on by default in the current release.

To turn off secure communications (`HTTPS`) in your OpenEdge 12.0 environment, perform either of the following actions:

1. Manually edit the `catalina.properties` file in `$OEMGMT/conf` to comment out `psc.as.https.port=9443`.
2. Use the OpenEdge Management console to turn off the `HTTPS` port.

Migrate OpenEdge Authentication Gateway configuration

OpenEdge Authentication Gateway instances created in OpenEdge 11.6.x or 11.7.x are not compatible with OpenEdge 12.0.

To migrate your Authentication Gateway instance configuration from 11.7.x to the current release:

1. Back up the instance to save your 11.7.x configuration.
2. Create a new Authentication Gateway server instance during the OpenEdge 12.0 installation.
3. Migrate the 11.7.x configuration from the backup to the new 12.0 instance:
   a) Configure the `HTTPS` port (the procedure is unchanged from 11.7.x).
   b) Copy the `tomcat-keystore.p12` file, which has the server instance's `TLS` private key and server certificate, to the 12.0 instance (`/oeauthserver/conf`).
   c) Merge any `HTTPS` keystore and password configuration.
   d) Merge any configuration changes made to the `conf/openedge.properties` file from the 11.7.x instance using your favorite merge tool.
   e) Copy the tomcat-users.xml from the 11.7.x instance to the 12.0 instance (`/oeauthserver/conf`).
   f) Copy any `.r` ABL application files from the 11.7.x instance (OERealm, Policy, and Event handlers) to the 12.0 instance.
   g) Copy the OpenEdge domains configuration file (`domains.json`) from the 11.7.x instance to the 12.0 instance (`/oeauthserver/webapps/ROOT/WEB-INF/config`).

h) Copy the users.properties file from the 11.7.x instance to the 12.0 instance (`/oeauthserver/webapps/ROOT/WEB-INF/config`).

i) Create a new `domains.keystore` file from the 11.7.x instance's `domainsks.csv` file using the `gendomreg` utility and add the new `domains.keystore` file to `/oeauthserver/webapps/ROOT/WEB-INF/config` in the 12.0 instance.

j) Merge the 11.7.x instance's `sts.properties` configuration with the 12.0 instance's `sts.properties` file using your favorite merge tool.

4. If you have client-server key security enabled for your 11.7.x instance, then:

   a) Create a new 12.0 server-key file using the same `-url` parameter as was used to generate the 11.7.x server-key file. Store the new server-key file in the location which is specified as the value of the `sts.server.keyfile` property in the `sts.properties` file.

   b) Install a new client-key file on the local server using the `stskeyutil` tool.

   c) Test the server-key access using the `stskeyutil` tool.

5. Start the 12.0 instance and then test access to the instance's OpenEdge domain configurations using the `stsclientutil` tool.

6. If you have enabled client-server key security for your Authentication Gateway, then:

   a) Install a new client-key in all OpenEdge 12.0 client installations that replaced the 11.7.x installations.

   b) Test the client-key access using the `stsclientutil` tool.

7. Test the 12.0 PAS for OpenEdge and OpenEdge database connections that are configured to use the Authentication Gateway.

7

Upgrade ABL Applications

OpenEdge Release 12.0 introduces some security changes that may require you to upgrade your ABL applications, depending on the specific OpenEdge features that your ABL applications use.

ABL callbacks

If the OpenEdge security domain configuration in your OpenEdge Database employs ABL callbacks, you must update the domain configuration by re-registering the new version 12 compiled module. Use the database Data Administration tool to register the version 12 module in the Authentication System dialog box.

Security domain and system tables loaded from dump file

If your product's database installation loads the OpenEdge domain and system tables from a dump (`.d`) file, you will need to migrate your dump file for use in creating a version 12 database. To perform this migration, you must define your product's default OpenEdge domain and system tables in a version 12 database, and then use the Data Administration utility to create a new dump file.

Additional changes that may be needed to application code

You might also need to update your ABL application code in the following cases:

- If you use an `oech1::` encoded OpenEdge domain access-code in the `CLIENT-PRINCIPAL:SEAL()` method, you must update the code to pass a clear-text value as the method parameter.
- If you use the `AUDIT-POLICY:ENCRYPT-AUDIT-MAC-KEY()` method to manage OpenEdge domain access codes in your database, you must replace this method with the `SECURITY-POLICY:ENCODE-DOMAIN-ACCESS-CODE()` method.
- If you use the `RCODE-INFO:MD5-VALUE` attribute, you must replace it with the `RCODE-INFO:SIGNATURE-VALUE` attribute. **Note** that the size of the return value increases from 16 bytes of storage to 32 bytes.

- If you use object serialization between ABL clients and application servers, you must migrate both clients and server to version 12.
- If you use object serialization to persistent storage (for example, file system or database), you must recreate the object data in the persistent storage. Serialized data written in a version 11 ABL application is incompatible with a version 12 ABL client's ability to read that data.

8

Migration notes

When upgrading from OpenEdge 11.7.x to OpenEdge 12.0, bear in mind the following:

- If you import OpenEdge 11.7.x installation configurations using the `configutil` utility, then the property files associated with the classic AppServer and the configurations related to OpenEdge Management, OpenEdge Business Process Management, and Tomcat-in-the-box will be ignored.

  For more information, see configutil in *Administer Progress Application Server for OpenEdge*.

- If you migrate your resource configuration from OpenEdge Management 11.7.x to OpenEdge Management 12.0 , then the following in the fathom.xml file will be ignored during the import process:

  - AppServer
  - AppServer Internet Adapter (AIA)
  - ODBC DataServer
  - REST resources
  - SonicMQ Adapter
  - WebSpeed
  - Web Services Adapter (WSA)