Upgrading to JADE Release 7.1

VERSION 7.1.09
Jade Software Corporation Limited cannot accept any financial or other responsibilities that may be the result of your use of this information or software material, including direct, indirect, special or consequential damages, or loss of profits. There are no warranties extended or granted by this document or software material.

You should be very careful to ensure that the use of this software material and/or information complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Jade Software Corporation Limited.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Copyright © 2017 Jade Software Corporation Limited.

All rights reserved.

JADE is a trademark of Jade Software Corporation Limited. All trade names referenced are the service mark, trademark, or registered trademark of the respective manufacturer.

For details about other licensing agreements for third-party products, you must read the JADE ReadMe.txt file.
# Contents

## Upgrading to JADE Release 7.1

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Important Information</td>
<td>7</td>
</tr>
<tr>
<td>JADE Release Support</td>
<td>8</td>
</tr>
<tr>
<td>Deimplementations and Deprecations</td>
<td>10</td>
</tr>
<tr>
<td>Binary Type pos Method Deprecation</td>
<td>10</td>
</tr>
<tr>
<td>Customizing the Presentation Client Installer</td>
<td>10</td>
</tr>
<tr>
<td>Java Framework</td>
<td>10</td>
</tr>
<tr>
<td>NoCRTC_RUNTIME Upgrade Initialization File Parameter</td>
<td>10</td>
</tr>
<tr>
<td>Replication Framework Deprecation</td>
<td>10</td>
</tr>
<tr>
<td>Server Read-Only User Deprecation</td>
<td>10</td>
</tr>
<tr>
<td>Silverlight XAML</td>
<td>11</td>
</tr>
<tr>
<td>Transient-Only Classes</td>
<td>11</td>
</tr>
<tr>
<td>Process Class getTransientOnlyClassesUsed Method</td>
<td>11</td>
</tr>
<tr>
<td>Visual Studio 2005 C++ Presentation Clients</td>
<td>11</td>
</tr>
<tr>
<td>Highlights in this Release</td>
<td>12</td>
</tr>
<tr>
<td>Platform Requirements for this Release</td>
<td>13</td>
</tr>
<tr>
<td>Accessing Details about Faults Fixed in Releases</td>
<td>14</td>
</tr>
<tr>
<td>How to Locate PARs Fixed in a Specific Release</td>
<td>14</td>
</tr>
<tr>
<td>Upgrading to JADE 7.1</td>
<td>15</td>
</tr>
<tr>
<td>Upgrading to JADE 7.1 from an Earlier Release</td>
<td>15</td>
</tr>
<tr>
<td>Running Two Releases of JADE on the Same Workstation</td>
<td>18</td>
</tr>
<tr>
<td>JADE Thin Client Upgrade</td>
<td>19</td>
</tr>
<tr>
<td>Upgrading a 32-bit Presentation Client Connecting to a 64-bit Application Server</td>
<td>19</td>
</tr>
<tr>
<td>Upgrading an SDS Native or RPS Secondary System</td>
<td>20</td>
</tr>
<tr>
<td>Upgrade Validation</td>
<td>20</td>
</tr>
<tr>
<td>Hot Fix Releases</td>
<td>20</td>
</tr>
<tr>
<td>JADE 7.1 Changes that May Affect Your Existing Systems</td>
<td>21</td>
</tr>
<tr>
<td>Compact JADE (PAR 61967)</td>
<td>21</td>
</tr>
<tr>
<td>JADE Remote Node Access (PAR 60364)</td>
<td>21</td>
</tr>
<tr>
<td>Relational Population Service (RPS) Mapping</td>
<td>21</td>
</tr>
<tr>
<td>Replayable Reorganization (NFS 60428)</td>
<td>22</td>
</tr>
<tr>
<td>SOAP XML Handling in Web Services (PAR 60987)</td>
<td>22</td>
</tr>
<tr>
<td>String Arrays and Dictionary Keys</td>
<td>22</td>
</tr>
<tr>
<td>Upgrading Collections (PAR 63704)</td>
<td>23</td>
</tr>
<tr>
<td>Changes in JADE Release 7.1.09</td>
<td>24</td>
</tr>
<tr>
<td>JADE Logical Certifier Utility (NFS 63393)</td>
<td>24</td>
</tr>
<tr>
<td>JadeTestCase Class Assert Methods (PAR 64825)</td>
<td>24</td>
</tr>
<tr>
<td>Presentation Client Installer Customization to be Deprecated</td>
<td>24</td>
</tr>
<tr>
<td>Resizing Dialogs (PAR 64258)</td>
<td>24</td>
</tr>
<tr>
<td>REST Services</td>
<td>25</td>
</tr>
<tr>
<td>Generated REST Service Description (PAR 64969)</td>
<td>25</td>
</tr>
<tr>
<td>REST Service Call of Protected or Read-only Properties (NFS 64481)</td>
<td>25</td>
</tr>
<tr>
<td>RPS Keywords (PAR 64608)</td>
<td>25</td>
</tr>
<tr>
<td>SDS Database Role (PAR 63603)</td>
<td>25</td>
</tr>
<tr>
<td>Upgrade Process Error 8724 (PAR 64710)</td>
<td>26</td>
</tr>
<tr>
<td>Using the JADE Report Writer White Paper</td>
<td>26</td>
</tr>
<tr>
<td>Changes in JADE Release 7.1.08</td>
<td>27</td>
</tr>
<tr>
<td>.NET</td>
<td>27</td>
</tr>
<tr>
<td>.NET Decimal Conversion (PAR 63449)</td>
<td>27</td>
</tr>
<tr>
<td>.NET Import and Usage (PAR 63621)</td>
<td>27</td>
</tr>
<tr>
<td>Advanced Notice of JADE 2016 Deprecation</td>
<td>27</td>
</tr>
<tr>
<td>Context-Sensitive Help to HTML5 Topics</td>
<td>27</td>
</tr>
<tr>
<td>Find Type Dialog (NFS 63581)</td>
<td>28</td>
</tr>
<tr>
<td>IndexLoadFactor Parameter in the [PersistentDb] Section (PAR 64067)</td>
<td>28</td>
</tr>
<tr>
<td>Node Object Type Values (PAR 64128)</td>
<td>28</td>
</tr>
<tr>
<td>Object Class autoPartitionIndex Method (PAR 63695)</td>
<td>29</td>
</tr>
<tr>
<td>Scaling the Splash Screen (PAR 63654)</td>
<td>29</td>
</tr>
</tbody>
</table>
Secure Sockets Library (PAR 63548, NFS 63550) ................................................. 29  
Table Sorting (PAR 63893) ............................................................................ 30  
Upgrade Validation (PAR 63810) .................................................................. 30  

Changes in JADE Release 7.1.07 ................................................................. 31  
Class Browser Displays the Schema Name (NFS 63547) ................................ 31  
Downloading Additional Thin Client Binaries (PAR 63174) ...................... 31  
Exception Dialog (NFS 63168) ...................................................................... 31  
External Function Calls (PAR 63250) ............................................................ 32  
FileFolder::browseForFolder Method (NFS 63418) ........................................ 32  
Imported Control Class Event Truncation (PAR 63460) ............................. 33  
JADE Debugger (PAR 46099) ....................................................................... 33  
JadeRecompileAllMethods Application Errors (PAR 63227) ........................ 33  
Logical Certifier Orphan Block Checking (PAR 63378) .............................. 34  
Method Signature Change (PAR 63453) ....................................................... 34  
Picture Class rotation Property (PAR 63420) ............................................... 34  
Relational Population Service OID Mapping (NFS 62231) ........................... 35  
REST Services .............................................................................................. 35  
Error if Unknown JSON Type (NFS 63335) ................................................. 35  
JadeRestService::getServerVariable (PAR 63247) ...................................... 35  
Parsing JSON Text (NFS 63367) ................................................................. 36  
ThreadPriority Parameter Removal (PAR 63251) ....................................... 37  
Time Millisecond Value (PAR 63572) ............................................................ 37  
Upgrade Validation in a Source-Stripped Database (PAR 63458) ................ 37  

Changes in JADE Release 7.1.06 ................................................................. 38  
Binary::uuidAsString Method (PAR 62807) .................................................. 38  
CMDDPrint::warnIfNoDefault Property (PAR 62950) .............................. 38  
Date Handling (PAR 62888) ........................................................................ 38  
Global Find and Replace Handling (NFS 63056) ......................................... 38  
JadeHTTPConnection Class Constants (NFS 62889) .................................... 38  
JADE Logical Certifier Utility ..................................................................... 39  
  JADE Logical Certifier Error 33 (PAR 62880) ............................................ 39  
  Logically Certifying Meta Data (PAR 62689) ............................................. 39  
  JadeRichText Control (NFS 62749) ............................................................ 39  
Loading Report Writer Files Clarification (PAR 62873) .............................. 40  
Notification Data Limits Clarification (PAR 62863) .................................... 40  
REST-based Web Services (PAR 62784) ..................................................... 40  
Schema and Class Deletion (NFS 62273) .................................................... 40  
String::asUuid Method (NFS 62803) ............................................................ 41  
Swapping F5 and F9 Accelerator Keys (NFS 63079) ................................... 41  
Synchronized Database Service (PAR 62798, 57217) ............................... 41  
JournalReadBuffers .................................................................................... 41  
JournalReplayBlocksize ............................................................................. 42  
Web Service Exposure Error Message (PAR 63094) .................................... 42  
Web Session Timeout (PAR 63011) .............................................................. 42  

Changes in JADE Release 7.1.05 ................................................................. 43  
Batch Load Lock Timeout (PAR 62567) ........................................................ 43  
Batch Schema Load Utility reportReplaceView Parameter (PAR 62375) .... 43  
Cache Size Limit Default Values (PAR 62677) ........................................... 43  
Dynamic Property Clusters ....................................................................... 43  
  Deleting Dynamic Clusters (PAR 62396) .................................................. 43  
  Dynamic Property Cluster Names (PAR 62436) ....................................... 44  
  Extracting and Loading Dynamic Property Clusters (PAR 62434) ........... 44  
JadeHTMLClass::buildFormActionOnly Method (PAR 62632) .................. 44  
JadeRecompileAllMethods Application (PAR 62539) .............................. 44  
MemoryAddress Value Adjustment (PAR 62517) ......................................... 44  
Microsoft SQL Server 2014 Support (PAR 62731) ....................................... 44  
Microsoft Visual C++ Redistributable Package (PAR 62506) .................... 45  
Thread Parameters in the [JadeServer] Section (PAR 62498) ....................... 45  

Changes in JADE Release 7.1.04 ................................................................. 46  
Converting User Database Files (PAR 62300) ............................................. 46  
Database Roll-Forward Recovery (PAR 60499) .......................................... 46  
DataPumpApplication Parameter (PAR 62311) ......................................... 46  
Exception::errorItem Property Length (PAR 62349) ................................. 46  
JadeHttp Logging of Message Meta Data (PAR 62345) ............................. 47  

Changes and New Features in JADE Release 7.1.03 ................................. 48
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>.NET</td>
<td>48</td>
</tr>
<tr>
<td>.NET Class Library (PAR 60740)</td>
<td>48</td>
</tr>
<tr>
<td>C# Exposure Wizard</td>
<td>48</td>
</tr>
<tr>
<td>Exposure Entity Maximum Lengths (NFS 61179)</td>
<td>48</td>
</tr>
<tr>
<td>ExternalCollection Subclasses (PAR 60285)</td>
<td>48</td>
</tr>
<tr>
<td>Saving Definition and Generate Values (PAR 59367)</td>
<td>48</td>
</tr>
<tr>
<td>Custom Control Tab Order (PAR 61929)</td>
<td>49</td>
</tr>
<tr>
<td>JADE .NET Developer's Reference</td>
<td>49</td>
</tr>
<tr>
<td>Ad Hoc Indexes</td>
<td>49</td>
</tr>
<tr>
<td>Adding User Classes at Run Time</td>
<td>50</td>
</tr>
<tr>
<td>AutoComplete Functionality</td>
<td>50</td>
</tr>
<tr>
<td>AutoComplete Selection from a List (PAR 61330)</td>
<td>50</td>
</tr>
<tr>
<td>Pseudo Type Variable Support (PAR 61634)</td>
<td>50</td>
</tr>
<tr>
<td>Application Class IsValidObject Method (PAR 52806)</td>
<td>51</td>
</tr>
<tr>
<td>Application Programming Interface (PAR 60679)</td>
<td>51</td>
</tr>
<tr>
<td>Class Number Extensions</td>
<td>52</td>
</tr>
<tr>
<td>Class Persistent Instances Methods (PAR 61167)</td>
<td>52</td>
</tr>
<tr>
<td>create Statement Handling (PAR 62019)</td>
<td>52</td>
</tr>
<tr>
<td>Disk Write-Cache Messages (PAR 60863)</td>
<td>52</td>
</tr>
<tr>
<td>DynamicDictionary Methods (PAR 58349)</td>
<td>53</td>
</tr>
<tr>
<td>Dynamic Clusters and Properties</td>
<td>53</td>
</tr>
<tr>
<td>Entity Name Maximum Length (NFS 61179)</td>
<td>54</td>
</tr>
<tr>
<td>External Function and External Method Entry Point Length (PAR 61412)</td>
<td>55</td>
</tr>
<tr>
<td>Extract Sort Exception (PAR 61403)</td>
<td>55</td>
</tr>
<tr>
<td>Generated JADE .NET Names (NFS 60673)</td>
<td>55</td>
</tr>
<tr>
<td>Interface Stub Methods (NFS 60359)</td>
<td>55</td>
</tr>
<tr>
<td>JADE Debugger</td>
<td>56</td>
</tr>
<tr>
<td>Attaching the JADE Debugger to a Running Application (NFS 2413)</td>
<td>56</td>
</tr>
<tr>
<td>debugApplicationWithParameter Method (NFS 61035)</td>
<td>56</td>
</tr>
<tr>
<td>Displaying the Current Exception Handler Stack (NFS 46856)</td>
<td>56</td>
</tr>
<tr>
<td>JADE Initialization File</td>
<td>57</td>
</tr>
<tr>
<td>AcceptZeroEnvironmentUUID Parameter</td>
<td>57</td>
</tr>
<tr>
<td>BackupCompressedFileGrowthIncrementParameter (PAR 60663)</td>
<td>57</td>
</tr>
<tr>
<td>DiskCacheWriteOnlyDbSegments Parameter (PAR 60707)</td>
<td>57</td>
</tr>
<tr>
<td>JADE Ad Hoc Index [JadeAdHocIndex] Section</td>
<td>58</td>
</tr>
<tr>
<td>JADE Application Server Upgrade Parameters</td>
<td>58</td>
</tr>
<tr>
<td>JADE Sentinel Configuration</td>
<td>58</td>
</tr>
<tr>
<td>JADE Server Section Parameter Values (PAR 60842)</td>
<td>58</td>
</tr>
<tr>
<td>Parameter Value Handling (PAR 61704)</td>
<td>58</td>
</tr>
<tr>
<td>UpgradeRuntimeTo64bit Parameter (NFS 61726)</td>
<td>59</td>
</tr>
<tr>
<td>JADE Logical Certifier (PAR 60649, 60505, 60506)</td>
<td>59</td>
</tr>
<tr>
<td>Detecting a Collection Object with Multiple Keys (PAR 61172)</td>
<td>59</td>
</tr>
<tr>
<td>JADE Remote Node Access Utility</td>
<td>59</td>
</tr>
<tr>
<td>Menu Item Changes (PAR 57911)</td>
<td>59</td>
</tr>
<tr>
<td>Termination Message Box</td>
<td>60</td>
</tr>
<tr>
<td>JADE Version Information Utility</td>
<td>60</td>
</tr>
<tr>
<td>Commit Limit Values (PAR 60968)</td>
<td>60</td>
</tr>
<tr>
<td>datmask and symmask Optional Parameters (PAR 61797)</td>
<td>60</td>
</tr>
<tr>
<td>Methods View Browser</td>
<td>60</td>
</tr>
<tr>
<td>JADE Scripts and Test Cases in the Methods View Browser (PAR 60442)</td>
<td>60</td>
</tr>
<tr>
<td>Methods View Browser Context Menu (PAR 60947)</td>
<td>61</td>
</tr>
<tr>
<td>Monitoring your Active Process Using JADE Sentinel</td>
<td>61</td>
</tr>
<tr>
<td>ODBC Driver Query Execution Timeout (NFS 60759)</td>
<td>61</td>
</tr>
<tr>
<td>OpenSSL Library</td>
<td>61</td>
</tr>
<tr>
<td>Application Server OpenSSL Cipher Suite (NFS 61541)</td>
<td>62</td>
</tr>
<tr>
<td>Library Version Upgrade (PAR 61368)</td>
<td>62</td>
</tr>
<tr>
<td>Recompiling All Methods (PAR 61988)</td>
<td>62</td>
</tr>
<tr>
<td>References Browser (NFS 61215)</td>
<td>62</td>
</tr>
<tr>
<td>Relationship View Diagram Printing (PAR 61264)</td>
<td>63</td>
</tr>
<tr>
<td>REST-Based Web Services</td>
<td>63</td>
</tr>
<tr>
<td>Snapshot Database Recovery</td>
<td>63</td>
</tr>
<tr>
<td>StringUtf8Array Elements (PAR 60605)</td>
<td>64</td>
</tr>
<tr>
<td>System Class IsValidProcess Method (PAR 60191, 61755)</td>
<td>64</td>
</tr>
<tr>
<td>Thin Client Downloads</td>
<td>64</td>
</tr>
<tr>
<td>Thin Client Automatic Download (PAR 61337)</td>
<td>64</td>
</tr>
</tbody>
</table>

Copyright 2017, Jade Software Corporation Ltd. All rights reserved JADE-7.1.09
<table>
<thead>
<tr>
<th>Topics</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upgrading 32-bit Thin Client Binaries to 64-bit (NFS 61726)</td>
<td>65</td>
</tr>
<tr>
<td>Transient Leak Detection (NFS 60726)</td>
<td>65</td>
</tr>
<tr>
<td>Unaudited Database Files and Partitions</td>
<td>65</td>
</tr>
</tbody>
</table>
Upgrading to JADE Release 7.1

This document covers the following topics.

- Important Information
- JADE Release Support
  - Deimplementations and Deprecations
- Highlights in this Release
- Platform Requirements for this Release
- Accessing Details about Faults Fixed in Releases
- Upgrading to JADE 7.1
  - Upgrading to JADE 7.1 from an Earlier Release
  - JADE Thin Client Upgrade
  - Upgrading an SDS Native or RPS Secondary System
  - Upgrade Validation
  - Hot Fix Releases
- JADE 7.1 Changes that May Affect Your Existing Systems
- Changes in JADE Release 7.1.09
- Changes in JADE Release 7.1.08
- Changes in JADE Release 7.1.07
- Changes in JADE Release 7.1.06
- Changes in JADE Release 7.1.05
- Changes in JADE Release 7.1.04
- Changes and New Features in JADE Release 7.1.03

Tip For details about using a web browser to view the JADE product information, see "JADE HTML5 Online Help", in Chapter 2 of the JADE Development Environment User’s Guide. For details about using Acrobat Reader to view JADE documents, see "JADE Product Information Library in Portable Document Format", in Chapter 2 of the JADE Development Environment User’s Guide.

The JADE Product Information Library document (JADE) provides a summary of contents of documents in the JADE product information library and navigation to the documents.

If you want to develop your own installation process for Windows, the JADE install and upgrade steps are documented in the ReadmeInstallSteps document in the documentation directory.

Note To customize the deployment upgrade on Windows, see "Customizing the Deployment Upgrade Process", in Appendix A of the JADE Runtime Application Guide.
Important Information

You must perform each of the following actions before and after upgrading to JADE release 7.1.

**Caution** Proceed to the next step only when all errors reported in the current step have been resolved.

1. On your current system before upgrading to JADE 7.1, perform each of the following actions.
   a. Perform a physical certify operation using the JADE Database Utility (`jdbutil.exe` or `jdbutilb.exe`), to ensure that the system is structurally sound.
   b. Perform a meta logical certify operation, to ensure that the meta model is clean.
   c. Perform a logical certify operation, to ensure that the user data is referentially correct.
   d. If your database has partitioned database files, ensure that any offline partitions are brought online.
   e. Use the JADE Database utility to take a full backup of your existing JADE 7.0 or JADE 6.3 database.
   f. If the `JadeReportWriterSchema` has been installed, you must extract the data and then delete the schema before upgrading; that is:
      
      i. In the JADE Report Configuration application, select the **Unload All** command from the View menu to unload all of your report data to a `.rwa` file. In the **File Name** text box of the Unload All dialog, specify the name and location of the file to which you want to extract all of your report writer data; for example:
         
         d:\jade\rpts\alldata.rwa
         
         Alternatively, you can use the **executeMethod** in the batch JADE Load utility to extract all reports, as shown in the following example.
         
         jadloadb path=d:\jade\system ini=d:\jade\myjade.ini
         executeClass=JadeReportWriterGlobal executeMethod=unloadAllToFile
         executeParam=d:\jade\rpts\alldata.rwa
         
      ii. Delete the `JadeReportWriterSchema` schema by using the **Remove** command in the Schema menu of the JADE development environment or by using the **deleteSchema** parameter in the batch JADE Load utility; for example:
         
         jadloadb path=d:\jade\system ini=d:\jade\myjade.ini
         deleteSchema=JadeReportWriterSchema
         
2. In JADE 7.1 immediately after your upgrade from an earlier release, perform each of the following actions.
   a. Perform a physical certify operation using the JADE Database Utility (`jdbutil.exe` or `jdbutilb.exe`), to ensure that the system is structurally sound.
   b. Perform a meta logical certify operation, to ensure that the meta model is clean.
   c. Perform a logical certify operation, to ensure that the user data is referentially correct.
   d. If the `JadeReportWriterSchema` is required, reload the 7.1 `JadeReportWriterSchema` and then reload the reports and data extracted in step 1.f of this instruction, by using one of the following actions.
      
      i. In the JADE Report Configuration application, select the **Load All** command from the View menu to load all report data from your extracted `.rwa` file.
      
      In the **File Name** text box of the Load All dialog, specify the name and location of the report data extract file you want to load; for example:
         
         d:\jade\rpts\alldata.rwa
— Use the `reportLoadAllFile` parameter in the batch load utility to specify the fully qualified name of a single unload (extract) file that contains all JADE Report Writer view, folder, system option, user, and report definitions that you want to load, as shown in the following example.

```
jadloadb path=d:\jade\system reportLoadAllFile=d:\jade\rpts\alldata.rwa ini=d:\jade\myjade.ini
```

e. Use the JADE Database utility to take a full backup of your JADE 7.1 database.
JADE Release Support

For details about the:


JADE 7.1 is built using Microsoft Visual Studio 2013, which requires the installation of appropriate C++ runtime binaries.

For details about the deimplementations and deprecations in this release, see the following subsection.

Deimplementations and Deprecations

This section contains the deimplementations and deprecations in this release. (See also "JadeHTMLClass::buildFormActionOnly Method (PAR 62632)" under "Changes in JADE Release 7.1.05" and "Advanced Notice of JADE 2016 Deprecation" under "Changes in JADE Release 7.1.08", later in this document.)

Binary Type pos Method Deprecation

From release 7.1.06, the Binary primitive type pos method has been deprecated, and will be deleted in a future JADE release.

**Note** Use the Binary type posBinary or posByte method instead of the deprecated pos method.

Customizing the Presentation Client Installer

The Customizable Presentation Client Installer (CPCI) that provides a simple customization mechanism for the installation of JADE presentation client software is to be deprecated in JADE 2016.

Java Framework

As notified in JADE 7.0.10, JADE's Java framework has been deprecated in this release.

A schema load now fails with error 6449 (*Java exposures no longer valid*) if a Java exposure exists in the schema being loaded.

NoCRTRuntimeUpgrade Initialization File Parameter

The NoCRTRuntimeUpgrade parameter in the [JadeAppServer] section of the JADE initialization file is no longer supported.

Replication Framework Deprecation

The JADE Replication Framework has been deprecated in this release.

Server Read-Only User Deprecation

As notified in JADE 7.0.10, the ReadOnlyUser value of the server parameter in the jadclient executable has been deprecated in this release, as there is no longer a requirement to run JADE from read-only media. (ReadOnlyUser is therefore no longer a valid value for the Server parameter in the [NonGuiClient] section of the JADE initialization file.)
Silverlight XAML

As notified in JADE 7.0.10, JADE’s Silverlight Extensible Application Markup Language (XAML) implementation has been deprecated in this release.

Transient-Only Classes

With the extension of class numbers in this release, the transient-only class feature is no longer available. (See also "Class Number Extensions" under "Changes and New Features in JADE Release 7.1.03").

Process Class getTransientOnlyClassesUsed Method

Because the extension of class numbers has resulted in the deprecation of transient-only classes, the Process class getTransientOnlyClassesUsed method has been deprecated in this release. This method is still defined in the RootSchema, but calling it raises exception 1112 *(Feature has been deprecated)*.

Visual Studio 2005 C++ Presentation Clients

JADE 7.1 does not support 32-bit presentation clients using the Visual Studio 2005 C++ runtime binaries that were supported by the JADE 7.0 and 6.3 releases.
Highlights in this Release

The highlights in JADE release 7.1, which help you to deliver high-performance, interoperable applications on Windows for both 32-bit and 64-bit platforms, are as follows.

- Class number extensions
  JADE has extended the number of user classes permitted to just under one million. In earlier releases, the limit for user classes was 31,279, with some class numbers reserved for transient-only classes. For details, see "Class Number Extensions", later in this document.

- Ad hoc indexes
  Ad hoc indexes enable you to create indexes suitable for optimizing ad hoc queries without requiring database reorganization. For details, see "Ad Hoc Indexes", later in this document.

- Web services extensions
  JADE now provides the Representational State Transfer (REST) Application Programming Interface (API), which provides you with an alternative to the Simple Object Access Protocol (SOAP) protocol to implement Web services. For details, see "REST-Based Web Services", later in this document.

- .NET API enhancements
  JADE has made a number of changes to facilitate your .NET development, including:
  
  - User class functionality, dynamic property access, and JADE method call invocation performance improvements
  
  - Implementation of a ServiceModel assembly that enables C# proxy objects to be serialized across .NET Windows Communication Foundation (WCF) Web services, also providing support for .NET WCF client applications to participate in system transactions
  
  - The JADE .NET Developer's Reference has been enhanced to include .NET and JADE requirements, object management information, a sample application, and an introductory tutorial

  For details, see " .NET", later in this document.

- Extended dynamic property support
  You can now access JADE dynamic properties explicitly, rather than using calls to the getPropertyValue and setPropertyValue methods. The JADE compiler supports enhancement this explicit dynamic property access and allows the extraction and load of schemas that contain dynamic properties. For details, see "Dynamic Properties", later in this document.

- Unaudited database files and partitions
  Database files and partitions can now be updated with auditing disabled, to eliminate journal disk space use and I/O overhead when loading data into a file or partition and restart recovery is not a requirement. For details, see "Unaudited Database Files and Partitions", later in this document.

- Snapshot database recovery
  JADE now supports the recovery of database restored from a snapshot created by third-party technology; for example, VMWare snapshots. For details, see "Snapshot Database Recovery".
Platform Requirements for this Release

JADE 7.1 database and application servers and standard and presentation (thin) clients run on the Windows operating systems specified in the following list.

- Windows Server 2016
- Windows 10
- Windows 8.1 and Windows Server 2012 R2
- Windows 8 and Windows Server 2012
- Windows 7 and Windows Server 2008 R2

As Windows Vista extended support from Microsoft ended in April 2017, JADE no longer supports it.

Note  The database server will run only on a 64-bit version of Windows. Clients can be 32-bit or 64-bit.

Compact JADE thin client only runs on the Mobile operating systems specified in the following list.

- Windows Mobile 5.0 for Pocket PC Phone Edition
- Windows Mobile 5.0 for Pocket PC
- Windows Mobile 6.0 or 6.1 Classic
- Windows Mobile 6.0 or 6.1 Professional
- Windows Mobile 6.5 Professional

Note  Windows Mobile 6 Standard and Windows Phone 7 are not supported.

To deploy JADE Web applications under Apache, a minimum of Apache 2.4.10 for Microsoft Windows is required.

Any node that hosts a JADE 7.1 database must be 64-bit, including single user application servers and single user standard clients. The 32-bit clients that are available are multiuser clients (application servers, presentation clients, and standard clients) that connect to the 64-bit database server.

Any system that is to run JADE 7.1 must have the Microsoft Visual C++ Redistributable Package installed. For JADE 7.1:

- 64-bit versions, the package is Visual C++ 2013 Redistributable Package (x64), version 12.0.21005.1
- 32-bit versions, the package is Visual C++ 2013 Redistributable Package (x86), version 12.0.21005.1
Accessing Details about Faults Fixed in Releases

To access the complete documentation about the Product Anomaly Reports (PARs) fixed in this release, run Parsys, our Fault Managements and Customer Contact system. This system also enables you to view the progress of your own contacts.

If you have any queries about Parsys, please direct them to JADE Parsys Support in the first instance, at parsyssupport@jadeworld.com. You can download the install shield for Parsys from the following URL.

https://www.jadeworld.com/developer-center/jade-support/parsys

When you first run the Parsys application, it downloads an update via the automatic thin client download feature. When this has completed and you have the log-on form ready and waiting, please contact JADE Parsys Support, who will then send you an e-mail message with your user code and password details. Parsys requires you to change your password when you first log on.

Note Because the encryption of passwords is a one-way algorithm, we cannot advise you of your password should you forget it, but we can reset it to a known value again.

How to Locate PARs Fixed in a Specific Release

This section describes the actions that enable you to locate Product Anomaly Reports (PARs) fixed in a specific release.

To locate the PARs fixed in a specific release

1. Select the Advanced Search command from the Search menu with the following settings.
   a. On the Basic Search Criteria sheet, the Latest option button is selected in the Mode group box.
   b. All is selected in the Priority list box.
   c. The PAR check box is checked in the Phase group box.
   d. The Fault and NFS types are selected.
   e. The Closed and Patched check boxes are checked in the Status group box.

   Note If you want to restrict the search to the hot fixes that were produced, check the A hot fix was created check box on the Advanced Search Criteria II (Optional) sheet.

2. On the Advanced Search Criteria III (Optional) sheet:
   □ In the Closed list box of the Releases group box, select the release whose fixed PARs you want to locate (for example, the 7.1.03 list item).

3. Click the Search button.
Upgrading to JADE 7.1

This section covers the following topics.

- Upgrading to JADE 7.1 from an Earlier Release
  - Running Two Releases of JADE on the Same Workstation
- JADE Thin Client Upgrade
- Upgrading an SDS Native or RPS Secondary System
- Upgrade Validation
- Hot Fix Releases

Caution Before you upgrade to JADE 7.1, refer to "Important Information" and to "JADE 7.1 Changes that May Affect Your Existing Systems", elsewhere in this document.

Upgrading to JADE 7.1 from an Earlier Release

Server nodes (database upgrades) require the 64-bit edition. Client nodes can upgrade the 64-bit edition or the 32-bit edition. This upgrade involves a migration of the data from an earlier release to JADE 7.1, due to the nature of the structural changes of the objects in the database.

Notes This database upgrade will require sufficient disk to accommodate an additional full copy of the database plus head-room of about 10 percent of the database size.

As the JADE Replication Framework has been deprecated in JADE 7.1, any applications that use the Replication Framework may cease to function correctly.

If you want to develop your own installation process, refer to the JADE install and upgrade steps documented in the ReadmeInstallSteps.pdf document in the documentation directory.

Caution As with any JADE release, you may need to recompile any external method Dynamic Link Libraries (DLLs) or external programs using the JADE Object Manager Application Programming Interfaces (APIs) with the new JADE Include and Library files before you attempt to run your upgraded JADE systems. (For details about the JADE Object Manager APIs, see Chapter 3 of the JADE Object Manager Guide.)

Some obsolete files are deleted from the JADE directories when upgrading from an earlier JADE release. If you require these files for your JADE system, you must save them before you upgrade or restore them from the original JADE 7.0 or 6.3 release medium.

Documentation and example files are not part of the installation and must be downloaded and installed from the JADE Web site or the release medium separately, if required, into the documentation folder and examples folder, respectively, of your JADE installation directory.

The JADE Setup program enables you to upgrade your binary and database files to JADE 7.1 from a JADE 6.3 release or higher, by performing the following actions.

1. Ensure that your JADE environment is JADE release 6.3 or higher.
   
   On the JADE 7.0 or 6.3 system, carry out the following certify operations. Proceed to the next certify operation only when any and all errors reported in the current operation are resolved.
   
   a. A physical certify using JADE Database utility (jdbutil.exe or jdbutil.exe), to ensure that the system is structurally correct. (For details, see Chapter 1 of the JADE Database Administration Guide.)
   
   b. A meta logical certify, to ensure that the meta model is clean. (For details, see "Running a Non-GUI JADE Logical Certifier", in Chapter 5 of the JADE Object Manager Guide.)
c. A logical certify, to ensure that the user data is referentially correct. (For details, see "Running the Diagnostic Tool", in Chapter 5 of the JADE Object Manager Guide.)

Note If you are unsure how to interpret the information output by the certify process, contact JADE Support (jadesupport@jadeworld.com) for advice.

2. If your database has partitioned database files, ensure that any offline partitions are brought online.

3. Use the JADE Database utility to take a full backup of your existing JADE 6.3 or 7.0 database.

Caution As roll-forward recovery of the installation and upgrade process is not supported, it is important that you backup your database before starting the JADE Setup process to install JADE 7.1 and upgrade your existing data.

4. If the JadeReportWriterSchema has been installed, you must extract the data and then delete the schema before upgrading; that is:

a. In the JADE Report Configuration application, select the Upload All command from the View menu to upload all of your report data to a .rwa file. In the File Name text box of the Upload All dialog, specify the name and location of the file to which you want to extract all of your report writer data; for example:

   d:\jade\rpts\alldata.rwa

Alternatively, you can use the executeMethod in the batch JADE Load utility to extract all reports, as shown in the following example.

   jadloadb path=d:\jade\system ini=d:\jade\myjade.ini schema=JadeReportWriterSchema executeSchema=JadeReportWriterSchema executeClass=JadeReportWriterGlobal executeMethod=uploadAllToFile executeParam=d:\jade\rpts\alldata.rwa

b. Delete the JadeReportWriterSchema schema by using the Remove command in the Schema menu of the JADE development environment or by using the deleteSchema parameter in the batch JADE Load utility; for example:

   jadloadb path=d:\jade\system ini=d:\jade\myjade.ini deleteSchema=JadeReportWriterSchema

5. If the dbghelp.dll file is present in the bin folder of the 7.1 system when upgrading to JADE 7.1, JADE Sentinel may crash during the upgrade process. After upgrading, a Windows error may occur and the JADE Sentinel fail to start.

Although the upgrade process (or JADE) itself will continue successfully, you should delete the dbghelp.dll file from the bin folder before you upgrade to JADE 7.1 or disable JADE Sentinel (that is, by setting the value of the EnableSentinel parameter in the [FaultHandling] section of the JADE initialization file to False).

6. Ensure that you have the appropriate privileges or capabilities to install applications.

Note Installing any Microsoft redistributable package requires administration privileges.

If you are upgrading the 64-bit edition, the vcredist_x64.exe Microsoft Visual C++ 2013 Redistributable Package (x64) is required. If you are upgrading the 32-bit edition of JADE, the vcredist_x86.exe Microsoft Visual C++ 2013 Redistributable Package (x86) is required. This will be installed during the JADE installation and is supplied on the JADE distribution media.

7. To start the JADE Setup program, invoke the setup.exe program from the Jade71 release medium or execute the executable program downloaded from the JADE Web site.

8. If they are not already installed, the required Microsoft Windows C++ Redistributable Packages are installed.

9. In the Welcome folder, click the Next button to continue the upgrade process.

10. Read the entire software license agreement in the Software License Agreement folder and then click the Yes
11. In the Installation Type folder, select the **Feature Upgrade** option button, to specify that you want to upgrade an existing JADE release. By default, the **Fresh Copy** option is selected.

12. In the Setup Type folder, select the type of installation that you are upgrading. By default, the **Development** option is selected for 64-bit installation or **JADE Client** for 32-bit installation.

**Note**  The **Custom** type applies only to a **Fresh Copy** installation type, and is not relevant when upgrading. The **SDS/RPS Database Server** option applies only to 64-bit **Feature Upgrade** installation type (it is not available for a JADE 6.3 to 7.1 upgrade).

13. In the Select Installation Folders folder, specify the locations of the JADE files that are to be upgraded.

The upgrade process defaults to the most-recently used JADE files, and displays these values in the **Install Directory**, **Executable Directory**, and **Database Directory** text boxes. The installation directory is most likely to be the root directory in which you installed JADE, unless you subsequently renamed the root directory or moved the files to another location.

If the locations are not as required, click the adjacent browse buttons (indicated by the ... ellipsis symbols) to display the common File Selection dialog that enables you to select the appropriate directories and files. By default, the **jade.ini** file located in your specified database directory is used. If required, use the **JADE INI File** text box to specify a different valid fully qualified directory and name of the JADE initialization file; for example:

```
d:\my\sys\jade\system\jade.ini
```

If Program Start folders are to be updated, specify the name of the folder in the **JADE Program Folder** text box. If you are unsure of the folder to be updated, click the adjacent browse button to display the common Folder selection dialog that enables you to select the folder.

The **Database Directory** text box enables you to explicitly specify the location in which the database (system) files are installed. When the destination folder is not **Program Files**, the database destination defaults to **system** under the install folder (for example, if you specify **c:\jade71** in the **Install Directory** text box, the database directory defaults to **c:\jade71\system**).

If the installation directory is a subdirectory of the programmatically determined location of **Program Files**, the **Program Files** portion of the install directory is replaced with the programmatically discovered location for the common application data directory (for example, if you specify **c:\Program Files\Jade71** in the **Install Directory** text box, the default database location is **c:\ProgramData\Jade71\system**).

The process checks whether the specified database directory is a valid system and that it is the correct ANSI or Unicode type.

14. The Start Copying Files folder summarizing your upgrade options is displayed. If the selections displayed in the Start Copying Files folder are correct, click the **Next>** button.

Alternatively, click the **<Back** button to modify your selections.

15. The Question dialog is displayed, advising you to ensure that you have taken a full backup of that database before you proceed with the upgrade process.

When you are sure that you are upgrading the correct system (and that it has been backed up), click the **Yes** button to start the upgrade process.

16. A warning message box is then displayed, advising you that Dynamic Link Libraries (DLLs) may need to be recompiled. Click the **OK** button, to recompile any required DLLs.

17. A warning message may be displayed if the upgrade validation process has not completed. If so, check the **jadeupgrade.log** file for information about what needs to be modified in your user schemas to pass the validation and enable application execution.

18. When the upgrade is complete, the JADE Setup program informs you that the JADE Setup was successfully
completed and that you can now view the ReadMe.txt file. To view the ReadMe.txt file, ensure that the check box is checked (the default).

The ReadMe.txt file is then displayed in a text editor (for example, Notepad). The ReadMe.txt file is a read-only text file installed in your JADE root directory that you can print or delete, if required. This file contains a reference to other JADE-related documents.

19. Click the Finish button to end the JADE upgrade process.

20. Re-establish any control file paths set for database files and any required partition file attributes such as location, label, and frozen or offline states.

21. Install any jadrap database services (that is, JADE Remote Node Access services) you had set up in JADE 7.0 or 6.3. (For details, see "Running the Server Node as a Service", in the JADE Remote Node Access Utility User’s Guide.)

22. If the JadeReportWriterSchema is required, reload the 7.1 JadeReportWriterSchema and then reload the reports and data extracted in step 1 of this instruction, by using one of the following actions.

   a. In the JADE Report Configuration application, select the Load All command from the View menu to load all report data from your extracted .rwa file.

      In the File Name text box of the Load All dialog, specify the name and location of the report data extract file you want to load; for example:

      d:\jade\rpts\alldata.rwa

   b. Use the reportLoadAllFile parameter in the batch load utility to specify the fully qualified name of a single unload (extract) file that contains all JADE Report Writer view, folder, system option, user, and report definitions that you want to load, as shown in the following example.

      jadloadb path=d:\jade\system reportLoadAllFile=d:\jade\rpts\alldata.rwa

      ini=d:\jade\myjade.ini

23. Use the JADE Database utility to take a full backup of your JADE 7.1 database.

After the upgrade has completed, the database files of the earlier release are retained in the directory database-directory_70 or database-directory_63 (for example, c:\jade\system_70). The database-directory (for example, c:\jade\system) contains the upgraded JADE 7.1 database.

For details about the JADE 7.0 to 7.1 upgrade process checking for methods that referenced Microsoft Developer Network (MSDN) external functions and marking any such references for recompile during the upgrade validation phase, see "External Function Calls (PAR 63250)" under "Changes in JADE Release 7.1.07", later in this document. The references must be corrected before restarting the validation. You must manually check and validate any references in systems already running on JADE 7.1.

Caution As with any JADE release, you may need to recompile any external method Dynamic Link Libraries (DLLs) or external programs using the JADE Object Manager Application Programming Interfaces (APIs) with the new JADE Include and Library files before you attempt to run your upgraded JADE systems. (For details about the JADE Object Manager APIs, see Chapter 3 of the JADE Object Manager Guide.)

Some obsolete files are deleted from the JADE directories when upgrading from JADE 7.0 or 6.3. If you require these files for your JADE system, you must save them before you upgrade or restore them from the original JADE 7.0 or 6.3 release medium.

Documentation and example files are not part of the installation and must be downloaded and installed from the JADE Web site or the release medium separately, if required, into the documentation folder and examples folder, respectively, of your JADE installation directory.

Running Two Releases of JADE on the Same Workstation

You can have two releases of JADE installed on the same workstation, if the files are in different directories.
If ODBC is installed, only the last installation of the JADE ODBC driver is available from the ODBC Data Source Administrator.

If you install multiple copies of JADE 7.1 on a machine, an initial dialog during the installation asks if the installation should install a new instance or maintain or update the installed instance. You should select the default Install a new instance of this application option.

**JADE Thin Client Upgrade**

When upgrading a presentation client to JADE release 7.1:

- Presentation client upgrades from JADE release 6.3.04 or earlier are rejected.
- Reverting from JADE 7.1 to JADE 6.2 is rejected.
- If you are upgrading presentation clients to JADE release 7.1, ensure that you have the appropriate privileges or capabilities to install applications. The configuration of User Account Control (UAC) and your current user account privileges may affect the behavior of the upgrade to JADE 7.1. For details about UACs, standard user accounts, and administrator accounts, see the Microsoft documentation.

If JADE is installed in the \Program Files directory (or \Program Files (x86) directory on a 64-bit machine with 32-bit JADE binaries):

- If the machine has had UAC disabled, the thin client upgrade will fail because of lack of permissions for standard users. For administration users, the necessary privileges are automatically granted so the upgrade will succeed.
- If UAC is not disabled, administrative users are prompted with an Allow or a Cancel choice but standard users must know and supply the user name and password of a user with administrative privileges to enable the upgrade to succeed.

For more details, see Appendix B, "Upgrading Software on Presentation Clients", in the *JADE Thin Client Guide*.

**Upgrading a 32-bit Presentation Client Connecting to a 64-bit Application Server**

For versions prior to JADE 7.1 and 7.0.11, when a 32-bit presentation client connects to a 64-bit application server, the application server upgrades the version of the presentation client but it does not change the 32-bit to 64-bit type of the presentation client, because:

- The presentation client does not check to see if the operating system on which it is running is 64-bit-capable (and it would have to inform the application server about this).
- Any support libraries needed by the presentation client (for example, ActiveX control and automation libraries) would also have to be downloaded or already installed in the presentation client.

By default, a 32-bit presentation client will be upgraded to a version requiring the Microsoft Windows Visual Studio 2013 C++ runtime binaries.

**Notes** Visual C++ runtimes are always upgraded (that is, the x64 version is installed) as part of the upgrade process.

JADE 7.1 does not support 32-bit presentation clients using the Visual Studio 2005 C++ runtime binaries that were required by JADE 6.3 releases.

In JADE 7.1, 32-bit presentation clients can be upgraded to 64-bit, as follows.

- The UpgradeRuntimeTo64bit parameter in the [JadeAppServer] section only on each application server, when set to true, causes a 32-bit client running on a 64-bit operating system (WoW64) to be upgraded to 64-bit binaries.

---

Copyright 2017, Jade Software Corporation Ltd. All rights reserved JADE-7.1.09
If the client is not running a JADE version that is using Visual Studio 2005 runtime libraries, the JADE thin client download will not revert to a version that uses Visual Studio 2005 runtime libraries.

For more details, see "Upgrading 32-bit Thin Client Binaries to 64-bit (NFS 61726)", later in this document.

**Caution** The UpgradeRuntimeTo64bit parameter takes effect only after the client binaries have downloaded the release that supports this option. This requires two thin client downloads to achieve the 32-bit to 64-bit upgrade (that is, one download to update to JADE 7.1, and a second to upgrade to 64-bit).

### Upgrading an SDS Native or RPS Secondary System

When upgrading to JADE 7.1 from a JADE 6.3 or 7.0 release, the Synchronized Database Service (SDS) native or Relational Population Service (RPS) secondary must always be re-cloned from the upgraded primary database.

On an RPS secondary system, it is possible to retain the SQL Server database providing that the upgrade is done as a high availability upgrade. For details, see the HighAvailabilityUpgradeSteps document.

**Note** The ReadmeInstallSteps.pdf and HighAvailabilityUpgradeSteps.pdf documents are installed in the \documentation folder by the InstallShield.

### Upgrade Validation

During the upgrade process, a validation script is run to check the integrity of the upgraded system. Any user schema entities that conflict with system schema entities are logged as errors in the jommsgn.log file.

All errors must be corrected and validation re-run before user applications can be executed on the updated system. If the system is in the un-validated state, a message box is displayed when you log on to the JADE development environment, asking if validation should be re-run.

### Hot Fix Releases

Hot fixes for JADE system files are released as binary files.

To apply the hot fix:

1. Shut down the system.
2. Copy the hot fix system files into the appropriate directory.
3. Start up the system.

**Caution** You must apply all of the files contained in the hot fix at the same time.

It is important to ensure that versions of JADE system files do not diverge from dependent binaries. Doing this ensures that dependent code files (JADE system files and libraries) are backed up and restored together. The default location of the JADE system files is the installation directory (that is, the bin directory).

When it is necessary to restore a database from backup and perform recovery, you must avoid reverting to earlier JADE system file and binary versions. When restoring the binaries directory, ensure that it is from the latest backup.
JADE 7.1 Changes that May Affect Your Existing Systems

This section describes only the changes in the JADE 7.1 release that may affect your existing systems. Some changes may result in compile errors during the load process, or cause your JADE release 7.1 systems to behave differently.

For details about the changes in JADE releases 7.1.09, 7.1.08, 7.1.07, 7.1.06, 7.1.05, 7.1.04, and changes and new features in JADE 7.1.03, see "Changes in JADE Release 7.1.09", "Changes in JADE Release 7.1.08", "Changes in JADE Release 7.1.07", "Changes in JADE Release 7.1.06", "Changes in JADE Release 7.1.05", "Changes in JADE Release 7.1.04", and "Changes and New Features in JADE Release 7.1.03".

Compact JADE (PAR 61967)

With the release of Windows Mobile 6.5.3, Microsoft has moved the OK and X buttons from the top of a maximized form to the soft menu bar at the bottom of the screen. However, as JADE creates a user-defined menu bar and Windows does not add the OK or X button to that type of menu bar, users could not close a maximized form.

This has been changed as follows for all supported versions of Windows Mobile.

- The Compact JADE control panel configuration program (jadecfg.cpl) has been changed to add a permanent soft menu bar. Under Windows Mobile 6.5.3, this bar contains an OK button. For earlier Windows Mobile versions, the OK button is displayed in the caption line of forms.

This enables users to commit configuration changes under version Windows Mobile 6.5.3.

- The jade.exe program has been changed for all Windows Mobile versions so that it creates a standard Windows soft menu bar if there are two or fewer visible top-level menu items and they do not have images defined. Those top-level menus are assigned to the soft menu buttons on the menu bar (a maximum of two are available).

Under Windows Mobile 6.5.3, the menu bar also contains the Windows button and an X or OK button, if required, for a maximized form. For earlier Windows versions, the X or OK button continues to be displayed in the caption line of the form. If there are more than two visible top-level menu items or a top-level item has a defined image, a standard Windows soft menu bar cannot be created. The menu bar is user-defined, and under Windows Mobile 6.5.3, there is no X or OK button displayed.

**Notes** In this situation, your application has to include a user-defined menu item that performs the close functionality using JADE logic. The alternative solution is to redefine the menu structure so that there are two top-level menu items only.

With the use of a standard Windows menu bar, a soft menu item button displays only a small amount of text. If the text is too long, the text is truncated with trailing points of ellipsis (…).

If you want your application to retain the previous JADE menu bar style:

1. Ensure that the form is created with three or more visible top-level menu items (or one with an image).
2. In the load event of the form, hide any menu items that do not need to be shown. If the number of menu items is decreased to two or fewer, the menu bar style remains as user-defined.

However, if the number of visible top-level items is increased beyond two, Windows changes the menu bar to a user-defined style and the X or OK button will again not be displayed.

JADE Remote Node Access (PAR 60364)

When the Run as Service check box is unchecked when the JADE server node is still running as a service, a message box is now displayed, advising you that you cannot uncheck this option when the service is running and that you must first stop the service.
In addition, the service state (for example, running, stopped, and so on) is now continuously updated to reflect
service state changes made externally; for example, by using the Task Manager.

Note The same behavior applies also to the JADE application server (jadapp.exe).

See also "JADE Remote Node Access Utility", under "Changes and New Features in JADE Release 7.1.03".

Relational Population Service (RPS) Mapping

If you use the OID Mapping option of Map to String on the Define RPS sheet of the Relational Population Service
wizard, to conform with the JADE 7.1 size requirements for object identifiers, you must upgrade your mapping to
the Map to String (7.1 format) value. (See also "Class Number Extensions" under "Changes and New Features in
JADE Release 7.1.03".)

Replayable Reorganization (NFS 60428)

In the batch JADE Schema Load utility (jadloaddb), the handling of the optional replayableReorg parameter has
changed. The replayableReorg parameter now defaults to the value of the EnableArchivalRecovery parameter
in the [PersistentDb] section of the JADE initialization file, and is constrained by the setting of the EnableArchivalRecovery parameter for the database.

Set the value of the replayableReorg parameter to false if you do not want to perform a replayable reorganization
of your JADE database when the schema is loaded. If you set the parameter to true or you let it use the default
value, the parameter is set to the value of the database EnableArchivalRecovery parameter.

SOAP XML Handling in Web Services (PAR 60987)

When handling Simple Object Access Protocol (SOAP) Extensible Markup Language (XML) in Web services in
earlier releases:

- TimeStampOffset JADE type values were encoded in the generated XML with the wrong sign for the
  Coordinated Universal Time (UTC) bias; that is, - instead of + and + instead of -. The JADE SOAP parser also
  expected the wrong sign to establish a passed value. This meant that JADE to JADE handling was be
correct, but C# to JADE communication resulted in the wrong value.

- A negative TimeStampInterval JADE type value ignored the negative sign when the value was parsed by
  JADE, and the interval was always positive. In addition, the XML constructed by JADE did not include a
  negative sign for a negative TimeStampInterval value.

The TimeStampOffset value now has the correct sign for UTC bias and the TimeStampInterval value negative
sign is now recognized by the JADE parser and included in the XML constructed by JADE.

String Arrays and Dictionary Keys

As the 30-character limit for schema entity names has increased in JADE 7.1 to the maximum of 100 characters
(for details, see "Entity Name Maximum Length"), the StringArray class now provides the JADEIdentifierArray
subclass, to enable you to store entities instead of in the StringArray subclass, which will no longer be sufficient
for storing entities. (The membership string size for a StringArray is 62 and that of the new JADEIdentifierArray is
100.)

If you are using entity names as dictionary keys, you must also consider the size of your dictionary key, as the total
size of the key cannot exceed 512 key units. A key unit is a byte for any non-character data type, or one character
for any character data type; that is, key sizes are string-encoding agnostic. Key sizes also must allow for a null
character to terminate any strings; characters are not null-terminated.
Upgrading Collections (PAR 63704)

In earlier releases, some collection meta data was not updated correctly during the upgrade process, which affected the membership size of a collections other than primitive Arrays, and the keys of external key dictionaries. The incorrect meta data did not cause errors when the collections were used, but could cause unnecessary reorganizations and the disk utilization of collection blocks were not optimal, which could result in larger files that contained a lot of large collections.

You can correct the meta data in a system that has already been upgraded, by running the following `jadclient` non-GUI client application.

```
bin\jadclient.exe path=xxx\system ini=xxx\jade.ini server=SingleUser
schema=RootSchema app=RootSchemaApp executeClass=JadeUpgradeAdmin
executeMethod=upgrade_Generate_Coll_ABEs
```
Changes in JADE Release 7.1.09

This section contains details about product and documentation changes in JADE release 7.1.09. See also:

- "Upgrading Collections (PAR 63704)" under "JADE 7.1 Changes that May Affect Your Existing Systems"
- "Binary Type pos Method Deprecation" under "Deimplementations and Deprecations"

For details about release 7.1.03 (the first general release of JADE 7.1) and changes in releases 7.1.04, 7.1.05, 7.1.06, 7.1.07, and 7.1.08, see "JADE 7.1 Changes that May Affect Your Existing Systems", "Changes and New Features in JADE Release 7.1.03", "Changes in JADE Release 7.1.04", "Changes in JADE Release 7.1.05", "Changes in JADE Release 7.1.06", "Changes in JADE Release 7.1.07", and "Changes in JADE Release 7.1.08", elsewhere in this document.

JADE Logical Certifier Utility (NFS 63393)

In earlier releases, you could not run the meta certify operation in multiuser mode.

Although the JADE Logical Certifier utility must be run in single user mode when certifying user schemas, you can now perform meta certify and repair operations in multiuser mode.

**Note**  It is not advisable to perform repair operations to meta data in multiuser mode.

JadeTestCase Class Assert Methods (PAR 64825)

The type of the parameters for the expected and actual values of the JadeTestCase class assertEquals method are declared as Any, which provides a general API that can be used to test for values of all JADE data types. However, this means that the compiler cannot do any type checking of the values. In JADE, two type Any variables are always different if the type of the value differs; for example, a1 is not equal to a2 if a1 and a2 have different types.

In the example `assertEquals(1, 1.Decimal)`, the first parameter (1) has type Integer with a value of 1 while the second parameter (1.Decimal) has type Decimal with a value of 1, so the assert will always fail. The assert should be coded as follows:

```java
assertEquals(1.Decimal, 1.Decimal);
```

The message output by the JadeTestCase class assertEquals method if the assert fails because of the operands having different types has now been changed to make this clear; that is:

```java
assertEquals - types differ, expected type-of=m value value-of=m but actual
type-of=n value = value-of=n
```

The JadeTestCase class assertEqualsMsg method has also been changed accordingly.

Presentation Client Installer Customization to be Deprecated

The Customizable Presentation Client Installer (CPCI) is to be deprecated in JADE 2016.

Resizing Dialogs (PAR 64258)

To cater for larger class and method names, the following dialogs can now be resized.

- Currency Format
- Long Date Format
- Short Date Format
- Number Format
REST Services

This section describes the REST Services changes in this release.

Generated REST Service Description (PAR 64969)

In earlier releases, the description generated from a REST Service class wrongly included all of the subclasses of the classes referenced in the JadeRestService subclasses communication methods. This produced a file that included a very large number of other classes.

The description now includes only the classes and the superclasses referenced by the specified parameter values and return types of the communications methods of the JadeRestService subclass selected for the application. Note, however, that no class description is included if those methods return only Any or a primitive type.

REST Service Call of Protected or Read-only Properties (NFS 64481)

If access is protected or read-only of a property of an input object to a REST service call, REST does not set the property.

RPS Keywords (PAR 64608)

In earlier releases, reserved T-SQL keywords (for example, the word ‘file’ in the query UPDATE BKdocument SET file=? WHERE oid = ?) were not quoted in RPS.

The third sheet of the System DSN wizard in the ODBC Data Source Administrator (the Data Sources (ODBC) program in the Windows Administrative Tools) now has the configurable Use ANSI quoted identifiers check box, which enables RPS to interrogate the RPS ODBC data source that it uses to determine if to quote SQL identifiers (that is, table, column, and procedure names). This allows reserved T-SQL keywords to be used in SQL statements, which is useful if you want a property to have the same name as a T-SQL reserved word.

SDS Database Role (PAR 63603)

In earlier releases, the JADE Database utility batch clearRole command could be performed on a backup of a secondary system leaving it in an inconsistent state.

It is generally unsafe to clear the role of a secondary system, as they almost always require recovery. You should clear the role by specifying the clearSDSRole parameter in the recovery from backup. If you do not specify the clearSDSRole parameter in the recovery from backup, the recovery operation is recovering a secondary database as a secondary. This means that:

1. No undo will be performed
2. A partial final journal will not be processed (when the secondary initializes, it re-fetches the journal from the primary)

The expectation is that the database will connect to the primary and then synchronize. When the last journal is not applied when recovering from a backup, the backup recovery is incomplete and file integrity is not restored. The undo action is suppressed, so there are further complications if there are incomplete transactions.

This has been corrected. The clearRole command is no longer permitted on SDS Secondary systems, and the jdbutil and jdbutilb executables showinfo command now always shows SDS Secondary systems as requiring recovery.
Upgrade Process Error 8724 (PAR 64710)

The upgrade process now tests for the existence of the JadeReportWriterSchema schema before the migration begins. When the upgrade process starts up, if it finds the JadeReportWriterSchema schema in the JADE system that is to be upgraded, error 8724 (Cannot upgrade if JadeReportWriterSchema is present) is now raised and the upgrade is aborted.

For details, see steps 4 and 22, respectively, in the upgrade instruction under "Upgrading to JADE 7.1 from an Earlier Release".

Using the JADE Report Writer White Paper

The JADE product information library now contains the Using the JADE Report Writer White Paper, which provides an understanding of how to use the JADE Report Writer to improve performance during execution of user reports.
Changes in JADE Release 7.1.08

This section contains details about product and documentation changes in JADE release 7.1.08. See also:

- "Upgrading Collections (PAR 63704)" under "JADE 7.1 Changes that May Affect Your Existing Systems"
- "Binary Type pos Method Deprecation" under "Deimplementations and Deprecations"

For details about release 7.1.03 (the first general release of JADE 7.1) and changes in releases 7.1.04, 7.1.05, 7.1.06 and 7.1.07, see "JADE 7.1 Changes that May Affect Your Existing Systems", "Changes and New Features in JADE Release 7.1.03", "Changes in JADE Release 7.1.04", "Changes in JADE Release 7.1.05", "Changes in JADE Release 7.1.06", and "Changes in JADE Release 7.1.07", elsewhere in this document.

.NET

This section describes the .NET changes in this release.

.NET Decimal Conversion (PAR 63449)

When converting a Common Language Runtime (CLR) Decimal data type to a JADE Decimal primitive type in earlier releases, the wrong value was written to a JADE property from a .NET or ActiveX decimal if the value contained 29 significant digits. The algorithm used to round a decimal value was wrong if the value of the number represented in the significant digits, ignoring the decimal point, was greater than or equal to $2^{95}$ (39,614,081,257,132,168,796,771,975,168), which affected .NET and ActiveX decimal values.

Appendix A of the JADE .NET Developer's Reference stated that when converting a CLR decimal data type to a JADE Decimal primitive type, there may be an overflow or data loss saving data to the database. From JADE 7.0.11 and 7.1.07, this is no longer true. If necessary, JADE rounds decimal values on assignment to the number of decimal places defined on the property.

.NET Import and Usage (PAR 63621)

In earlier releases, scaling issues could occur when a .NET control was loaded, as Windows assumed that JADE was not dots per inch (dpi)-aware. JADE is dpi-aware if the value of the Form class scaleForm property is true, so that JADE scales the forms according to the current system dpi. However, when a .NET assembly is subsequently loaded, the Microsoft dpi-awareness handling is invoked and it is assumed that the forms need to be rescaled using its virtualization philosophy, which distorts the displayed forms.

On Windows 8.1 and above, JADE now informs the operating system that it is dpi-aware (system-wide; not the scaling of each dpi monitor), which prevents Windows from attempting to scale again.

Advanced Notice of JADE 2016 Deprecation

From the JADE 2016 release, Compact JADE will no longer be available; that is, support for JADE databases running on Windows Mobile devices will no longer be available.

Compact JADE in thin client mode will continue to be available for the lifetime of JADE 7.1.

Context-Sensitive Help to HTML5 Topics

Earlier releases of JADE provided context-sensitive help from the development environment to the required Portable Document Format .pdf file.

With the provision of the full product information library in both PDF (print) and HTML5 (Web) formats, from JADE release 7.1.07, you can now specify that context-sensitive help is obtained from .htm topics in the HTML5 Web format of the product information.
Context-sensitive help to HTML5 topics is controlled by a new `UseJadeWebHelp` parameter in the [JadeHelp] section of the JADE initialization file. This parameter is `true` by default, in which case it reads the new `JadeHelpBaseUrl` parameter. If a value is specified for the `JadeHelpBaseUrl` parameter, it uses that URL. If the value is `<default>` or it is empty, the URL is determined by the internal hard-coded URL for the current release. For example, the [JadeHelp] section of the JADE initialization file could contain the following parameter values.

```
[JadeHelp]
UseJadeWebHelp = true
JadeHelpBaseUrl = https://www.jadeworld.com/docs/jade-71/Default.htm
#
```

Where the `.htm` extension is used, `.html` is also valid.

Set the value of the `UseJadeWebHelp` parameter to `false` if you want to continue using context-sensitive help to specific sections in the appropriate PDF files.

**Find Type Dialog (NFS 63581)**

The Find Type dialog (accessed using the F4 function key) is now larger and can be resized.

**IndexLoadFactor Parameter in the [PersistentDb] Section (PAR 64067)**

The [PersistentDb] section of the JADE initialization file now contains the `IndexLoadFactor` parameter, which specifies the percentage of entries that are retained when an index block becomes full and is split. A value of 95 means that 95 percent of entries are retained in the current block and 5 percent of entries are moved to a new block.

Statistically, a 66 percent load factor (the default value) provides optimal loading when entries are added in random key order and a higher load factor (for example, 95 percent) provides better loading when entries are added in sequential key order.

The minimum parameter value is 50, and the maximum parameter value is 95.

**Node Object Type Values (PAR 64128)**

In earlier releases, the table listing the type of the node object returned by the `Node` class `nodeType` method was incomplete. The following table lists all node types that can be returned by this method.

<table>
<thead>
<tr>
<th>Integer Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Undefined</td>
</tr>
<tr>
<td>1</td>
<td>Database server (jadrap or jadserv)</td>
</tr>
<tr>
<td>2</td>
<td>Application server (jadapp or jadappb in multiuser mode)</td>
</tr>
<tr>
<td>APPLICATION_SERVER + DATABASE_SERVER</td>
<td>Application server and database server (jadapp or jadappb in single user mode)</td>
</tr>
<tr>
<td>4</td>
<td>Standard client node (jade in multiuser mode; not as a thin client)</td>
</tr>
<tr>
<td>STANDARD_CLIENT_NODE + DATABASE_SERVER</td>
<td>Standard client node and database server (jade in single user mode)</td>
</tr>
<tr>
<td>16</td>
<td>Non-GUI (jadclient) node</td>
</tr>
<tr>
<td>JADCLIENT_NODE + DATABASE_SERVER</td>
<td>Non-GUI (jadclient) node and database server</td>
</tr>
<tr>
<td>32</td>
<td>Database administration (jdbadmin) node</td>
</tr>
<tr>
<td>DBADMIN_NODE + DATABASE_SERVER</td>
<td>Database administration (jdbadmin) node and database server</td>
</tr>
</tbody>
</table>
Non-GUI nodes include user-written executables that use the JADE Object Manager API (C++) and the JADE .NET API (C#).

**Object Class autoPartitionIndex Method (PAR 63695)**

The Object class `autoPartitionIndex` method cannot be used if the database file for that object is encrypted, as the database cannot invoke the `autoPartitionIndex` method using an encrypted buffer. If this occurs, exception 3009 *(File encrypted and partition unspecified)* is raised.

If the file is encrypted, use the Object class `setPartitionID` or `setPartitionIndex` method to explicitly set the partition in the created object.

**Scaling the Splash Screen (PAR 63654)**

In earlier releases, the JADE splash screen was not scaled under different monitor dots per inch (dpi) settings, so that the text drawn for different items on the screen could overlap and was not presented well.

The `StatusPos`, `ApplicationPos`, `SchemaPos`, `ServerPos`, `PathPos`, `VersionPos`, and `AviPos` splash screen parameter values in the [Jade] section of the JADE initialization file are now based on 96 dpi. The splash screen positions are scaled if the user has a different dpi setting.

**Secure Sockets Library (PAR 63548, NFS 63550)**

JADE supports the 1.0.2g-level OpenSSL libraries, which removes support for the insecure SSLv2 method. In addition, JADE has removed support for the insecure SSLv3 method.

From JADE 7.1.07, JADE accepts connections using only TLSv1 or above. The TLSv1.2 method is now the JADE default value. This has the following effects on existing JADE systems.

- If a value of `SSLv2`, `SSLv23`, or `SSLv3` in the `SSLMethoName` parameter is present in the [JadeAppServer] or [JadeThinClient] section of an existing JADE initialization file, it is overwritten with `<default>`, a message is written to the `jommsg` log file, and JADE attempts to make an SSL connection using the new `TLSv1.2` default method. This connection can fail for the following reasons.
  - Existing X509 certificates can be rejected by the OpenSSL libraries if they are incompatible with the upgraded requirements of `TLSv1` or higher.
  - Existing connections can be refused if the explicit list of ciphers defined in the JADE initialization file are incompatible with the upgraded requirements of `TLSv1` and higher.

  **Note** As the previously distributed versions of the `server.pem` and `client.pem` OpenSSL insecure example certificates are subject to this condition, a new version of the example certificates is provided with this release.

  **Tip** Unless you have special requirements, leave the value of the `SSLCipherNames` parameter blank, to use the default, compatible list of ciphers provided by the OpenSSL libraries.

To enhance SSL security, the default values of the `SSLMethoName` and `SSLCipherNames` parameters in the [JadeAppServer] and [JadeThinClient] sections of the JADE initialization file are as follows.

- `SSLMethoName=TLSv1.2`
- `SSLCipherNames=Not specified` (compatible ciphers are available from the OpenSSL online documentation or `openssl.exe`
Table Sorting (PAR 63893)

The handling of the Table class sortType property that converts from a string to the specified sort type is based on the current locale used by the application and the value of the EnhancedLocaleSupport parameter in the [JadeEnvironment] section of the JADE initialization file.

In earlier releases in a Windows 10 environment, a change of the default time formatting for New Zealand to use AM/PM format meant that the conversion of a String primitive type to a Time primitive type failed using some JADE methods if the String time included the 'AM/PM' format (in particular, sorting a Table control by a time-based column failed to sort correctly). JADE now handles an 'AM/PM' string format.

In addition, the Table class sorting for Date and Timestamp primitive type-based columns has also changed. In earlier releases, the strings were converted to a Timestamp or Date primitive type using the current locale format. If the string was not in the appropriate format, the sorting failed to produce the required sequence. If the conversion of the date part fails based on the locale format, JADE now attempts to determine the format (d/M/y, M/d/y, or y/M/d) based on the type of data (that is, year length = 4 and whether the month is alpha).

Upgrade Validation (PAR 63810)

In a major JADE release, some changes to the JADE model or interpreter engine require methods that make use of the changed feature to be rebuilt, by recompiling the methods. The upgrade process fails if methods cannot be recompiled.

In earlier releases, the failure of any method to compile failed the upgrade validation and the process exited with error 8701 (General Upgrade error code). The jadloadb or jadclient upgrade process now exits with error 8711 (Uncompiled or in error methods were detected in this database) if recompiling one or methods fails, making it clear that you must reload some methods and restart the validation.
Changes in JADE Release 7.1.07

This section contains details about product and documentation changes in JADE release 7.1.07.

For details about release 7.1.03 (the first general release of JADE 7.1) and changes in releases 7.1.04, 7.1.05, and 7.1.06, see "JADE 7.1 Changes that May Affect Your Existing Systems", "Changes and New Features in JADE Release 7.1.03", "Changes in JADE Release 7.1.04", "Changes in JADE Release 7.1.05", and "Changes in JADE Release 7.1.06", elsewhere in this document.

See also "Binary Type pos Method Deprecation" under "Deimplementations and Deprecations".

Class Browser Displays the Schema Name (NFS 63547)

In earlier releases, the editor pane of the Class Browser included the schema name only if the class was a subschema copy class.

The editor pane now always includes the schema name in which the class is defined; for example:

Class: RootSchema::Form (189)

Downloading Additional Thin Client Binaries (PAR 63174)

When files and directories are downloaded to the presentation client from the application server, the directory structure under the application server download directory (for example, ...download\jade.exe) for the architecture of the presentation client is duplicated in the JADE installation directory on the presentation client. The files in those directories on the application server are downloaded and copied to the equivalent directory on the presentation client. (The directories are created if they are not present.)

The exception to this, however, is the directory that contains the jade.exe executable on the presentation client and the application server download directory that contains the jade.exe executable, as those directories are assumed to be the equivalent bin directories in both environments. The result is that all of the files in the download bin directory of the application server are copied to the presentation client bin directory, even if the names are different. In addition all subdirectories of the application server bin directory are copied as subdirectories of the presentation client bin directory. For example, the:

- Application server has:

  ......\download\bin\ (which contains jade.exe)

  ......\download\bin\sub-bin\

- Presentation client has:

  <JADE-install-directory>\mybin\ (which contains jade.exe)

The result is that the files in bin are copied to mybin and a sub-bin directory is created as a subdirectory of mybin.

Exception Dialog (NFS 63168)

The error object reported by the default exception handler now includes the type name before the oid if the class number is valid; for example:

Error OID: 2922.1 (transient)

If there is no class in the current system that has the specified class number, only the oid is displayed, as was the case in earlier releases.
External Function Calls (PAR 63250)

JADE now publishes external functions declared in the RootSchema jomos external function library that call Windows library functions as defined in the Microsoft Developer Network (MSDN). These external functions are declared to avoid having to declare ANSI- and Unicode-specific definitions for some commonly called Windows library functions; for example, josShellExecute calls ShellExecuteA in an ANSI environment and ShellExecuteW in a Unicode environment.

The following table maps functions defined in the RootSchema jomos external function library to the corresponding MSDN function.

<table>
<thead>
<tr>
<th>JADE External Function</th>
<th>Windows Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>josCreateDirectory</td>
<td>CreateDirectory</td>
<td>Creates a directory that inherits information from other directories</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(security attributes default to null)</td>
</tr>
<tr>
<td>josDeleteDirectory</td>
<td>RemoveDirectory</td>
<td>Deletes an existing empty directory</td>
</tr>
<tr>
<td>josFileAccess</td>
<td>GetFileAttributes</td>
<td>Retrieves attributes for a specified file or directory (for details, see</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the paragraph that follows this table)</td>
</tr>
<tr>
<td>josFileCopy</td>
<td>CopyFile</td>
<td>Copies an existing file to a new file</td>
</tr>
<tr>
<td>josFileDelete</td>
<td>DeleteFile</td>
<td>Deletes an existing file</td>
</tr>
<tr>
<td>josGetKeyState</td>
<td>GetKeyState</td>
<td>Retrieves the status of the specified key; that is, whether</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the key is up, down, or toggled</td>
</tr>
<tr>
<td>josGetLastError</td>
<td>GetLastError</td>
<td>Retrieves the last error code value of the calling thread</td>
</tr>
<tr>
<td>josShellExecute</td>
<td>ShellExecute</td>
<td>Opens or prints the specified file; for example, to start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>another program under Microsoft Windows</td>
</tr>
</tbody>
</table>

In addition to these functions, the josValidateDirectory external function is declared, which validates that the specified name is a directory. For more details, see Appendix B of the JADE External Interface Developer’s Reference.

The Windows class now provides the getWindowHandle method, which returns the Windows handle as a MemoryAddress. This method has the following signature.

```java
getWindowHandle(): MemoryAddress;
```

**Note**  If a node can execute as a 32-bit or 64-bit node, you should use this method rather than the Window class hwnd and getHwnd methods, to ensure that the correct Windows handle is used.

The signature of the josShellExecute method has been changed in JADE 7.1.07. Prior to JADE 7.1.07, the Windows handle parameter was declared as an Integer, and it is now a MemoryAddress. A similar change has been made to the unpublished _getCurrentThread, _getThreadCPUTimes, _shellExecute, and _shellExecuteUnicode methods. All of these functions now use a MemoryAddress rather than an Integer for the Windows handle, to ensure that the correct value is used for the currently executing build type (that is, 32-bit or 64-bit). The upgrade process from a JADE 7.0 to a JADE 7.1 release has been changed to include a check for methods that reference these functions. Any such references will result in the method being marked for recompile during the upgrade validation phase. You must correct the references before restarting the validation process.

You must manually check and correct any references in systems that are already running on a JADE 7.1 release.

**FileFolder::browseForFolder Method (NFS 63418)**

The FileFolder class browseForFolder method now includes the Folder text box below the directory list box, so that the user can specify the name of the required folder, using a standard (common) Microsoft dialog.
Imported Control Class Event Truncation (PAR 63460)

Although the maximum size of identifier names was increased to 100 characters in JADE 7.1, the .NET and ActiveX import process restricted the length of imported control events names to 14 characters. When a control is added to a form, event method names became control-name_event-name, so if the event name was long and not truncated, to construct a valid event method name meant that the control name had to be unacceptably short.

From this release, imported control event names are restricted and truncated to 49 characters, which allows a control name to be up to 50 characters.

**Note**  
Reloading an assembly that uses the old naming schema retains the previously assigned names so that your existing logic is not affected. As a result, you must delete the assembly and reload it, to get the new naming convention.

JADE Debugger (PAR 46099)

After re-compiling a method while using the JADE debugger, the debugger continues to display the old method source under the following circumstances.

- The application is displaying a modal form.
- The debugger next stops on a breakpoint in the same method that was compiled and that method was the last displayed by the debugger.

When the application has stopped on a breakpoint and the **Continue** action is performed, the debugger clears the debugger displays, including the last method shown when the execution stack becomes empty.

If a modal form is displayed, the execution stack does not become empty and the debugger displays are not cleared. If the next breakpoint encountered is in the same method, the debugger retains the currently displayed method.

When the application becomes idle and a modal form is displayed, the debugger clears the debugger displays and the new method source is displayed when a breakpoint is next encountered in that method.

**Notes**  
If the method is changed and re-compiled while the debugger has stopped on a breakpoint in that method (or in another method while that method is on the call stack), the interpreter will not reload that method logic until all copies of the method execution have exited from that method.

The debugger continues to display the old source while breakpoints are only encountered in that method until the application execution again becomes idle.

After a breakpoint is encountered in any other method, any breakpoint subsequently encountered in the changed method displays the new source, regardless of whether the old or new version of the method is being executed.

JadeRecompileAllMethods Application Errors (PAR 63227)

The **JadeRecompileAllMethods** application, run from `jadclient`, now returns a non-zero exit code 1183 (*Uncompiled or in error methods remain*) if methods could not be compiled or if any methods did not compile without error.

In addition, the output is also now logged to the **JadeRecompileAllMethods.log** file in the default log directory. The log file contains details of the methods that were not successfully compiled.
Logical Certifier Orphan Block Checking (PAR 63378)

The Logical Certifier now handles deleting collections that have missing blocks. Orphan blocks may be created when a dictionary with missing blocks is deleted using a delete or an orphan repair. If any unreachable blocks exist prior to the collection being deleted, they are not reported as orphans until after the collection header is deleted. A message is logged to the Logical Certify repair.log file if a collection that contains missing blocks is deleted. You should run the Logical Certifier again, to determine whether any orphan collection blocks have been created.

The Logical Certifier now provides the following error codes for the orphan checks.

- **Error 40 - Orphan dictionary block**
  This error is detected when a dictionary block is found but the parent instance that owns the dictionary does not exist.
  Repair:
  ```
  orphanBlock filename
  ```

- **Error 41 - Orphan blob/slob**
  This error is detected when a blob or slob subobject is found but the parent instance that owns the blob or slob does not exist.
  Repair:
  ```
  orphanSlobOrBlob filename
  ```

- **Error 42 - Orphan dynamic property cluster**
  This error is detected when a dynamic property cluster is found but the parent instance that owns the cluster does not exist.
  Repair:
  ```
  orphanCluster filename
  ```

- **Error 43 - Orphan subobject (collection)**
  This error is detected when a collection subobject is found but the parent instance that owns the collection does not exist.
  Repair:
  ```
  orphan oid
  ```

Method Signature Change (PAR 63453)

The signature of the following methods has changed. The type of the pNames or names parameter that was StringArray in earlier releases is now type JadeIdentifierArray (which may cause a warning when upgrading to this release from a JADE 7.0 release).

- `Schema::getWebServiceConsumerNames`
- `JadeAuditAccess::getClassPropertyNames`
- `JadeAuditAccess::getChangedPropertyNames`

Picture Class rotation Property (PAR 63420)

The description of the Picture class rotation property is updated to provide more information, as follows.
- The picture is sized according to the size of the picture and the value of the stretch property setting (ignoring the value of the rotation property).

- The picture is then drawn to that size and rotated about the central point of the control. Any parts of the picture outside the control are clipped.

As a result, the best way to handle the rotation property is to:

1. Construct the picture in another Picture control to the required size without rotation, with the appropriate stretch property value and control size.

2. Use the Window class createPictureAsType method to obtain the binary picture value from the Picture control that you created in the previous step of this instruction.

   If you call the createPictureAsType method with a bit value less than the bit value of the original image, color distortion can occur.

3. Resize the destination Picture control so that it fits the rotated picture. (Set the destination stretch property value to Stretch_None and the picture property to the binary value returned from createPictureAsType method.)

In addition, creating a picture with the pictureType method set to PictureType_Jpeg results in picture image quality reduction, because the JPEG format is lossy; that is, picture quality is reduced to lessen the size of the resulting image. To retain existing quality and produce a smaller binary image, use a picture type such as PictureType_Png, which has a lossless compression of images for greater clarity.

**Note** Any stretching of an image can cause image distortion.

### Relational Population Service OID Mapping (NFS 62231)

In earlier releases, the default value in the OID Mapping Options combo box on the Define RPS sheet of the Relational Population Service wizard was Map to String.

As the maximum length of JADE entities increased from 30 to 100 in JADE 7.1, JADE now supports two RPS object identifier (oid) length mapping options: Map to String (7.0 format) and Map to String (7.1 format). The 7.0 format size is 16, and the 7.1 format size is 28.

If you want to use the recommended 7.1 format oid string mapping in a new RPS mapping, select the Map to String (7.1 format) option.

**Notes** If you select the Map to String (7.1 format) option in an existing RPS mapping, it causes a reorganization and the generation of table alter scripts.

You cannot extract a schema from a system with the Map to String (7.1 format) option set and load it into a JADE system that has a different oid mapping. (As the schema load will fail, you must upgrade the system into which you want to load the schema.)

### REST Services

This section describes the REST service changes in this release.

### Error if Unknown JSON Type (NFS 63335)

JADE now provides error 11153 *(The type found in the Json is unknown)*, which is caused when a type specified in the JSON is not a known JADE class.

If this error is raised, fix the task generating the JSON so that it specifies the correct JADE class.

### JadeRestService::getServerVariable (PAR 63247)

The JadeRestService class now provides the getServerVariable method, which has the following signature.
getServerVariable(var: String): String;

This method returns the specified HTTP header information for your REST service request from the Internet Information Server (IIS). This method must be called during the processing of a REST service message; for example, from a re-implementation of the JadeRestService class processRequest method.

Calling the method when a message is not being processed results in null always being returned. In addition, the method must be called on the same node as the application. If you call the method from a server method and the application is not running on the server, a 31039 error (Connection invalid invocation) occurs when trying to access the TCP/IP connection, and error 1242 (A method executing in another node was aborted) is reported to the REST service.

As the var parameter is IIS-dependent, it is therefore subject to change. Refer to the ServerVariables function in your Internet Information Services (IIS) documentation for details. Common server environment variables, documented in the IIS documentation under the ServerVariables function, include those listed in the following table.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Returns…</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP_ACCEPT_LANGUAGE</td>
<td>A string describing the language to use for displaying content</td>
</tr>
<tr>
<td>HTTP_USER_AGENT</td>
<td>A string describing the browser that sent the request</td>
</tr>
<tr>
<td>HTTPS</td>
<td>ON if the request came in through a secure channel (SSL) or it returns OFF if the request is for a non-secure channel</td>
</tr>
<tr>
<td>REMOTE_ADDR</td>
<td>IP address of the remote host making the request</td>
</tr>
<tr>
<td>SERVER_NAME</td>
<td>Host name, DNS alias, or IP address of the server as it would appear in self-referencing URLs</td>
</tr>
<tr>
<td>SERVER_PORT</td>
<td>Port number to which the request was sent</td>
</tr>
<tr>
<td>URL</td>
<td>Base portion of the URL</td>
</tr>
</tbody>
</table>

The method in the following example returns the IP address of the REST service as determined by IIS.

```java
processRequest(httpIn: String; queryStr: String; pathIn: String; methodType: String) updating;
vars
  str : String;
begin
  str := self.getServerVariable("ALL_HTTP");
  inheritMethod(httpIn, queryStr, pathIn, methodType);
end;
```

**Parsing JSON Text (NFS 63367)**

Standard JSON syntax can include the type of the objects to create; for example:

```json
/> "_type": "Customer", .. in Microsoft JSON format
/> "$type": "Customer", .. in Newtonsoft (JSON.NET) format
```

This "_type" special type tag must appear as the first entry following the { or [ symbol that begins the object contents description in Microsoft format. In Newtonsoft, the "$type" tag can appear after the object reference tag; for example:

```
"id": "12", "$type": "Customer",
```

In Newtonsoft, the type can also include a namespace such as "MyNameSpace.Customer", which JADE will ignore.

If the JSON does not include the type tag, the type of the object is assumed to be the implied type of the expected object; for example, the type of the method parameter to be populated or the type of the property reference.
If the JSON includes the type tag, the type must be the implied type or a subclass of the implied type. If it is not, an exception is generated.

**ThreadPriority Parameter Removal (PAR 63251)**

The `ThreadPriority` parameter has been removed from the `[JadeClient]` and `[JadeServer]` sections of the JADE initialization file.

Because JADE was incorrectly overriding a below-normal thread priority, this parameter has been removed on client and server nodes, as standard management tools exist to set the default thread priority.

**Time Millisecond Value (PAR 63572)**

The millisecond value of a time expressed as a string represents a fraction of a second, but in earlier releases, JADE assumed it was the number of milliseconds. Any value that was greater than 999 was rejected as being invalid. Any value where the value was one or two digits was wrongly converted; for example, `.1` represents 1/10 of a second (100 milliseconds), where previously JADE would have treated it as 1 millisecond.

JADE now handles millisecond values greater than three digits.

**Upgrade Validation in a Source-Stripped Database (PAR 63458)**

In earlier releases, the upgrade validation failed if the source was not present or was encrypted for a method that needed to be recompiled.

When upgrading a database that has method source stripped, the methods that need to be recompiled are now marked in error and the upgrade process is allowed to complete and error 8703 (*Method could not be recompiled - no source present*) is raised, indicating that method source needs to be loaded and the methods recompiled. The list of methods that could not be compiled is output to the `jadeupgrade.log` file. When the methods marked as requiring source recompilation have been successfully recompiled, the upgrade of the system does not need to be re-validated after any subsequent schema loads if error 8703 is raised.

The upgrade process now recompiles encrypted methods and the upgrade does not fail if the re-compilation succeeds.
Changes in JADE Release 7.1.06

This section contains details about product and documentation changes in JADE release 7.1.06.

For details about release 7.1.03 (the first general release of JADE 7.1) and changes in releases 7.1.04 and 7.1.05, see "JADE 7.1 Changes that May Affect Your Existing Systems", "Changes and New Features in JADE Release 7.1.03", "Changes in JADE Release 7.1.04", and "Changes in JADE Release 7.1.05", elsewhere in this document.

See also "Binary Type pos Method Deprecation" under "Deimplementations and Deprecations".

Binary::uuidAsString Method (PAR 62807)

When calling the Binary primitive type uuidAsString method, to be a valid Universally Unique Identifier (UUID), the binary should be 16 bytes. If it is less than 16 bytes, the value will be internally padded with zero bytes on the end, to make it 16 bytes long before the conversion is performed.

If it is longer than 16 bytes, exception 1091 (Binary too long) is raised.

CMDPrint::warnIfNoDefault Property (PAR 62950)

In earlier releases, the default value of the CMDPrint class warnIfNoDefault property was documented as false. However, the default value is true; that is, a warning message box is displayed by default if there is no printer default for the system on the common Print dialog.

Date Handling (PAR 62888)

In earlier releases, the Thai calendar was the only non-Gregorian calendar handled by JADE.

JADE now handles any calendar available within Microsoft Windows. Non-Gregorian calendars are handled in the locale-based date routines such as parseForCurrentLocale, longFormat, shortFormat, and so on, as well as the setTextFromDate and getTextAsDate methods in the JadeEditMask class.

In addition, the Date primitive type isLeapYear method now uses the currently set locale and calendar of the user to determine whether the date is a leap year. (In earlier releases, this method returned whether the date was a leap year in terms of the effective Gregorian date.)

Global Find and Replace Handling (NFS 63056)

When you use the Edit menu Global Find/Replace command or the SHIFT+CTRL+F3 keys and JADE has located all occurrences of the text in the classes and schemas that match your search criteria, if the number of found occurrences within a method is greater than 1, the Global Find/Replace Results Browser now displays the number of occurrences of the searched text for a method in parentheses to the end of the method name; for example, AllOrderedDoodahs::findCustomer (2).

JadeHTTPConnection Class Constants (NFS 62889)

The JadeHTTPConnection class provides the following constants, which were not documented in earlier 7.1 releases.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type and Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Protocol</td>
<td>String = &quot;Protocol&quot;</td>
<td>Protocol header</td>
</tr>
<tr>
<td>State NoData</td>
<td>Integer = 6</td>
<td>No data available; not supported</td>
</tr>
<tr>
<td>Verb CONNECT</td>
<td>String = &quot;CONNECT&quot;</td>
<td>The CONNECT operation; not supported</td>
</tr>
<tr>
<td>Verb DELETE</td>
<td>String = &quot;DELETE&quot;</td>
<td>The DELETE operation; not supported</td>
</tr>
</tbody>
</table>
Name | Type and Value | Description
--- | --- | ---
Verb HEAD | String = "HEAD" | The HEAD operation; not supported
Verb OPTIONS | String = "OPTIONS" | The OPTIONS operation; not supported
Verb PUT | String = "PUT" | The PUT operation; not supported
Verb TRACE | String = "TRACE" | The TRACE operation; not supported

In addition, the existing **State Failure** class constant indicates a failed HTTP connection state; for example:

- Open failed
- Send request headers failed
- Invalid verb specified for send headers
- WinINet authentication failure
- WinINet set option failure
- WinINet open/connect failure

**JADE Logical Certifier Utility**

This section describes the JADE Logical Certifier utility changes in this release.

**JADE Logical Certifier Error 33 (PAR 62880)**

The JADE Logical Certifier utility now detects error 33 (*DynaDictionary incomplete or inconsistent*) when a *DynaDictionary* instance references a missing or inconsistent class or property definition; for example, when the membership class of the *DynaDictionary* is deleted, or when a property that is used as a member key property of a *DynaDictionary* is deleted. The repair action is:

```
delete oid
```

**Logically Certifying Meta Data (PAR 62689)**

In earlier releases, logically certifying meta data did not report invalid *VersionInfo* instances.

The logical certification of meta data now reports as errors versioned meta schema instances where the schema is invalid or it is not versioned. Error 99 is reported if any such instances are encountered (no fix is generated for this error).

**Note** This new check may encounter old errors in your user systems. For assistance with creating manual fixes to correct invalid versioned meta schema instances, contact JADE Support ([jadesupport@jadeworld.com](mailto:jadesupport@jadeworld.com)).

**JadeRichText Control (NFS 62749)**

You can now set the background color, the font underline type, and whether selected text is drawn as a URL in a *JadeRichText* control, by using the following *JadeRichText* control properties.

- **selBackColor**, which specifies the color of the background of the currently selected text.
- **selFontUnderlineType**, which specifies the underline style of the currently selected text (that is, one of none, line, dash, dash-dot, dash-dot-dot, dotted, thick, wave, and invert).
- **selLink**, which specifies whether the currently selected text is a link that will be drawn as a URL. If the user clicks this link, the existing *JadeRichText* control `linkClicked` event method is called, passing the text of the link.
For details (including code examples and new Underline_Type_ constants defined in the JadeRichText class), see the JadeRichText control in the JADE Encyclopaedia of Classes (Volume 3).

Loading Report Writer Files Clarification (PAR 62873)

In earlier releases, the documentation did not state that as all JADE Report Writer files (that is, files with a suffix of rrv, rwo, rwf, rwu, rwr, and rwa) are loaded only into the current schema version, they should therefore be loaded after any database reorganization transition. (See also "Upgrading to JADE 7.1 from an Earlier Release").

Notification Data Limits Clarification (PAR 62863)

The documentation in earlier releases did not state that the limit of 48K bytes applied to notifications containing binary or string (Binary, String, StringUtf8) data only when data is sent across the network.

For applications running within the server node, the limit for notifications containing binary or string (Binary, String, StringUtf8) data is 2G bytes. Note, however, that this applies only to single user and server applications. In multiuser applications, persistent notifications are sent via the database server, even if the receiving process is on the same node as the sender.

In notification cause events, exception 1267 (Notification info object too big) is raised if the binary or string userInfo data exceeds the applicable value.

REST-based Web Services (PAR 62784)

In earlier 7.1 releases, if the JadeRestService class processRequest method was called by user logic and the call was not part of the JADE REST service web message handling framework, a JADE crash could have resulted; for example, when creating a transient instance of a JadeRestService subclass and calling the processRequest method. (This occurred because the JADE REST service had not been initialized and the logic assumed that it had.)

To enable manual testing of JadeRestService methods, the handling has been changed to allow such a call, as follows.

- The REST service will be initialized if it has not been already, unless the application is already attached to a JADE Web Service Manager.

  Exception 11126 (A Rest Service method was called but the service was never initialised) will result.

- If the JadeRestService class reply method is called and its processing is not associated with a received web message, exception 11127 (JadeRestService.reply was called but there is no web message to reply to) is raised.

  Re-implement the reply method on the JadeRestService subclass that is being used. When using manual testing processing, to avoid raising exception 11127, the reply method should not call inheritMethod.

Schema and Class Deletion (NFS 62273)

Documentation in earlier releases did not state that instances of JadeDynamicObject and DynaDictionary are not deleted if classes referenced by JadeDynamicObject or DynaDictionary instances are deleted.

In a JadeDynamicObject, if the type of a property that has been assigned a value is removed by deleting the class or removing the schema, the value is no longer valid and attempting to use it will raise exception 1046 (Invalid class number).

If the membership type of a DynaDictionary is removed by deleting the class or removing the schema, any dynamic dictionaries that have been populated with that membership class are no longer valid and attempting to use it will raise exception 1046.
String::asUuid Method (NFS 62803)

The String primitive type now provides the asUuid method, which has the following signature.

\[
asUuid(): \text{Binary};
\]

The asUuid method returns a binary by formatting the string as a Universally Unique Identifier (UUID).

If the string is not formatted as a valid UUID representation (that is, as returned by the Binary primitive type uuidAsString method), exception 1407 (Invalid argument passed to method) is raised.

The code fragment in the following example shows the use of the asUuid method.

\[
\begin{aligned}
\text{vars} & \quad \text{str} : \text{String}; \\
& \quad \text{bin} : \text{Binary}; \\
\text{begin} & \quad \text{str} := "4dfc912a-b466-01d0-1027-000085823b00"; \\
& \quad \text{bin} := \text{str.asUuid}();
\end{aligned}
\]

Swapping F5 and F9 Accelerator Keys (NFS 63079)

Visual Studio uses the F9 key to set breakpoints, by default, and uses the F5 key to run and continue debug execution. JADE uses the F5 key to set breakpoints and the F9 key to run and continue debug execution. Switching between the two development environments can lead to confusion and frustration when using the F5 and F9 keys.

The Editor Key Bindings sheet on the Preferences dialog now provides the Swap F5 (Toggle Breakpoint) and F9 (Execute/Continue) accelerators check box. As this check box is unchecked by default (that is, false), the behavior in earlier JADE releases is unchanged; that is, the F5 accelerator key causes break points to be toggled and the F9 accelerator key executes or debugs a script method and continues execution of the JADE debugger.

If you check this check box, the menu caption and accelerator key for the Jade debugger Breakpoints menu Toggle command is changed to be triggered by the F9 key. In addition, the JADE development environment Jade menu Execute, Debug, Unit Test, and Unit Test Debug commands and the JADE debugger Debug menu Continue and Animate commands will be changed to use the F5 key.

Notes The setting of the Swap F5 (Toggle Breakpoint) and F9 (Execute/Continue) accelerators check box is saved when you extract your user preferences to a file and they are restored when you reload your preferences.

Changing the setting of the check box takes effect in the JADE development environment as soon as you close the Preferences dialog. It will not take effect within any currently active JADE debug session, and it is read only when you initiate a new debug session.

Synchronized Database Service (PAR 62798, 57217)

In earlier releases, the JADE Initialization File Reference did not state that the [SyncDbService] section could contain the following parameters. In addition, it was not documented that the maximum value of the JournalXferBlocksize parameter in the [SyncDbService] section has increased from 512K to 1G.

JournalReadBuffers

<table>
<thead>
<tr>
<th>Value Type</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>4</td>
</tr>
<tr>
<td>Purpose</td>
<td>The JournalReadBuffers parameter specifies the number of buffers to use when reading a journal file on disk. The minimum value for this parameter is 2 and the maximum value is 100.</td>
</tr>
</tbody>
</table>
Parameter is read when ...
The SDS service is next initialized.

Applicable to database role...
Primary (applies to journal transfer).
Secondary (applies to journal replay).

**JournalReplayBlockSize**

- **Value Type**: Integer prefix multiplier
- **Default**: 128K

**Purpose**
The `JournalReplayBlockSize` parameter specifies the size in bytes of each read buffer used when replaying a journal file.

The minimum value for this parameter is 4K and the maximum value is 1G.

Parameter is read when ...
The SDS secondary service is next initialized.

Applicable to database role...
Secondary.

**Web Service Exposure Error Message (PAR 63094)**

In earlier releases, if a Web service encountered a property whose instance was a subclass of the defined class type and that subclass was not included in the exposure, an assertion failure occurred.

This has been changed so that a new exception (11058) is now raised. If this exception is raised, update the exposure to include the subclass.

**Web Session Timeout (PAR 63011)**

In earlier releases, if a user schema re-implemented the `timerEvent` method on a `WebSession` class being used by a Web Service Provider application, JADE did not start the `WebSession` time-out timer. This caused Web sessions not to time out.

This behavior has been changed so that if the Web session `timeout` value is set (in the `ReadTimeout` parameter of the [WebOptions] section of the JADE initialization file, by XML configuration, or in your application logic), when a Web session is created, the time-out timer will be started regardless of whether or not the Web Service Provider application in your user schema has re-implemented the `timerEvent` method on the `WebSession` subclass being used.

**Note**
To enable the session time out process to function correctly, it is then the responsibility of that re-implemented method to call the `inheritMethod` instruction.
Changes in JADE Release 7.1.05

This section contains details about product and documentation changes in JADE release 7.1.05.

For details about release 7.1.03 (the first general release of JADE 7.1) and changes in release 7.1.04, see "JADE 7.1 Changes that May Affect Your Existing Systems", "Changes and New Features in JADE Release 7.1.03", and "Changes in JADE Release 7.1.04", elsewhere in this document.

Background Process Lock Timeout (PAR 62567)

You can now specify a default lock timeout for the background process, in the BackgroundProcessServerTimeout parameter in the [JadeClient] and [JadeServer] sections of the JADE initialization file.

The background process lock timeout is specified in milliseconds, with the default value of 30000 (that is, 30 seconds).

**Caution** You should not change this value unless the background process is having locking problems. Before you increase this value, examine the application to determine which objects are being locked and whether locks are being held for too long. For example, new nodes cannot sign on if the system.nodes dictionary is locked by the application. It is better to change the application to minimize the locking of system collections.

Batch Schema Load Utility reportReplaceView Parameter (PAR 62375)

In earlier releases, the JADE Schema Load Utility User's Guide did not state that you can specify the optional reportReplaceView parameter in the batch Schema Load utility (jadloadb) with both the reportViewFile and reportLoadAllFile parameters, as they both contain views.

If you specify the reportReplaceView parameter with one of the other report-related parameters (for example, the reportFoldersFile parameter), it is ignored as they do not contain view files.

Cache Size Limit Default Values (PAR 62677)

Cache size limit default values increased in JADE 7.0.05 but were not updated in the product information prior to the JADE 7.0.12 release and this JADE 7.1.05 release.

In the JADE initialization file, the default value of the:

- ObjectCacheSizeLimit parameter in the [JadeClient] and [JadeServer] sections has increased from 8M to 80M
- TransientCacheSizeLimit parameter in the [JadeClient] and [JadeServer] sections has increased from 5M to 40M
- RemoteTransientCacheSizeLimit parameter in the [JadeServer] section has increased from 5M to 40M

Dynamic Property Clusters

This section describes the dynamic property cluster changes in this release.

Deleting Dynamic Clusters (PAR 62396)

As a dynamic cluster assigned to a class cannot be deleted if the class has instances, the Delete button on the Dynamic Clusters dialog is disabled if the owning class has instances, because the instances may still have data associated with deleted properties that were owned by the cluster.
Dynamic Property Cluster Names (PAR 62436)

To clarify documented dynamic property cluster naming conventions, runtime and design-time dynamic property cluster names must:

- Begin with a lowercase or an uppercase alphabetic character.
- Be unique within the hierarchy of the class in which they are defined; that is, they cannot be used in a superclass or a subclass of the class, including in any superschemas or subschemas.

Extracting and Loading Dynamic Property Clusters (PAR 62434)

In earlier releases, if a dynamic property cluster was assigned to a class but it contained no properties, the cluster was not extracted in the schema file and was lost if the schema was loaded into another database.

Defined dynamic property clusters are now included in the attributes of a class's definition in the schema, and are loaded regardless of whether the cluster has defined properties, which enables you to share code via schema extract and load actions.

JadeHTMLClass::buildFormActionOnly Method (PAR 62632)

Because the JadeHTMLClass class buildFormAction method in earlier releases could not generate correct HTML in all circumstances, the method has now been replaced with the JadeHTMLClass class buildFormActionOnly method, which has the following signature.

```java
buildFormActionOnly(): String;
```

If you call the new buildFormActionOnly method, you must change your HTML or JADE code to generate the form tag method attribute wherever you were using the superseded buildFormAction method with a non-null value in the meth parameter (that is, a value of "get" or "post").

The return value of buildFormActionOnly method is the same as the return value of the superseded buildFormAction method when it was called with a parameter value of null.

JadeRecompileAllMethods Application (PAR 62539)

In earlier releases, the JadeRecompileAllMethods non-GUI client application in the jadclient executable recompiled only uncompiled methods or methods in error.

To force a recompilation of all successfully compiled methods, the JadeRecompileAllMethods application now has the optional includeAlreadyCompiled command line argument, which has a Boolean (true|false) value, as shown in the following example.

```java
jadclient path=c:\jade\system ini=c:\jade\system\testjade.ini schema=RootSchema app=JadeRecompileAllMethods endJade includeAlreadyCompiled=true
```

As the includeAlreadyCompiled argument defaults to false, specify includeAlreadyCompiled=true on the command line to recompile all previously (successfully) recompiled methods.

MemoryAddress Value Adjustment (PAR 62517)

The MemoryAddress primitive type now provides the adjust method, which has the following signature.

```java
adjust(offset: Integer64): MemoryAddress;
```

This method adjusts the value of the MemoryAddress of the receiver by an integral amount (for example, when stepping through blocks of memory) and returns a new MemoryAddress value.

Microsoft SQL Server 2014 Support (PAR 62731)

The RPS relational target database now also supports Microsoft SQL Server 2014.
Microsoft Visual C++ Redistributable Package (PAR 62506)

Any system that is to run JADE 7.1 must have Microsoft Visual C++ 2013 Redistributable Package installed. JADE 7.1 32-bit presentation clients require Visual C++ 2013 Redistributable Package (x86), version 12.0.21005.1 (that is, not version 12.0.30501 documented in earlier 7.1 releases).

By default, a 32-bit presentation client will be upgraded to a version requiring the Microsoft Windows Visual Studio 2013 C++ runtime binaries.

Thread Parameters in the [JadeServer] Section (PAR 62498)

In earlier releases, when any of the new MinShortThreads, MaxShortThreads, MinLongThreads, or MaxLongThreads parameters were not specified in the [JadeServer] section of the JADE initialization file, server initialization wrote them out with the corresponding MinServerThreads or MaxServerThreads parameter value carried over, or with the default MinServerThreads or MaxServerThreads parameter value.

From this release, when the MinServerThreads and MaxServerThreads parameters are not specified to seed the MinShortThreads, MaxShortThreads, MinLongThreads, and MaxLongThreads parameter specifications, the MinShortThreads, MaxShortThreads, MinLongThreads, and MaxLongThreads parameters now use their own default values, which are as follows.

- MinShortThreads default value is 20
- MaxShortThreads default value is 100
- MinLongThreads default value is 15
- MaxLongThreads default value is 100

If the value of a MinServerThreads or MaxServerThreads parameter is used to seed any of the MinShortThreads, MaxShortThreads, MinLongThreads, and MaxLongThreads parameters during initialization, the MinServerThreads and MaxServerThreads parameters are now removed from the [JadeServer] section of the JADE initialization file after initialization, as they are superseded by the new parameters.

When the MinServerThreads and MaxServerThreads parameters are not specified in the [JadeServer] section and the new MinShortThreads, MaxShortThreads, MinLongThreads, and MaxLongThreads parameters do not exist, the new parameters are written to the [JadeServer] section with the value <default>. 
Changes in JADE Release 7.1.04

This section contains details about product and documentation changes in JADE release 7.1.04. For details about release 7.1.03 (the first general release of JADE 7.1), see "JADE 7.1 Changes That May Affect Your Existing Systems" and "Changes and New Features in JADE Release 7.1.03", elsewhere in this document.

Converting User Database Files (PAR 62300)

When converting a JADE user database (for example, upgrading from JADE 6.3 or 7.0 to 7.1 or converting from one operating system or hardware platform to another), the limit of 1,000 files that the JadeConvertDb application in the jadclient program can handle has been removed.

In earlier releases, systems containing more than 1,000 files could be upgraded or converted only by specifying each file individually on the command line.

Database Roll-Forward Recovery (PAR 60499)

A large number of audit records can be written in a second, or even a thousandth of a second. Roll-forward recovery now provides a finer, more-specific termination condition capability to support roll-forward recovery terminating at a specified audit serial number. When rolling forward to an audit serial number, audit records are processed until the audit record serial number is greater than the termination serial number.

The Roll-Forward Recovery Options group box on the JADE Database utility Initiate Recovery and Restore Database And Roll Forward dialogs now provides the Recover transactions to audit serial number option button and a text box in which to specify the serial number of the last audit record to which committed transactions are recovered.

The batch JADE Database utility provides the optional lastSerialNumber parameter that you can specify for the recover and restore commands instead of the optional recoverTo or lastJournal parameter, to specify the last audit record to which committed transactions are recovered.

The following exceptions can be raised.

- 3193 (Roll-forward terminate serial number is earlier than audit trail), when a roll-forward recovery to a serial number is attempted with termination criteria earlier than the first audit trail record. The audit trail serial number and the termination serial number criteria are written to the recovery.log file.
  
  Re-issue the roll forward request with the corrected termination serial number criteria.

- 3194 (Roll-forward terminate before backup recovery complete), when a roll-forward recovery of an online backup is performed specifying a termination condition that terminates the roll forward before the end backup checkpoint record is processed; that is, before file integrity is restored. The audit trail serial number and Log Sequence Number (LSN) and the End-Backup LSN are written to the recovery.log file.
  
  Re-issue the roll forward request with termination criteria beyond the end backup checkpoint record.

DataPumpApplication Parameter (PAR 62311)

If the value of the DataPumpApplication parameter in the [JadeRps] section of the JADE initialization file is set to <default>, the JadeRpsDataPump application is run from the schema in which the RPS mapping is defined (that is, this value is not equivalent to specifying the supplied RootSchema JadeRpsDataPump application, which is the supplied RPS Datapump application).

Exception::errorItem Property Length (PAR 62349)

The length of the Exception class errorItem property (which is displayed in the default exception dialog as error item) has increased to 540 characters from 60 characters.
This change allows for the JADE 7.1 increased property length to 100 characters, as the `errorItem` property is set by the module that raised the exception and it usually contains additional information about the exception (for example, an invalid property exception would contain the name of the property in error).

**JadeHttp Logging of Message Meta Data (PAR 62345)**

To suppress logging completely, set the value of the `Trace` parameter in the [Jadehttp Logging] section of the `jadehttp.ini` file to `false`.

When the value of this parameter is set to `true`, logging does occur but it does not log user data, only details of the message meta data; that is, it logs messages acknowledging only that a message has been received or sent and it does *not* include any of the text sent or received from the client, as this text could contain personal information, passwords, credit card details, and so on.

**Note** You cannot use this parameter to inspect and debug data passing through the `jadehttp` library.
Changes and New Features in JADE Release 7.1.03

This section summarizes the product and documentation changes and new features in JADE release 7.1.03. For details about the changes in release 7.1 that may affect your existing systems, see "JADE 7.1 Changes that May Affect Your Existing Systems", earlier in this document.

.NET

This section describes the .NET changes in this release.

.NET Class Library (PAR 60740)

Under certain circumstances in earlier releases, JADE notifications did not behave as expected when app.doWindowEvents was indirectly called from a .NET process.

Note The following applies only to .NET applications that use JADE methods exposed via the C# Exposure wizard in the JADE development environment.

If a .NET application calls exposed JADE methods and the JADE method contains a beginNotification or beginClassNotification instruction, the notification will be delivered to the thread that performed the .NET "new JoobContext()".

A beginNotification or beginClassNotification instruction in a JADE method is regarded as subscribing to a different notification than one done via RegisterNotificationHandler or UnregisterClassNotificationHandler in .NET, even if all of the parameters are equivalent. An event that is explicitly (or implicitly) caused will be delivered to all subscribers, even if that language that caused the notification did not subscribe to the notification.

When a JoobContext is Disposed, all subscriptions are cancelled.

As the JADE code is running within .NET, the JADE process will not have an idle state, so the Application class doWindowEvents method must be called. When doing so, review all of the documented cautions in Volume 1 of the JADE Encyclopaedia of Classes regarding the use of this method.

C# Exposure Wizard

This section describes the C# Exposure Wizard changes in this release.

Exposure Entity Maximum Lengths (NFS 61179)

The JADE C# exposure wizard enables you to specify an exposure name with a maximum of 100 characters. (In earlier releases, the exposure name could not exceed 30 characters.)

ExternalCollection Subclasses (PAR 60285)

Uses of the ExternalArray, ExternalDictionary, and ExternalSet subclasses of the ExternalCollection class are not supported by the JADE C# exposure, and are excluded from the generated C#. In earlier releases, uses of these classes were generated and the C# compile failed.

The C# Exposure Wizard now ignores uses of these classes in the list of features that can be exposed (for example, a property reference).

Note You must manually alter any existing C# exposures that include such features, to exclude the exposed feature.
Saving Definition and Generate Values (PAR 59367)

The C# Exposure Wizard now saves and restores the following values.

- Name space
- The Current schema only or Superschemas up to value
- The name of the Top Most Schema if the Superschemas up to option button is selected
- The last output directory used to generate the exposure
- The Sign-on Schema Name value
- The Sign-on Application Name value
- The value of the Sign-on to Jade as Multiuser check box

Custom Control Tab Order (PAR 61929)

In releases earlier than JADE 7.0.11, tabbing into and within .NET controls in JADE that use Microsoft control subclasses was not handled.

Tab handling for .NET controls in JADE that use Microsoft control subclasses has now been implemented so that tabbing into the control is seamlessly integrated with other JADE controls on the form. If the .NET control has more than one subcontrol that can have focus, tabbing now moves through those subcontrols in .NET TabIndex order. When the TAB key is pressed on the last such subcontrol, focus moves to the next JADE control in the JADE tabIndex order. (SHIFT+TAB performs the same function in reverse tabIndex order.)

**JADE .NET Developer’s Reference**

The JADE .NET Developer’s Reference has been enhanced to include the following information. (Chapters 1, 2, and 3 from earlier releases have been renumbered Chapters 5, 6, and 7, respectively.)

<table>
<thead>
<tr>
<th>Chapter 1</th>
<th>.NET and JADE requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 2</td>
<td>JADE .NET object management</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Sample client-server application</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>Introduction to JADE .NET tutorial</td>
</tr>
<tr>
<td>Appendix A</td>
<td>Mapping JADE primitive types to Common Language Runtime (CLR) data types</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Developing applications to store, edit, and query spatial information directly through the JADE .NET API</td>
</tr>
</tbody>
</table>

**Ad Hoc Indexes**

Ad hoc indexes, which enable you to create indexes suitable for optimizing ODBC ad hoc queries without requiring database reorganization, are defined as a subclass of DynaDictionary, with instances mapped to a RootSchema map file called _indexes.dat. The _indexes.dat database file is partitioned (for example, _indexes_part0000000001.dat), with each ad hoc index instance created in a new partition to facilitate a fast drop index operation, which uses the existing dropPartition operation. The classes that represent index meta data are mapped to the _indexdefs.dat RootSchema database file. (The _indexes.dat and _indexdefs.dat files are of Kind_Utility.)

The Ad Hoc Index Controller application reads from the main database (or from journals when catch-up is required), and creates and maintains the user-defined index. The ODBC driver function that performs ODBC queries can then use a combination of ad hoc indexes and collections in the main database to satisfy queries.
The JADE development environment Ad Hoc Index Browser enables you to create or modify ad hoc index definitions, and to drop unwanted indexes. As the ad hoc index definition does not define any meta data, the data is not included when the schema is extracted. The Ad Hoc Index Browser enables you to save an ad hoc index to an Extensible Markup Language (XML) file and to load a saved index. For details, see "Maintaining Ad Hoc Indexes", in Chapter 9 of the JADE Development Environment User's Guide. For details about using the jadclient executable to create indexes suitable for optimizing ad hoc queries without requiring database reorganization, see "Ad Hoc Index Batch Interface", in Chapter 1 of the JADE Runtime Application Guide.

**Adding User Classes at Run Time**

Your user applications can add user classes at run time. While these user-defined classes are visible in the JADE development environment, they are not considered as part of the JADE model, you cannot reference them directly in JADE methods, they are not extracted, reorganized, and so on.

You cannot add properties to user classes in the JADE development environment. You can only define runtime dynamic properties on user classes. Instances of any lifetime can be created (that is, persistent, transient, and shared-transient lifetimes).

The following table summarizes the classes and methods that enable you to maintain user classes.

<table>
<thead>
<tr>
<th>Class</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>addUserCollectionSubclass</td>
<td>Creates a user collection class as a subclass of the specified superclass in the receiving schema</td>
</tr>
<tr>
<td>Schema</td>
<td>addUserSubclass</td>
<td>Creates a user class as a subclass of the specified superclass in the receiving schema</td>
</tr>
<tr>
<td>Schema</td>
<td>deleteUserSubclass</td>
<td>Deletes a user class from the specified superclass in the receiving schema</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>addExternalKey</td>
<td>Adds an external key definition to a user class at run time</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>addMemberKey</td>
<td>Adds a member key definition to a user class at run time</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>clearKeys</td>
<td>Clears existing key definitions</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>endKeys</td>
<td>Indicates the end of a single or multiple key definition</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>setLength</td>
<td>Sets or changes the element length for an array</td>
</tr>
<tr>
<td>JadeUserCollClass</td>
<td>setMembership</td>
<td>Sets or changes the membership of a user class at run time</td>
</tr>
</tbody>
</table>

For details, see volumes 1 and 2 of the JADE Encyclopaedia of Classes. See also "Adding User Classes at Run Time", in Chapter 21 of the JADE Developer's Reference.

**AutoComplete Functionality**

This section describes the AutoComplete changes in this release.

**AutoComplete Selection from a List (PAR 61330)**

When you used the TAB key to select an item from the displayed AutoComplete list box in earlier releases, the tab character was appended to your selected option when it was inserted in the editor pane.

When you select an item in the list by using the TAB key, the tab character is no longer appended to the selected entry when it is inserted into the editor pane.

**Pseudo Type Variable Support (PAR 61634)**

The JADE editor AutoComplete feature has been enhanced as follows.
The JADE pseudo types of `InstanceType`, `MemberType`, and `SelfType` are now handled, and offer AutoComplete suggestions as required.

**Note**  
`KeyType` parameters are not handled.

The displayed method signature handles a `ParamListType` pseudo type parameter when highlighting the current parameter in the displayed method signature.

Note, however, that if the first parameter of the method being called is a `ParamListType`, if the terminating bracket of the call has not yet been entered, it is not possible to accurately determine which parameter is currently being entered when there is more than one parameter.

**Application Class `isValidObject` Method (PAR 52806)**

The performance of the `Application` class `isValidObject` method has been improved. This method now takes into account the presence of the object in the cache if automatic cache coherency is enabled.

**Application Programming Interface (PAR 60679)**

The JADE C Application Programming Interface (API) has been extended, as follows. (For details, see Chapter 3 and Chapter 6, respectively, of the JADE Object Manager Guide.)

The JADE API `jomcalls.h` file now provides the following Application Programming Interface (API) call.

- `jomGetNodeContextHandle`

The existing header file `joscalls.h` in the `include` directory on the JADE release medium now contains the following new functions.

- `josDskDateToGregorian`
- `josDskTimeStampToGregorianHMSm`
- `josDskTimeStampCompare`
- `jomDecimalFromReal`
- `jomDecimalFromInt64Scale`
- `jomDecimalFromString`
- `jomDecimalToReal`
- `jomDecimalToString`
- `jomDecimalCompare`

The `include` directory on the JADE release medium now contains the new header file `jombuild.h`, which provides defines that enable you to perform build and runtime version checking. Each JADE Dynamic Link Library (DLL) exports a function named using the define `MAKEMODULEVERSIONFUNC`; for example, the function name exported by `jomutil.dll` in the 7.1.03 version of JADE is "moduleVersion_jomutil_7_1_03".

Static importers of the DLL can use the following to ensure that the importing DLL or executable, the `jomutil.lib`, and the `jomutil.dll` are consistent.

**Declaration:**

```c
extern "C" DllImport int JOMAPI MAKEMODULEVERSIONFUNC(jomutil)();
```

**Invocation:**

```c
int jomutilpatch = MAKEMODULEVERSIONFUNC(jomutil)();
```
If the `jombuild.h` version included by the user sources does not match the `jomutil.lib`, the compile-time linker raises an error; for example:

```
mysource.obj : error LNK2019: unresolved external symbol __imp_moduleVersion_jomutil_7_0_7 referenced in function MyVersionChecker
```

At run time, if the importer version and the `jomutil.lib` version do not match `jomutil.dll`, the dynamic linker raises the following.

The procedure entry point moduleversion_jomutil_7_0_8 could not be located in the dynamic library `jomutil.dll`.

### Class Number Extensions

JADE has extended the number of user classes permitted to just under one million. The:

- First user class number is 2,048.
- Maximum user class number is 999,999. (System classes have class numbers starting at 1,000,000.)
- Number of user classes is 997,952.
- Maximum number of exclusive collections (subobjects), binary large objects (blobs), and string large objects (slobs) in a class is 65,535.

### Class Persistent Instances Methods (PAR 61167)

The `Class` class now provides the `countPersistentInstances64` and `countPersistentInstancesLim64` methods to return the number of non-exclusive 64-bit persistent instances in the receiver class. These methods have the following signatures.

```
countPersistentInstances64(): Integer64;
countPersistentInstancesLim64(limit: Integer64): Integer64;
```

### create Statement Handling (PAR 62019)

When finding undeleted transient instances (that is, possible transient leaks), JADE now handles `create` statements in which the `as` expression is a static class name. If the `as` expression is not a static class name, the `create` statement is ignored because the type returned by the expression cannot be determined.

The type of the `create` statement is treated as transient if the `create`:

- Qualifier is `transient`
- Is not qualified and the class to be created is defined as defaulting to transient

### Disk Write-Cache Messages (PAR 60863)

Prior to JADE release 7.0.09, no warning was issued during database file or journal file opening if device disk write-caching was enabled and Windows drive cache flush was disabled.

As each new volume is encountered when opening a writable file, the database now logs the device write-cache settings. If the device reports that write-cache is enabled and the device write-cache type is not write-through, warning messages are logged if non-volatile write-cache is not enabled on the device and the device does not support flush-cache operations or you have configured the device as power protected (suppressing flush-cache operations). Under these circumstances or if the device settings cannot be determined, a power outage could result in irrecoverable data loss or corruption.
DynaDictionary Methods (PAR 58349)

The DynaDictionary class can now contain the following methods, which enable you to specify in the sortOrder parameter the locale identifier of the locale used for comparison. A value of zero (0) indicates the Binary sort order.

- The `addExternalKeyWithSortOrder` method, which adds an external key specification to a dynamic dictionary, has the following signature.

  ```java
  addExternalKeyWithSortOrder(keyType: Type;
  keyLength: Integer;
  descending: Boolean;
  caseInsensitive: Boolean;
  sortOrder: Integer);
  ```

- The `addMemberKeyWithSortOrder` method, which adds a member key specification to a dynamic dictionary, has the following signature.

  ```java
  addMemberKeyWithSortOrder(propertyName: String;
  descending: Boolean;
  caseInsensitive: Boolean;
  sortOrder: Integer);
  ```

For details, see Volume 1 (that is, EncycloSys1.pdf) of the JADE Encyclopaedia of Classes.

Dynamic Clusters and Properties

There are now two types of dynamic properties: runtime and design-time.

- Runtime dynamic properties are essentially unchanged from JADE 7.0, with the only difference being that the color of property text (magenta) in the hierarchy browser. Runtime dynamic properties cannot be maintained (created, changed, or deleted) in the JADE development environment.

  For details, see "Runtime Dynamic Properties", in Chapter 21 of the JADE Developer's Reference.

- Design-time dynamic properties are new in JADE 7.1. Design-time dynamic properties are the same as static properties, except that they:
  - Are stored in a binary large object (blob) associated with the owner dynamic cluster.
  - Generally do not require a reorganization to add, change, or delete them.
  - Are slightly slower to access (similar to static blob access).
  - Can be maintained only from the JADE development environment or the Schema Load utility.
  - Are displayed in the development environment hierarchy browser with a default color of maroon.

  For details, see "Dynamic Clusters and Properties", in Chapter 4 of the JADE Development Environment User's Guide.

The JADE development environment supports dynamic properties as follows.

- Dynamic property display

  The Window sheet of the Preferences or JADE Installation Preferences dialog now enables you to change the color with which the text of dynamic design-time and runtime property names are displayed in the Properties List of a hierarchy browser. By default, design-time dynamic properties are displayed in maroon and runtime dynamic properties in magenta.

- Dynamic property cluster maintenance
The Classes menu now provides the **Dynamic Property Clusters** command, which accesses the Dynamic Clusters dialog that enables you to view the clusters defined for the class, add a new cluster, delete an existing cluster, and view a list of the properties defined in a cluster. (This command is enabled only when the selected class is allowed dynamic clusters.)

- Design-time dynamic property maintenance

The Properties menu now provides the **Add Dynamic Attribute** and **Add Dynamic Reference** commands.

**Entity Name Maximum Length (NFS 61179)**

The 30-character limit for schema entity names has increased to the maximum of 100 characters, to ensure, for example, that class, method, and property names integrated with .NET are unique. (See also "*String Arrays and Dictionary Keys*" under "Jade 7.1 Changes that May Affect Your Existing Schemas").

The names of the following entities now have a maximum length of 100 characters.

- ActiveX component
- Ad hoc index
- Application
- C# exposure
- Class
- Constant (global, class, or local)
- Dynamic property
- Dynamic property cluster
- Form
- HTML document
- Identifier in a method (that is, a property or method of the receiver, a parameter, a local variable, a constant, or a class or primitive type name)
- Imported .NET object
- Interface
- **JadeDynamicObject** property
- Map file
- Mask queue in the messaging framework
- Menu item
- Method
- Method view
- Non-GUI client application service
- Package (imported and exported)
- Primitive type
- Property
- Relational Population Service (RPS) mapping
- Schema
- Server node
- Translatable string
- Web service application
- Web service exposure definition
- XAML document

In addition, the:
- **SystemLimits** category now provides the `Max_Identifier_Length` global constant, which has an **Integer** value of 100.
- Control name maximum length is now 86 characters.
- **Exception** class `errorItem` property value now has a maximum of 540 characters.

**External Function and External Method Entry Point Length (PAR 61412)**

External function and external method entry points can now be up to 255 characters in length. If they are longer than 100 characters, they must be enclosed in single quote (') or double quote (""") characters or exception 6007 (Token too long) is raised.

The schema extract process encloses all entry point names in quote characters.

**Extract Sort Exception (PAR 61403)**

In earlier releases, if the input file passed to the File class `extractSort` method contained a null character, because of the way embedded Binary and String properties were handled, a 1415 internal exception was raised and a process dump was generated.

The sort code now recognizes the null character and raises a new exception 5320 (Unable to process all input in the file).

**Generated JADE .NET Names (NFS 60673)**

The JADE .NET Import Wizard now uses a different mechanism to ensure that the generated JADE names are unique, resulting in more-workable JADE names. The results of this change are as follows.

- Fewer names need to be qualified
- The added qualifiers will mostly be simple one-level low-value numeric values (for example, _1, _2, and so on)
- Multiple-level qualifiers (for example, _2_1) will not often be needed

**Note** This change is applied only to new .NET assembly imports. Existing imports are unaffected, and reloading an assembly uses the mechanism under which the JADE names were generated so that existing interfaces are not affected.

The style of the generated names is saved in the database and in the forms definition (.ddb) file when extracted.

**Interface Stub Methods (NFS 60359)**

When stub methods are created for an interface implemented on a class, the included text description now includes the name of the interface for which they were created; that is:

```
this method stub has been automatically generated by Jade to satisfy interface
implementation requirements for the interface-name interface
```
JADE Debugger

This section describes the JADE debugger changes in this release.

Attaching the JADE Debugger to a Running Application (NFS 2413)

JADE now enables you to dynamically attach the JADE debugger to a running application, provided that the JADE development environment is also being run by that client.

The JADE User Interrupt menu now provides the Attach Debugger command for each application, which enables you to initiate the JADE debugger. The application execution halts in the debugger at the next debugged method logic statement.

By default, only methods that begin execution after the JADE debugger is attached can be debugged, because the JADE Interpreter executes methods using two different execution engines: normal execution and debugger execution. Normal execution is significantly faster and cannot be debugged. As a result, any methods already in execution will not halt on breakpoints or allow debug break and step. The exceptions to this are:

- Running the application in Run in Debug Ready mode. The JADE Run Application dialog now contains the Run in Debug Ready Mode check box, which runs the application in debug mode but without the debugger being initiated when this is checked. Attaching to that application then allows debugging of any method already on the execution stack. (Note that if the application is idle and no modal dialogs are displayed, attaching the debugger is equivalent to running in Debug Ready mode.)

- If the debugger was attached to an application, the debugger is closed and then subsequently re-attached.

If the application is in a loop, attaching the debugger breaks only if the application is debug-ready or if the logic begins execution of another user-defined method after the attach action.

If you use the JADE User Interrupt menu Break Application command and the logic being executed is debug-ready, it causes a debugger break if logic is being executed.

dbgApplicationWithParameter Method (NFS 61035)

The Application class can now contain the debugApplicationWithParameter method, which has the following signature.

```java
    debugApplicationWithParameter(schemaName: String;
                           applicationName: String;
                           passedObject: Object);
```

Executing this method initiates the application specified in the applicationName parameter requested application in JADE debug mode, passing the value specified in the passedObject parameter to the initialize method defined in the application. The initialize method must expect an object parameter; otherwise an exception is raised.

The JADE debugger stops on the first logic statement executed in the user application.

The conditions that apply to the Application class debugApplication method apply to the method; that is:

- To debug the application, the JADE development environment must be running.
- You must have a defined user profile in the JADE development environment.
- The schema specified in the schemaName parameter must exist.
- The application must be defined in that schema or a superschema.

Displaying the Current Exception Handler Stack (NFS 46856)

The View menu of the JADE debugger now contains the Show Exception Handler Stack command, which displays the exception handlers armed by the receiving process in the current node.

The exception handler stack displays:
The name of the application that is arming the exceptions.

For each armed exception handler, the following information is displayed.
- The stack depth (the handlers at the top of the stack are invoked first)
- Whether the exception handler is a global or a local exception handler
- The name of the class and method that armed the exception (this applies to local exception handlers only)
- The name of the exception class
- The name of the class and method of the exception handler method
- The name of the class of the exception handler receiver

JADE Initialization File

This section describes the JADE initialization file changes in this release.

AcceptZeroEnvironmentUUID Parameter

The [JadeServer] section of the JADE initialization file can now contain the AcceptZeroEnvironmentUUID parameter, which is set to false by default, indicating that the environment identity specified in the server command line parameter of a client shortcut must match the identity of the database server, as shown in the following example.

```
server=TcpIp://localhost:6005/48cf13df-bf6d-df11-87e2-2e5925024153
```

For more details, see "Format of the Server URI String", in "Format of the Server URI String", in Chapter 3 in the JADE Installation and Configuration Guide.

Set the AcceptZeroEnvironmentUUID parameter to true if zeroes can be specified for the environment identity in the server command line parameter of a client shortcut, as shown in the following example.

```
server=TcpIp://localhost:6005/00000000-0000-0000-0000-000000000000
```

BackupCompressedFileGrowthIncrement Parameter (PAR 60653)

The [PersistentDb] section of the JADE initialization file can now contain the BackupCompressedFileGrowthIncrement parameter, which has a default value of 1G. This parameter specifies the size of the increment by which a backup file increases during a compressed backup.

If the value supplied is not a multiple of the value of the BackupBlockSize parameter, it is rounded up to the next highest multiple.

The minimum value is 64M bytes. If the file to be backed up is smaller than the value of the BackupCompressedFileGrowthIncrement parameter, the backup file is pre-allocated to that size. The maximum value is 64G bytes.

DiskCacheWriteOnlyDbSegments Parameter (PAR 60707)

The [PersistentDb] section of the JADE initialization file can now contain the DiskCacheWriteOnlyDbSegments parameter, which has an integer default value of 16, unless limited by available physical memory. This parameter specifies the minimum and maximum number of segments that the disk cache is to use when the database mode is write-only; for example, during upgrades.

For the minimum and maximum values, refer to the DiskCacheMinSegments and DiskCacheMaxSegments parameters, respectively, in the [PersistentDb] section.
**Note**  The value of this parameter can be constrained by the actual maximum number of segments calculated during initialization.

The parameter is read when the database server node is initialized; for example, when you restart the database server.

**JADE Ad Hoc Index [JadeAdHocIndex] Section**

The JADE initialization file can now contain the [JadeAdHocIndex] section, which enables you to specify options for the worker applications that build, drop, and delete an ad hoc index, and for the controller application that starts worker applications when there is an ad hoc index maintenance operation to be performed.

For details about the **BuildCommitPeriod** and **MaxBuildWorkers** parameters, see "JADE Ad Hoc Index Section [JadeAdHocIndex]", in the JADE Initialization File Reference.

**JADE Application Server Upgrade Parameters**

The [JadeAppServer] section of the JADE initialization file on each application server now provides the following parameter, which enables you to upgrade 32-bit presentation (thin) client binaries to 64-bit binaries when the client is using a 64-bit operating system.

In addition, you can control the specific architecture of the binaries downloaded to a specific presentation client.

- **UpgradeRuntimeTo64bit**

For details, see "Upgrading 32-bit Thin Client Binaries to 64-bit (NFS 61726)" under "Thin Client Downloads" and "Upgrading a 32-bit Presentation Client Connecting to a 64-bit Application Server", elsewhere in this document.

**JADE Sentinel Configuration**

The **EnableSentinel** parameter in the [FaultHandling] section and the **ExcludeDiskCache** parameter in the [JadeSentinel] section of the JADE initialization file enable you to configure the **jadesentinel.exe** program.

The JADE Sentinel runs as a debugger attached to its parent process, looks for specific exceptions, performs out-of-process process dumps, and terminates the program, as appropriate. For more details, see "Monitoring Your Active Process Using JADE Sentinel", later in this document.

**JADE Server Section Parameter Values (PAR 60842)**

The default, minimum, and maximum values of the [JadeServer] section parameters listed in the following table are:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxLongThreads</td>
<td>100</td>
<td>1</td>
<td>4095</td>
</tr>
<tr>
<td>MaxShortThreads</td>
<td>100</td>
<td>2</td>
<td>4095</td>
</tr>
<tr>
<td>MinLongThreads</td>
<td>15</td>
<td>1</td>
<td>4095</td>
</tr>
<tr>
<td>MinShortThreads</td>
<td>20</td>
<td>2</td>
<td>4095</td>
</tr>
<tr>
<td>TransportIdlePollInterval</td>
<td>120000</td>
<td>5000</td>
<td>Max_Integer (#7FFFFFFF)</td>
</tr>
</tbody>
</table>

**Parameter Value Handling (PAR 61704)**

In earlier releases, the `<default>` value for a JADE initialization file section and parameter pair was case-sensitive.

The value is now case-insensitive (for example, `<default>` and `<DEFAULT>` are both valid).
UpgradeRuntimeTo64bit Parameter (NFS 61726)

The [JadeAppServer] section only of the JADE initialization file on each application server can now contain the UpgradeRuntimeTo64bit parameter, which defaults to false, indicating that the normal thin client upgrade process occurs. The UpgradeRuntimeTo64bit parameter is read when the application server starts up. Set this parameter to true to allow the downloading of 64-bit binaries to clients that are running 32-bit JADE binaries on a 64-bit operating system (WoW64). When this parameter is true and a thin client connects to the application server, if the client is running 32-bit binaries on a 64-bit operating system, a download of the JADE 64-bit binaries will occur instead of 32-bit binaries.

Note For the UpgradeRuntimeTo64bit parameter to take effect, the presentation client must first be upgraded with the version of JADE that supports this parameter. The previous thin client versions do not pass their operating system type in the initial handshake that is used to determine whether a download may be required. As a result, to get clients upgraded to 64-bit requires two downloads: one to update to JADE 7.1, and a second to upgrade to 64-bit.

For more details, see "Upgrading 32-bit Thin Client Binaries to 64-bit (NFS 61726)".

JADE Logical Certifier (PAR 60649, 60505, 60506)

The JADE Logical Certifier now:

- Writes output of meta data certification to _metacert.* files, except for the fix file, which continues to be output as _logcert.fix. (Repair files are unaffected by this change.)
- Appends output to the _logcert.log file and the _repair.log file, rather than overwriting them.
- Uses the JadeLog directory instance as the default value of the Log File Directory text box on the JADE Logical Certifier dialog.

Detecting a Collection Object with Multiple Keys (PAR 61172)

In earlier releases, the JADE Logical Certifier did not detect if a collection contained an object multiple times, where the keys for one entry were valid and the keys for the other entries were not valid; that is, it was still possible to locate the entry using the includes statement.

The JADE Logical Certifier now detects if an object is in a MemberKeyDictionary multiple times, where the keys for one entry were valid and the keys for the other entries were not valid. The existing user data error 3 has been renamed 3A (Object was found via foreach relatedObj in coll but not via coll.includes(relatedObj)) and a new user data error 3B (Object was found in the collection multiple times with valid and invalid keys) is provided. The repair is rebuild coll; that is, if the object has an invalid key path, it is your responsibility to repair the error yourself.

JADE Remote Node Access Utility

This section describes the JADE Remote Node Access utility (jadrap) changes in this release. (See also "JADE Remote Node Access (PAR 60364)", under "JADE 7.1 Changes that May Affect Your Existing Schemas").

Menu Item Changes (PAR 57911)

The Network menu and the Threads command in the Options menu have been removed from the Remote Node Access utility window. (Values are updated by using the appropriate parameters in the [JadeServer] section of the JADE initialization file.)

In addition, the Show Configuration and Show Connections commands have been added to the File menu, to enable you to display server network and thread configuration details and current client connections information, respectively. (You can clear these details, by selecting the Clear Display command in the File menu.)
Termination Message Box

The JADE Remote Node Access utility has an optional new style TERMINATION message box, which is controlled by using the TerminationMsgbox parameter with a value of TryAgain in the [JadeServer] section of the JADE initialization file. If you do not specify this value or the parameter has a <default> value, the existing behavior applies (that is, the message box displays Yes and No buttons only).

When the TerminationMsgbox parameter has a value of TryAgain, the message box displays the following buttons.

- **Cancel**
  Closes the message box.

- **Try Again**
  Rechecks to see if there are any active nodes. If there are no active nodes, jadrap terminates. If there are active nodes, the message box is redisplayed.

- **Continue**
  Shuts down jadrap, even if active nodes are attached.

JADE Version Information Utility

This section describes the JADE Version Information utility changes in this release.

Commit Limit Values (PAR 60988)

The JADE Version Information utility (jverinfo) now reports the total commitLimit and available commitLimit, rather than the total virtual memory and available virtual memory.

These commitLimit values are the combination of physical memory and page file size.

datmask and sysmask Optional Parameters (PAR 61797)

The JADE Version Information utility (jverinfo) can now contain the optional datmask and sysmask parameters if you want to specify alternative file mask patterns for user data files and system files, respectively.

The jverinfo syntax is now as follows.

```
jverinfo [binpath=binary-directory-name] [binmask=binary-file-type]
[datpath=user-data-directory] [datmask=user-data-file-type]
[out=output-file-name|stdout] [syspath=system-file-directory]
[sysmask=system-file-type] [noThinClients] [noExtraDirs]|
[extraDirs=[path][directory-name]]
```

The default user data file type is *.dat and the default system file type is _*.bin.

Methods View Browser

This section describes the Methods View Browser changes in this release.

JADE Scripts and Test Cases in the Methods View Browser (PAR 60442)

The customized Methods View Browser now enables you to run and debug JadeScript and JadeTestCase subclass methods. (In earlier releases, you could do this from the Class Browser only.)
Methods View Browser Context Menu (PAR 60947)

The JADE Methods View Browser now displays the Methods View menu as the context menu, and this menu now includes the *Open* command.

Monitoring your Active Process Using JADE Sentinel

JADE now provides JADE Sentinel (`jadesentinel.exe`), which monitors your active process. If, for any reason, a fault or problem occurs in this active process, JADE Sentinel takes a diagnostic copy in case the main process is unable to do so itself. No matter what happened to the process, JADE Sentinel ensures that you still get all of the information you need.

The `jadesentinel.exe` is started by any JADE-built or supplied executable. It is a small executable that links to `jomos.dll`, in which all of the work occurs. It runs as a debugger attached to its parent process, looks for specific exceptions, performs out-of-process process dumps, and terminates the program, as appropriate.

The `jadesentinel.exe` is started automatically when the JADE-built executable starts. If it is necessary to start `jadesentinel.exe` manually, it has the following command line syntax.

```
jadesentinel.exe pid=pid ini=path\jade-initialization-name [prefix=executable-name]
```

The `pid` value specifies the process identifier of the process to monitor (that is, the parent process), the `ini` value specifies the path and name of the JADE initialization file, and the optional `prefix` value specifies the prefix to put on the dump file (for example, `jadrap, jade, jadload`, and so on).

The `[FaultHandling]` section of the JADE initialization file contains the following parameter.

```ini
EnableSentinel=<default>
```

The *EnableSentinel* parameter, which is *true* by default, is read when a JADE executable starts up. This controls starting the `jadesentinel.exe` program that monitors the executing parent, and when required to do so, does a process dump. (Process dumps are output to the directory specified in the `ProcessDumpDirectory` parameter of the `[FaultHandling]` section of the JADE initialization file.)

The JADE initialization file now contains the `[JadeSentinel]` section, which provides the following parameter.

```ini
ExcludeDiskCache=<default>
```

The `ExcludeDiskCache` parameter, which is *true* by default, is read when the process dump is about to occur, to exclude disk cache from the process dump.

**Note** You should set these values to *false* only when requested to do so by JADE Support.

ODBC Driver Query Execution Timeout (NFS 60759)

Open Database Connectivity (ODBC) queries executed in the JADE ODBC driver can now be timed out, by using the `QueryTimeout` parameter in the `[JadeOdbc]` section of the JADE initialization file to specify the number of seconds that a query executes before timing out. The default value of zero (0) indicates that there is no timeout. This parameter is valid for the server node in which the ODBC Server application is executing (it sets the default for all queries from all connections) or for the standard (fat) client ODBC driver.

The query timeout can also be set by the ODBC tool submitting the query. This setting overrides the JADE initialization file setting for that statement only.

A new exception 8360 (*Query timeout expired*) is raised when a query timeout has been set for an ODBC query and that time has expired before the query execution completes.

OpenSSL Library

This section describes the OpenSSL library changes in this release.
Application Server OpenSSL Cipher Suite (NFS 61541)

In earlier JADE releases, only the TLSv1 protocol was supported.

JADE now supports TLS 1.1 and TLS 1.2. You can now specify the TLsv1.1 or TLsv1.2 value in the SSLMethod parameters in the [JadeThinClient] and [JadeAppServer] sections of the JADE initialization file. In addition, the methodType property in the JadeSSLContext class now supports the new MethodTLsv1_1 and MethodTLsv1_2 constants defined in the JadeSSLContext class.

Library Version Upgrade (PAR 61368)

The OpenSSL library has been upgraded to version 1.0.1h.

Recompiling All Methods (PAR 61988)

The jadclient executable enables you to automate the running of the JadeRecompileAllMethods non-GUI client application to force a recompile of all methods; that is, methods in error, methods not compiled, and successfully compiled methods.

The JadeRecompileAllMethods application now has the following syntax.

```
jadclient path=database-path
    [ini=jade-initialization-file]
    schema=RootSchema
    app=JadeRecompileAllMethods
    endJade
    [command-line-arguments]
```

The list of command line arguments that you can define after the endJade parameter in command=value; format are as follows.

- allMethods=true|false
- inError=true|false
- notCompiled=true|false

The following example recompiles only those methods that are in error.

```
jadclient.exe path=c:\jade\system ini=c:\jade\system\testjade.ini schema=RootSchema
    app=JadeRecompileAllMethods endJade notCompiled=false
```

The default action is to recompile all methods that are in error or are not compiled (that is, the inError and notCompiled arguments default to true).

As the allMethods argument defaults to false, specifying allMethods=true on the command line now allows all methods to be recompiled, including methods currently compiled without error, methods that have been compiled but are in error, and methods that have not been compiled.

The allMethods=true argument overrides the inError and notCompiled arguments.

References Browser (NFS 61215)

The References Browser for class, property, method, class constant, global constant, and translatable string entities now includes the number of references to the entity when there are two or more references to that entity; for example:

- MySchema::ListTest::ListBox1_displayRow (2)

This indicates that there are two references to the entity in the method ListTest::ListBox1_displayRow.
Relationship View Diagram Printing (PAR 61264)

In earlier releases, the relationship view of classes diagram always printed in landscape mode. This diagram now prints in the orientation mode currently set for the printer to which it is output.

REST-Based Web Services

Earlier JADE releases supported only the Simple Object Access Protocol (SOAP) and WSDL-based Web services. JADE now implements the Representational State Transfer (REST) stateless architecture style as a simpler alternative to SOAP Web services. Mainstream Web 2.0 service providers such as Google, Salesforce, and Facebook have endorsed this easier-to-use, resource-oriented model to expose their services. REST-based Web services, implemented using HTTP, offer a light-weight alternative to the Web services available in earlier releases.

REST works with resources that are identified with a uniform resource identifier (URI). A resource represents a static or dynamically-generated Web page. REST resources are named with nouns as part of the URI rather than verbs; for example, /customers rather than /getCustomers.

To use REST services, a client sends an HTTP request using the GET, POST, PUT, or DELETE verb.

The traditional HTTP error messages (for example, 200 - OK and 404 - Not found) can be used to indicate whether a request is successful. If a request is successful, information can be returned in Extensible Markup Language (XML) or JavaScript Object Notation (JSON) format.

Session handling is not performed, so there is no timeout of connections. Additionally, information is not retained between requests from a client. If that was required, it would need to be provided by the application developer.

For details, see Chapter 11, “Building Web Services Applications”, of the JADE Developer's Reference. See also the JadeRestService class, in volume 1 of the JADE Encyclopaedia of Classes.

Snapshot Database Recovery

You can now recover a database backup that was captured and restored by a third-party snapshot tool; for example, VMWare or Amazon Elastic Block Store (EBS) snapshots.

For details about external third-party snapshot backups, see "Non-JADE Backups" in the Developing a Backup Strategy White Paper.

The third-party tools are responsible for detecting any errors that were raised during the movement of data. A restored snapshot is considered to be database data exactly as it was copied when the snapshot was created. No JADE backup or restore verification checks can be performed.

A roll-forward recovery must be performed to establish database integrity.

A snapshot recovery requires the snapshot processing to happen after the snap begin call, completes before the snap end call, and all journals generated between these two points are required to restore file-level integrity.

When performing a snapshot recovery:

- If a database is accidentally shut down during snapshot processing, do not restart it until the snapshot processing is complete. When restarted, the database will perform a snapshot recovery.
- Ensure that changing the access mode to DB_SNAPSHOT occurs before snapshot processing begins.
- Ensure that snapshot processing completes before restoring access mode to DB_DEFAULT.
The `pMode` parameter of the `jomChangeAccessMode` Application Programming Interface (API) call, documented in Chapter 3 of the JADE Object Manager Guide, now has the `DB_SNAPSHOT`, `DB_ARCHIVE`, and `DB_DEFAULT` mode constants, which conditions the database for external snapshot backup, quiesces the database for archive backup, and restores the database to initial mode and usage values, respectively.

The `JadeDatabaseAdmin` class now provides the `Mode_Exclusive` constant (which requests exclusive access to the database) and the `Mode_Snapshot` constant, which can be specified in the mode parameter of the `changeDbAccessMode` method to condition the database for external snapshot backup.

The `jadbadmin` program enables you to specify `StartSnapshot` and `EndSnapshot` database actions, to condition the database for recovery from a non-JADE backup and to take the database out of snapshot mode, respectively. (For details, see “StartSnapshot” and “EndSnapshot”, respectively, in Chapter 2 of the JADE Database Administration Guide.)

**StringUtf8Array Elements (PAR 60605)**

The maximum length of elements in a user-defined `StringUtf8Array` is 8,000 UTF8 characters.

**System Class isValidProcess Method (PAR 60191, 61755)**

The `System` class now provides the `isValidProcess` method, which has the following signature.

```java
isValidProcess(process: Process): Boolean;
```

This method returns `true` if the process specified by the `process` parameter represents a signed-on application. A `Process` instance without a corresponding signed-on application can exist under certain circumstances; for example, if an error occurs during process sign off that prevents the `Process` instance from being deleted. The `isValidProcess` enables you to identify these `zombie Process` instances.

When a zombie process is encountered in a monitor operation, the instance is deleted; for example, an interrupt or force off user, call stack request, and so on.

**Thin Client Downloads**

This section describes the thin client download changes in this release.

**Thin Client Automatic Download (PAR 61337)**

When a thin client installation was in progress in earlier releases, another initiation of JADE using the same binaries could cause the installation process to fail. JADE uses a file lock when determining the download requirements, but once the thin client installation was initiated, the file lock was lost and the second client could also attempt to perform the installation.

This has now been changed so that the file lock is retained by the JADE thin client install process for standard Windows clients. (This feature is not available in Windows for mobile clients). This prevents multiple clients attempting to perform the same download process. If another JADE thin client is initiated while the first client is processing the download requirements and the application server indicates that a download is required, the file lock is will be detected. If the value of the `AskToDownload` parameter in the [JadeThinClient] section of the JADE initialization file is set to `true`, a message box will be displayed (using a temporary copy of `jaddinst.exe` to do the message box display), informing the client that another client is performing the download and asking him or her to wait. If the value of the `AskToDownload` parameter is `false`, the information is output only to the client `jommsg.log` file. In both cases, the JADE executable terminates with exit code 14165 (The thin client software is currently being downloaded, please wait and try again shortly. The user performing the download is).

**Note** You should not use the same JADE binaries to access different application servers at the same time. If the application servers have different download files, a conflict of interest will occur when two JADE thin clients are initiated at the same time and the installation process will most likely fail with files being in use.
Upgrading 32-bit Thin Client Binaries to 64-bit (NFS 61726)

JADE enables you to upgrade 32-bit presentation (thin) client binaries to 64-bit binaries when the client is using a 64-bit operating system. In addition, you can control the specific architecture of the binaries downloaded to a specific presentation client.

**Note** Visual C++ runtimes are always upgraded (that is, the x64 version is installed) as part of the upgrade process.

The [JadeAppServer] section only of the JADE initialization file on each application server can now contain the following parameter, which defaults to `false`, and is read when the application server starts up.

- **UpgradeRuntimeTo64bit**

When the `UpgradeRuntimeTo64bit` parameter is set to `true`, presentation clients that are running a 32-bit version of `jade.exe` on a 64-bit operating system cause a download and installation of 64-bit binaries to that client (with the exceptions described later in this section). The default value of `false` indicates that 32-bit version presentation clients continue to download 32-bit binaries when required. Presentation clients that are running a 32-bit version of `jade.exe` on a 32-bit operating system continue to download 32-bit binaries when required.

If the client is not running a JADE version that is using Visual Studio 2005 runtime libraries, the JADE thin client download will not revert to a version that uses Visual Studio 2005 runtime libraries.

**Note** The `UpgradeRuntimeTo64bit` parameter applies only after the next thin client download has occurred, because any currently deployed `jade.exe` currently does not communicate the operating system type to the application server. To get 32-bit clients to upgrade to 64-bit binaries therefore requires two download and install processes: one to upgrade the `jade.exe` so that it will identify the operating system type to the application server, and the second to upgrade from 32-bit to 64-bit binaries.

Transient Leak Detection (NFS 60726)

The **Search for creates of selected class in selected schema** option button on the Find Possible Transient Create Leaks dialog (accessed from the **Find Possible Transient Leaks** command in the Schema menu) has been changed as follows.

- The caption for the option button is now **Search for creates of selected class in all schemas**.
- The **Select the schema above to get its list of classes** combo box beneath the **Search for creates of selected class in all schemas** option button enables you to select the class defined in the schema selected in the combo box of the **Search selected schema for all creates** option.

When you click the **Search** button after selecting the **Search for creates of selected class in all schemas** option button and selecting a class in the **Select the schema above to get its list of classes** combo box, all creates of that class in **all** schemas are searched; that is, the search is performed in the schema in which the class is defined and all subschemas. If the class is a **RootSchema** class, all schemas are searched.

In earlier releases, creates were searched for in the selected schema only.

Unaudited Database Files and Partitions

When batch-loading a database file or when generating ad hoc indexes, for example, there is significant auditing overhead. (See also "Ad Hoc Indexes", earlier in this document.) Until the index build is complete, the database file or partition is known to be building (application preserved state) and if the build is restarted, the file or partition is first dropped.

From this release, you can update database files and partitions with auditing disabled, to eliminate journal disk space use and I/O overhead when loading data.
When auditing is re-enabled for a file or partition, a copy of the file or partition is compressed by default, and inserted into the journal. During database roll-forward or replay, the file at the database location is replaced by the file reconstructed from the journal. Subsequent audited updates therefore replay correctly.

**Caution** Disable the auditing of database files and partitions only when restart recovery is not required.

Unaudited operations cannot be used when you have RPS secondaries, as this would result in SQL inconsistencies.

The following table summarizes the methods used to manage unaudited operations.

<table>
<thead>
<tr>
<th>Class</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DbFile</td>
<td>disableAuditing</td>
<td>Disables the auditing associated with object operations performed against the file</td>
</tr>
<tr>
<td>DbFile</td>
<td>drop</td>
<td>Removes the file and marks it as deleted</td>
</tr>
<tr>
<td>DbFile</td>
<td>enableAuditing</td>
<td>Re-enables the auditing associated with object operations performed against the file</td>
</tr>
<tr>
<td>DbFile</td>
<td>isAuditing</td>
<td>Returns <strong>true</strong> if auditing associated with object operations performed against the file is enabled and returns <strong>false</strong> when auditing has been disabled</td>
</tr>
<tr>
<td>JadeDatabaseAdmin</td>
<td>doQuietpoint</td>
<td>Attempts to establish a database quiet point</td>
</tr>
<tr>
<td>JadeDbFilePartition</td>
<td>disableAuditing</td>
<td>Disables auditing associated with object operations performed against the partition</td>
</tr>
<tr>
<td>JadeDbFilePartition</td>
<td>drop</td>
<td>Removes the partition and marks it as deleted</td>
</tr>
<tr>
<td>JadeDbFilePartition</td>
<td>enableAuditing</td>
<td>Re-enables the auditing associated with object operations performed against the partition</td>
</tr>
<tr>
<td>JadeDbFilePartition</td>
<td>isAuditing</td>
<td>Returns <strong>true</strong> if auditing associated with object operations performed against the partition is enabled and returns <strong>false</strong> when auditing has been disabled</td>
</tr>
</tbody>
</table>

In addition, the **DbFile** class now provides the **EnableAudit_NoCompress** class constant.

For details, see Volume 1 of the *JADE Encyclopaedia of Classes.*