JADE™

Platform Differences Guide

VERSION 6.3

Contents

Chapter 3 Portable GUI JADE and Functional Differences, continued

Chapter 4 Multiuser or Client-Server Issues 24

Chapter 5 Differences between Compact JADE and Standard JADE 28

Chapter 6 Windows Vista Support 33

Before You Begin

The JADE *Platform Differences Guide* is intended as a source of information when you are administering JADE systems or running deployed JADE applications.

Who Should Read this Guide

The main audience for the JADE *Platform Differences Guide* is expected to be system administrators.

What's Included in this Guide

The JADE *Platform Differences Guide* has six chapters.

Chapter 1	Covers differences when running JADE applications on a server under UNIX
Chapter 2	Covers Microsoft assumptions that differ under UNIX
Chapter 3	Covers portable GUI JADE and functional differences
Chapter 4	Covers multiuser and client-server issues
Chapter 5	Covers differences between Compact JADE and standard JADE
Chapter 6	Covers Windows Vista support

Related Documentation

Other documents that are referred to in this guide, or that may be helpful, are listed in the following table, with an indication of the JADE operation or tasks to which they relate.

Title	Related to…
JADE *Database Administration Guide*	Administering a JADE database
JADE *Development Environment User's Guide*	Using the JADE development environment to development JADE applications
JADE *Encyclopaedia of Classes*	System classes (Volumes 1 and 2), Window classes (Volume 3)
JADE *Encyclopaedia of Primitive Types*	Primitive types and global constants
JADE *Initialization File Reference*	Maintaining JADE initialization file parameter values
JADE *Installation and Configuration Guide*	Installing and configuring JADE
JADE *Runtime Application Guide*	Administering JADE deployed runtime applications
JADE *Thin Client Guide*	Administering JADE thin client environments
JADE *Web Application Guide*	Implementing, monitoring, and configuring Web applications

Conventions

The JADE *Platform Differences Guide* uses consistent typographic conventions throughout.

Convention	Description
Arrow bullet (➢)	Step-by-step procedures. You can complete procedural instructions by using either the mouse or the keyboard.
Bold	Items that must be typed exactly as shown. For example, if instructed to type **foreach**, type all the bold characters exactly as they are printed.
	File, class, primitive type, method, and property names, menu commands, and dialog controls are also shown in bold type, as well as literal values stored, tested for, and sent by JADE instructions.
Italic	Parameter values or placeholders for information that must be provided; for example, if instructed to enter *class-name,* type the actual name of the class instead of the word or words shown in italic type.
	Italic type also signals a new term. An explanation accompanies the italicized type.
	Document titles and status and error messages are also shown in italic type.
Blue text	Enables you to click anywhere on the cross-reference text (the cursor symbol changes from an open hand to a hand with the index finger extended) to take you straight to that topic. For example, click on the "ASCII File Conventions" cross-reference to display that topic.
Bracket symbols ([ ])	Indicate optional items.
Vertical bar (|)	Separates alternative items.
`Monospaced font`	Syntax, code examples, and error and status message text.
ALL CAPITALS	Directory names, commands, and acronyms.
SMALL CAPITALS	Keyboard keys.

Key combinations and key sequences appear as follows.

Convention	Description
KEY1+KEY2	Press and hold down the first key and then press the second key. For example, "press SHIFT+F2" means to press and hold down the SHIFT key and press the F2 key. Then release both keys.
KEY1,KEY2	Press and release the first key, then press and release the second key. For example, "press ALT+F,X" means to hold down the ALT key, press the F key, and then release both keys before pressing and releasing the X key.

In this document, the term Microsoft *Windows* refers to Windows 2003 Server, Windows Vista, Windows XP, Windows 2000, or Windows CE. When there are differences between the versions of Microsoft Windows, the specific version of Microsoft Windows is stated. This also applies to Linux, which is a specific version of UNIX developed by SUSE or Red Hat. The term *UNIX* is used when an issue is generic to all versions of UNIX and the term *Linux* is used if the issue is specific to the SUSE or Red Hat implementation of UNIX.

With the exception of the **jade.exe** program, when referring to Windows program executables in this document, the **.exe** file suffix is omitted; for example, **jadclient** refers to **jadclient.exe** on Windows and **jadclient.sh** on UNIX. Similarly, the Windows **.dll** (Dynamic Link Library) and UNIX **.so** (shared object library) file suffixes are omitted. For example, **jomos** refers to **jomos.dll** (Windows) and **libjomos.so** (Linux).

Chapter 1 Differences When Running JADE Applications on a Server under UNIX

This chapter covers the differences between JADE applications running under a Microsoft Windows or a SUSE or Red Hat operating system on a UNIX platform.

Note Server methods are prevented from executing code that requires GUI objects (for example, forms and controls), because the server runs outside of the JADE GUI engine that implements these objects.

ActiveX Type Library Support

As ActiveX is a Windows product, ActiveX automation server type libraries cannot be imported into JADE running on UNIX.

Application Class *playSound* and *playSoundAsync* Methods Not Supported

The **Application**::**playSound** method (which plays the specified **.wav** file and returns when the sound file has been played) and the **playSoundAsync** method (which starts the playing of a specified **.wav** file and then returns immediately) are not available when invoked from a method running under the SUSE or Red Hat operating system on UNIX.

ASCII File Conventions

As UNIX uses different **Cr**/**Lf** conventions from those of Microsoft Windows for ASCII files, a single **Lf** character is used to mark the end of line. However, JADE handles text in the **TextBox** class with the **CrLf** and the **Lf** end-of-line sequences in both Windows and Linux environments.

External Library and Function Calls

Although external library and function calls are architecture-dependent, these are supported on server nodes running under UNIX but they cannot be ported from a Windows operating system to the UNIX operating system or the reverse. For example, the **user32.dll** in Windows does not exist under UNIX, which has shared object libraries (**.so** files) rather than dynamic link libraries.

Calling external functions from JADE on UNIX platforms has the following restriction.

- You can have no more than 13 32-bit arguments.

  Each **Integer**, **Boolean**, **Character**, **String**, and **Binary** primitive counts as one 32-bit argument, but a **Real** primitive counts as two 32-bit arguments. If you are therefore passing three **Real** values, you can have a maximum of seven other arguments.

External Process Differences

Differences exist when defining external processes for a JADE application running on a server under UNIX; for example, command line processing.

File and Directory Path Names

The virgule, or forward slash, separator (/) is used in file and directory paths in the UNIX operating system, in contrast to the backslash character (\ ) used as a separator in file and directory path names on a workstation running under a Microsoft Windows operating system.

Tip As the virgule separator (/) used in the UNIX operating system can be used as the separator in file and directory paths in a Windows operating system, you may wish to change existing backslash character separators (\ ) to virgule separators (/) so that file and directory paths may be recognized by both Windows and UNIX.

The exception to this is Universal Naming Convention (UNC) names, in which the separator character preceding a host name and a service under a Windows operating system must be a backslash character; for example:

```
\\wilbur1a\public/jade/system/_userscm.dat
```

UNC names are not supported on UNIX, nor does UNIX have any concept of disk drive letters. File and directory path names in UNIX operating system are case-sensitive.

File Naming Conventions

The following table lists the differences (and provides examples) in file naming conventions under a Windows operating system and a SUSE or Red Hat operating system on UNIX.

File Type	Windows	UNIX
Executable programs	*program-name*.exe (**jdbutilb.exe**)	No file suffix (**jdbutilb**)
Libraries	*library-name*.dll (**jomdb.dll**)	*library-name*.so (**libjomdb.so**)

The suffix of .**dat** is used for database files under both Windows and UNIX operating systems.

See also "Directory Locations", in Chapter 5 of the JADE *Installation and Configuration Guide*.

Internet Interface

The **jadehttp** library file is not supported on server nodes running under UNIX, as it uses the named pipe communications protocol that does not exist on UNIX platforms.

However, you can use the Apache HTTP Server to connect from a Web browser to JADE applications running under a Windows, SUSE, or Red Hat operating system. (For details, see "Connecting to JADE Applications from an Apache HTTP Server", in Chapter 5 of your JADE *Installation and Configuration Guide*.)

JadeRichText Control Features

For a list of the **JadeRichText** control facilities that are not supported by Java and therefore do not apply when running in a portable GUI environment, see "Portable GUI JADE and Functional Differences" under "Microsoft Windows Assumptions that Differ Under UNIX", later in this chapter. (The **JadeRichText** control applies only to client nodes.)

Multiple Locale Support

Multiple concurrent locales are not supported on server nodes running under UNIX. (Windows operating systems support a locale for each thread, whereas UNIX supports a locale for each process.)

Note By default, the UNIX operating system does not recognize the English Australian and English New Zealand locales. For details about using these locales for SUSE Linux Enterprise Server or the Red Hat distribution of Linux, see "Locales and Translatable Messages", in Chapter 3 or Chapter 4 of your JADE *Installation and Configuration Guide*, respectively.

NamedPipe and InternetPipe Classes Not Supported

The **NamedPipe** class and its **InternetPipe** subclass are not supported on server nodes running under the UNIX operating system, as the equivalent Win32 named pipes support does not exist.

Portable Document Format (PDF) Hyperlink Handling

When using **evince** to allow the built-in JADE development environment help to work on Linux, the following versions are provided with the Linux distribution.

- SUSE Linux Enterprise Server 10 (SLES 10) **evince** version 0.4.0, which displays PDF hyperlinks but does nothing with them
- Red Hat Enterprise Linux 5 (RHEL 5) **evince** version 0.6.0, which supports following hyperlinks within a document and between other documents in the JADE product information library

Portable Printing

The **PrintFileFormat** and **PrintDataType** parameters, respectively, in the [JadePrinting] section of the JADE initialization file specify the meta file format and print data type for portable printing. JADE print data can be saved in the database in the following formats.

- Scalable Vector Graphics (SVG), which is the default value on all operating systems

  Caveat As the contents of **ActiveXControl**, **JadeRichText**, and **OleControl** controls are displayed by the Windows operating system Application Programming Interface (API) calls, they cannot generate SVG format files. When one of these controls is printed to an SVG meta file, a picture is therefore created from the controls that are currently displayed and it is this picture that is stored in the SVG file, which may result in a slightly lower-quality display.

  In addition, if you perform a find operation from a print preview screen, JADE cannot search in these controls.

- Windows Enhanced Meta Files (EMF)

Data can be sent to the printer in the following print data formats.

- Postscript (PS), which is the default value on UNIX
- Windows Graphical Device Interface (GDI) commands, which is the default value on Windows

The combinations of GDI, PS, and SVG formats that are permitted depend on the operating system and executable on which the JADE system is running.

Windows users can choose SVG or EMF to save print commands in the database. UNIX users can save files only in the SVG format. Not all options are available under all operating systems and executables.

Valid combinations are listed in the following table.

Operating System	Executable	Print File Format	PS Data Format	GDI Data Format
Windows	jade.exe	SVG	Yes	Yes
		EMF	No	Yes
	jadepg.exe	SVG	Yes	Yes
		EMF	No	No
UNIX	jadepg	SVG	Yes	No
		EMF	No	No

The format of the data sent to the printer depends on the format in which the command data has been saved and the operating system and executable that is used. If the commands are saved as EMF under Windows, the output can be GDI format only. If commands are saved as SVG when running under Windows, printer output can be GDI or PS, while only the PS data type is available on UNIX.

For more details, see "Portable Printing" under "Printer Class", in Chapter 1 of your JADE *Encyclopaedia of Classes*. See also "JADE Printing Section [JadePrinting]", in your JADE *Initialization File Reference*. See also "Portable GUI JADE and Functional Differences" under "Microsoft Windows Assumptions that Differ Under UNIX", later in this chapter.

JADE applications written in a Windows environment rely on the Windows environment to provide specific common fonts. When a JADE application written under Windows runs in a portable GUI non-Windows environment, none of the Windows standard fonts are normally available. For details about substituting preferred fonts on client nodes for those defined in an application, see "Font Substitutions Section [JadeFontSubstitutions]", in your JADE *Initialization File Reference*.

Starting and Stopping a Server

The starting and stopping of a server differs under Windows and UNIX.

For details about the JADE Remote Node Access utility for servers running under Windows, see Chapter 1, "Using the JADE Remote Node Access Utility", in your JADE *Remote Node Access Utility User's Guide*. For details about starting and stopping a server under UNIX, see Chapter 3 or Chapter 4 of your JADE *Installation and Configuration Guide*.

Chapter 2 Microsoft Windows Assumptions that Differ under UNIX

This chapter covers assumptions or practices in a Windows operating system that may not function as expected when JADE is running on a UNIX platform.

Note The issues outlined in this chapter are assumptions only, and not differences between JADE running under a Microsoft Windows or a UNIX operating system (which are documented in Chapter 1, "Differences When Running JADE Applications on a Server under UNIX".)

With the exception of the **jade.exe** program, when referring to Windows program executables, the **.exe** file suffix is omitted. Similarly, the Windows **.dll** (Dynamic Link Library) and UNIX **.so** (shared object library) file suffixes are omitted. For example, **jomos** refers to **jomos.dll** (Windows) and **libjomos.so** (Linux).

Architecture Awareness

To determine the architecture in which a method or application is running, use the **Node** class **getOSPlatform** method. For details, see "Node Class", in Chapter 1 of your JADE *Encyclopaedia of Classes*.

Database Files

In JADE applications running on a server under UNIX, the database files are physically different from those in an application that runs under Microsoft Windows. These database files cannot therefore be copied from a Windows platform (any Windows node) to a UNIX platform, or the reverse.

Note Read-only schema files are not supported on JADE running on UNIX servers.

For details about using the **jadclient** program to convert data between ANSI and Unicode formats in a JADE user database, see Chapter 4, "Converting a User Database", of your JADE *Runtime Application Guide*.

Directory in Which the Server is Running

JADE server nodes running under a Microsoft Windows operating system are usually initiated in the **bin** directory. However, JADE server nodes running under UNIX can be initiated in any directory.

The JADE executable program **jade.exe** running under Windows is initiated in the working directory in which the JADE binary files are installed.

The server program **jaded** running under UNIX can be started and stopped in any location. (For details, see Chapter 3 or Chapter 4 of your JADE *Installation and Configuration Guide*.)

In JADE systems running under UNIX, standard output is directed to **stdout** and error information is directed to **stderr**.

read and *write* Instructions

In a JADE application running on a server under UNIX, the input to the **read** instruction comes from **stdin** and the **write** instruction is output to **stdout** (and **stderr**, if applicable).

Relative Directory Handling in the Command Line *path* Parameter

When a relative path name is used in the **path** parameter of the command line, the path name is first converted to an absolute path, by using the following rules.

- On Windows only, a relative path name with a single leading slash character is pre-pended by the first two characters of the JADE HOME directory (that is, ***drive-letter:***).
- Path names with no leading slash character are pre-pended by the JADE HOME directory.

  On Windows, the JADE HOME directory is assumed to be the parent of the **bin** directory.

In the following Windows examples, the JADE HOME directory is assumed to be **c:\jade**.

Path Specified in the Command Line	Actual Path
path=/jade63/system	c:/jade63/system
path=system	c:/jade/system

In the following UNIX examples, the JADE HOME directory is assumed to be **/jade**.

Path Specified in the Command Line	Actual Path
path=/jade63/system	/jade63/system
path=system	/jade/system

Serial Port Names

Windows serial ports, also known as COM ports (communication ports), are generally called **COM1**, **COM2**, and so on.

UNIX serial ports generally have the naming convention shown in the following examples.

```
/dev/ttyS0
/dev/ttyS1
```

Sorting Text in Tables and List Boxes

Windows sorts text in tables and list boxes by using the collating sequence of the locale that is in use, which is not necessarily the same as binary value ordering.

As UNIX knows little about locales and collating sequences, it sorts text in tables or list boxes by using a binary value evaluation only.

Thin Client Default File Location

By default, the **jade.exe** and **jadepg** executable program files do not directly create or update files in the binary directory.

Thin client file locations are as follows.

- The **JadeWorkDirectory** parameter in the [JadeEnvironment] section of the JADE initialization file determines the directory in which work files are created.

  By default, this directory is created at the same level as the JADE binary directory and is named **temp** on Windows and **tmp** on Linux. For example, the directory is named **c:\Jade\temp** if the JADE installation directory is **c:\Jade\bin** on Windows.

  The **JadeWorkDirectory** parameter can specify an absolute path or a relative path (relative to the JADE HOME directory, which is **c:\Jade** in the above example).

- The presentation cache file is written into the directory defined by the value of the **JadeWorkDirectory** parameter unless the **FormCacheFile** parameter in the [JadeThinClient] section of the JADE initialization file specifies the location of the form cache file.

- The presentation client automatic download process lock files are created in the directory specified by the **JadeWorkDirectory** parameter.

- The automatic presentation client download **log** file is written to the location specified by the **LogDirectory** parameter in the [JadeLog] section of the JADE initialization file.

The only situations in which a file creation or update will occur within the JADE binary directory are as follows.

- If the JADE initialization file is positioned in the binary directory (the default action for a presentation client).

  You can avoid this by using the **ini** parameter in the JADE presentation client command line to specify an alternate location on initiation of the presentation client.

- When the **jaddinst** executable program installs files downloaded by the JADE thin client automatic download facility.

Unicode Character Length

The size of a JADE Unicode character is 4 bytes on Linux and 2 bytes on Windows.

Unicode characters in strings are stored as logical lengths rather than physical lengths.

Chapter 3 Portable GUI JADE and Functional Differences

This chapter summarizes the JADE and functional areas in a portable GUI environment that differ from those in a Windows GUI environment. See also "Portable Printing" in Chapter 1, "Differences When Running JADE Applications on a Server Under UNIX".

- Application::*alert* Method and Global::*alert* Method
- Application Class *playSound* and *playSoundAsync* Methods
- Close Button on Forms
- Closing Open Menus and Combo Boxes
- Common Dialogs
- Control Subclasses
- Form::*alwaysOnTop* Method
- Form::*registerWindowMsg* Method
- Interrupt Menu
- *keyDown* Events
- Meta File Images not Handled
- Minimize and Maximize Buttons on Forms
- *newcopy* Command Line Option
- Paint Requests
- Region Handling
- Skins
- Sound Class *isPlayable*, *loadFromFile*, and *play* Methods
- System Tray Handling
- Undo and Redo Functionality
- Using the Task Manager to Close a Portable GUI JADE Application
- Window::*doWindowEvents* Method
- Window::*drawDeskTopRectangle* Method

- Window::*drawMode* Property
- Window::*getPoint* Method
- Window::*hwnd* Method

JADE applications written in a Windows environment rely on the Windows environment to provide specific common fonts. When a JADE application written under Windows runs in a portable GUI non-Windows environment, none of the Windows standard fonts are normally available. For details about substituting preferred fonts on client nodes for those defined in an application, see "Font Substitutions Section [JadeFontSubstitutions]", in your JADE *Initialization File Reference*.

Application::*alert* Method and Global::*alert* Method

The **Application** class **alert** method and the **Global** class **alert** method are not supported and will be ignored, as Java has no sound catalogue.

Application Class *playSound* and *playSoundAsync* Methods

The **Application** class **playSound** and **playSoundAsync** methods are not supported and are ignored.

Close Button on Forms

Although the **Form** class **allowClose** method is honored for non-MDI forms, Java always paints the form **Close** button as enabled unless the form is skinned. In addition, the system menu **Close** command is always shown enabled, which may confuse users.

Closing Open Menus and Combo Boxes

Clicking on the caption of a non-skinned top-level form does not close an open menu or a combo box unless the form is moved or one of the form buttons is clicked, because Java does not generate any event to notify JADE that the caption **click** event occurred.

Common Dialogs

Some common dialog properties are not supported; for example, Windows standard fonts.

When the **open** method for a common dialog is called in a portable GUI environment, *all* open forms for *all* applications running out of the same **jadepg** executable program are disabled and cannot be accessed until the common dialog is closed (because Java has no concept of separate totally asynchronous multiple applications running side by side.)

JADE has coded around this limitation for all standard JADE forms. However, because the common dialog forms are pure Java forms taken from the Java toolkit, they cannot be forced into the JADE operational mode. An additional side-effect of this is that if the common dialog **open** method call is attempted to be stepped through (using the F7 or F8 key, for example) in the JADE debugger, the system totally locks up because the JADE debugger has been disabled and the application forms are suspended (a concept Java does not normally allow but JADE has been able to make function for JADE forms only). This limitation also applies to any other Java forms referenced in the future or if your application invokes Java forms outside of the JADE definition.

Control Subclasses

The following differences to **Control** subclasses apply to Portable GUI JADE.

- As ActiveX controls are not supported by Java, **ActiveXControl** controls are handled as empty windows. All **ActiveX**-related property accesses and method calls are ignored.
- As Windows Component Object Model (COM) is not supported by Java, **Ocx** and **OleControl** controls are handled as empty windows. All **Ocx**-related and **OleControl**-related property accesses and method calls are ignored.
- The **MultiMedia** control class has the following differences and limitations.
  - It does not support the same set of formats as the Microsoft Windows version. It supports some of the many Audio Video Interleaved (AVI) file types, but not Microsoft Windows AVI files. In addition, it does not support files with embedded Joint Photographic Experts Group (JPEG) images that are not 8-bit (which may play, but will be distorted).
  - The **playReverse** method is not supported and is treated the same as the **play** method.
  - The **notifyPosition** event is not generated.
  - The **record** and **save** method functions are not available.
  - The **showMenu**, **showName**, **showOpenMenu**, **showPosition**, and **showRecord** properties are ignored.
- The following **JadeRichText** control facilities are not supported by Java.
  - The **autoURLDetect** property
  - The **bulletIndent** property
  - The **bulletStyle** property
  - The **getLine** method **lineHeight** parameter
  - The **getTextProtection** and **setTextProtection** methods, which raise exception 14217 (*This feature is not available in this environment*) when they are called in a portable GUI environment
  - The **selTextRTF** property contains a text string only (that is, it does not handle an RTF string)
  - The **insertObject** method (including images)
  - The **insertObjectDialog** method
  - The **insertTable** method
  - The **linkClicked** event method
  - The **loadFromFile** method cannot replace the current selection
  - The **objectPropertiesDialog** method
  - The **protected** event method

- The **Load_ReplaceSelection** or **LoadFromFile_ReplaceSelection** constant value in the **replace** parameter of the **load** or **loadFromFile** method, respectively, handles only a text string and not rich text
- The **saveInFile** method for current selection does not support partial RTF access
- The **selChanged** event is not called
- The buffer of the **undo** method includes all programmatic changes
- The **zoom** property

- The **JadeXamlControl** control class is not supported.

Form::*alwaysOnTop* Method

The **Form** class **alwaysOnTop** method is not supported and is ignored.

Form::*registerWindowMsg* Method

The **Form** class **registerWindowMsg** method is not supported and is ignored.

Interrupt Menu

The JADE User Interrupt system menu is not available.

keyDown Events

Changes to the value of the **keyCode** parameter by JADE logic during a **keyDown** event has no impact on the value of the **keyCharCode** parameter of the subsequent **keyPress** event or the value of the **keyCode** parameter of the subsequent **keyUp** event.

Meta File Images not Handled

Meta file image handling is not supported by Java. Use of meta file images (that is, **Window.PictureType_MetaFile**) generates an exception when running JADE in a portable GUI environment.

Minimize and Maximize Buttons on Forms

The value of the **borderStyle** property of a top-level form controls whether the form has minimize and maximize form buttons if the form is not skinned. (This does not apply to MDI children).

If the border style allows resizing, the form has minimize and maximize buttons. If the border style does not allow resize, the form has no minimize and maximize buttons. The **maxButton** and **minButton** properties are ignored.

If the form is skinned, the default form system menu reflects whether Java allows maximize and minimize operations according to the value of the **borderStyle** property.

newcopy Command Line Option

The **jadepg** executable with the **newcopy** command line value is not supported and is ignored.

Paint Requests

Java generates a paint request for the smallest rectangle that encloses the required paint areas. GUI changes in a **paint** event therefore cause continuous painting if the resulting paint rectangle includes the same window again.

Region Handling

The **JadeMask** class **createRegionFromMask** and **JadeSkinWindow** class **myImageMask** properties are ignored, as they require use of Windows region functions.

Skins

Because the native window manager is not in use when skins are in use, the menu options to manipulate the window are not available (for example, when moving the form to another desktop).

Sound Class *isPlayable*, *loadFromFile*, and *play* Methods

The **Sound** class **loadFromFile** and **play** methods are not supported and will be ignored. The **isPlayable** method always returns **false**.

System Tray Handling

As system tray handling is not supported:

- The **Form** class **hasSystemTrayEntry** method returns **false**.
- The **Form** class **setSystemTrayEntry** and **removeSystemTrayEntry** methods are ignored.

Undo and Redo Functionality

Each portable GUI text box control can undo and redo the last 100 actions.

In addition to the number of undo and redo actions available, the other differences from standard Windows functionality are as follows. (These differences are Java limitations.)

- Java stores the undo actions at a finer granularity than Windows; for example, if text is selected when another character is entered, two undo requests are necessary to undo the action.

  The first undo request undoes the character entry and the second one undoes the selected text deletion. Note, however, that the undo action does *not* result in the original text being selected again.

- The cursor is not always restored to the position at which the action undo action occurred.
- The undo action does not re-select the text that was originally selected.

Using the Task Manager to Close a Portable GUI JADE Application

The portable GUI presentation client runs on top of Java. When the Task Manager is used to close a portable GUI JADE application, Java closes the application in an abrupt manner without giving JADE the opportunity to close gracefully.

As a result, the presentation client connection to the application server gets closed as though the connection has been lost. When the application server eventually realizes that the connection is no longer there, it:

1. Waits for a reconnection (with the application remaining in the JADE Monitor for the number of minutes specified by the **KeepAliveTime** parameter in the [JadeAppServer] section of the JADE initialization file)
2. Aborts any current processing (generating exceptions for the **showModal** call, for example)
3. Closes the application

Window::*doWindowEvents* Method

Although the **Window** class **doWindowEvents** method in a portable GUI environment behaves the same as the method in a Windows GUI environment in most cases, note the following differences in a portable GUI environment.

- Keyboard and mouse events are discarded if they are not for the window or windows involved. All other events are processed for all windows; for example, **paint** or **activate** events. (In a Windows GUI environment, all events not for the window or windows involved are left in the queue and are handled when normal event processing resumes.)
- If the form is not skinned, the user can move, resize, minimize, or maximize forms but cannot close a form. This is because JADE never receives any mouse events in the caption or border area of a non-skinned form and events therefore cannot be discarded.

Window::*drawDeskTopRectangle* Method

The **Window** class **drawDeskTopRectangle** method draws only on forms belonging to the current application, as Java does not permit applications to draw on the environment's desktop directly.

This affects dragging docking controls so that only that portion of a dragged docking control over an application form can be drawn. When the drag process ends, the required drawing in the associated window occurs.

Window::*drawMode* Property

The **Window** class **drawMode** property value in some cases may not produce exactly the same result as that produced under Windows, as follows.

- The values **DrawMode_Copy** (13), **DrawMode_Black** (1), and **DrawMode_White** (16) are fully supported.

- The values **DrawMode_Xor** (7) and **DrawMode_Invert** (6) should produce the required result except that the generated drawing color is different.
- The value **DrawMode_NotXorPen** (10) is treated the same as **DrawMode_Xor** (7).

Although all other values attempt to perform the required actions using Java in-built facilities, they are not guaranteed to work correctly.

Window::*getPoint* Method

The **Window** class **getPoint** method is not supported, and returns **-1**.

Window::*hwnd* Method

The **Window** class **hwnd** method always returns zero (**0**), as it is a *Windows* handle to a form or control.

Chapter 4 Multiuser or Client-Server Issues

This chapter covers the differences between JADE running on client and server nodes that occur because of multiuser versus single user issues, rather than those that relate directly to server nodes running under UNIX as opposed to a Windows operating system.

Overview

An exception is raised or an incorrect result is returned if any of the following methods is invoked from a server method.

Application::globalLockException	Application::msgBox	Collection::inspect
Collection::inspectModal	Exception::debug	Form::showModal
Form::popupMenu	Object::inspect	Object::inspectModal
ObjectArray::limitExceeded	Process::debug	RootSchemaApp::msgBox

The **Application** class **startApplication**, **startApplicationWithParameter**, and **startAppMethod** methods invoked from a server method or server application start only applications of type **ApplicationType_Non_GUI** or **ApplicationType_Non_GUI_Web**. On a client node, they start all types of application.

Non-GUI applications (that is, those whose **Application** class **applicationType** property is set to **ApplicationType_Non_GUI** or **ApplicationType_Non_GUI_Web**) for standard (fat) clients are run as GUI applications that do not display forms; that is, **ApplicationType_GUI_No_Forms**. Form creation raises an exception unless the application is in exception state.

- When running JADE in standard mode, non-GUI applications behave as follows.
- In a Windows environment, the application displays an **Interrupt** button on the taskbar.

- Creation of a form is permitted while in exception state so that the **Debug** button on the exception dialog functions correctly.
- The JADE executable program (**jade.exe**) does not exit while non-GUI applications are running.

Debugging and Inspecting Server Methods

Server methods can be neither debugged nor inspected in a multiuser environment on a server node.

Executable Programs Not Supported

The following executable GUI programs are not supported in multiuser mode on a server node.

- **jdbutil** (JADE Database utility)
- **jadload** (Schema Load utility)
- **jddlutl** (Dump and Load utility)
- **jadapp** (JADE application server)
- **jadclient** (JADE non-GUI client)
- **jadreg** (JADE Registration utility)

Batch versions of these utilities are available (that is, **jdbutilb**, **jadloadb**, **jddlutlb**, **jadappb**, **jadclient** on both a Windows and a UNIX platform, and **jadregb**)raised or an incorrect result is returned if any of the following methods is invoked. For details, see the following respective sections.

- "Running the JADE Database Utility in Batch Mode", in Chapter 1 of your JADE *Database Administration Guide*
- "Loading a Schema and Forms in Batch Mode", in Chapter 1 of your JADE *Schema Load Utility User's Guide*
- "Running the Dump and Load Utility in Batch Mode", in Chapter 1 of your JADE *Dump and Load Utility User's Guide*
- "Running an Application Server on UNIX" and "Running a Non-GUI Client Application on UNIX" in Chapter 3 (SUSE Linux Enterprise Server on Intel) or Chapter 4 (the Red Hat distribution of Linux on Intel), of your JADE *Installation and Configuration Guide*
- "Running a Non-GUI Client Application using jadclient", in Chapter 1 of your JADE *Runtime Application Guide*
- "Reregistering a JADE System in Batch Mode", in Chapter 1 of your JADE *Installation and Configuration Guide*

Message Box Display on a Server Node

As the **Application**::**msgBox** method has no meaning in a server node environment, a message box is always executed on the client node or presentation client workstation, even if it is called from a server method.

Print Facilities on Server Nodes Not Supported

Print facilities (the **Printer** class and **CMDPrint** class) are not supported on server nodes. An exception is raised if a printing operation (for example, calling the **Printer**::**setPrinter** method to set the output printer) is invoked from any of the following.

- A **serverExecution** method.
- A server application running under the **jadrap.exe** JADE Remote Node Access utility (because printing requires the **jade.exe** program).

Remote Node Access Utility Not Supported

As the JADE Remote Node Access utility (**jadrap**) is a GUI utility, it cannot run under the SUSE or Red Hat operating system on UNIX. For details about the **jaded** daemon for JADE running on UNIX servers, see Chapter 3 (SUSE Linux Enterprise Server on Intel) or Chapter 4 (the Red Hat distribution of Linux on Intel), of your JADE *Installation and Configuration Guide*.

Support of Classes and Methods Implemented in jadpmap Library

You can access all GUI properties and methods (which are marked as **clientExecution** methods) from a server method except for anything that brings up a modal type dialog; that is, **app.msgBox**, **Form**::**showModal**, **Form**::**popupMenu**, and the common dialog class methods.

The other exceptions to this are the **app.doWindowEvents**, **app.checkPictureFile**, and **app.loadPicture** methods, which are executed relative to the server.

Caution Use of GUI methods and properties is *very* expensive in a server method. A **clientExecution** method requires that all transient objects passed to the server are passed back with the client execution (and passed back to the server after the client execution is complete). Accessing GUI properties and methods within a server execution should therefore be done only in exceptional circumstances.

Class Availability

An exception is raised or an incorrect result is returned if any of the following methods is invoked classes listed in the following table are not available in a JADE application running on a server.

Class	Windows Single User	Windows Client	Windows Server	Windows CE	UNIX Client	UNIX Server
External database	Yes	Yes	Yes	No	Yes	Yes
GUI-classes	Yes	Yes	No	Most	Yes	No
Locale classes	Yes	Yes	No	Yes	Yes	No
Common dialog	Yes	Yes	No	Yes	Yes	No
InternetPipe	WE (see below)	WE (see below)	WE (see below)	No	No	No
NamedPipe	Yes	Yes	Yes	No	No	No
OleObject	Yes	Yes	Yes	No	No	No
Printer	Yes	Yes	No	No	Yes	No
Sound	Yes	Yes	Yes	Yes	Yes	No
WebSession	WE (see below)	WE (see below)	WE (see below)	Yes	Yes	Yes

In this table, **WE** indicates a Windows Enterprise operating system, which can be Microsoft Windows 2003 Server, Windows Vista, Windows XP, or Windows 2000.

A **NamedPipe** server can be run only under a Windows operating system that supports services. Client nodes can be running under any Windows operating system.

Chapter 5 Differences between Compact JADE and Standard JADE

This chapter summarizes the differences between a Compact JADE and a standard JADE implementation.

Overview

When running a Compact JADE application, the following differences from a standard JADE application apply.

- Thin client or single user installations only.

  Both the Personal Digital Assistant (PDA) device and emulated version on the desktop versions of Compact JADE support running JADE only as a presentation client or a single user node.

  The presentation client can connect to standard JADE application servers, which can be ANSI or Unicode format.

- The following multiuser features are not available in Compact JADE.
  - Standard clients
  - Application servers
  - Synchronized Database Service (SDS)
  - Relational Population Service (RPS)
  - ODBC external database
  - Web service provider applications
- PDA and desktop implementations of Compact JADE on presentation clients and single user nodes are Unicode format without the following standard JADE functionality.
  - JADE development environment
  - JADE method compiler
  - JADE editor
  - ActiveX and OLE

- Web service provider, Web-enabled forms, HTML forms, and the Web Monitor
- External databases and relational views
- JADE Report Writer application
- Printing
- Multimedia
- Rich text
- Multiple Document Interface (MDI)

- Compact JADE GUI on a PDA device under Windows CE does not provide the Color dialog and Font dialog controls and form-related functionality.
- Compact JADE on a PDA device does not implement the following standard JADE functionality.
  - Schema Load utility (**jadload**)
  - Database reorganization
  - Database dump and load utility (**jddlutl**)

  To perform these tasks, use the desktop version of Compact JADE.
- JADE initialization files are not used on a Windows Mobile device.

  On a Windows Mobile device, the **Application** class **getProfileString** and **setProfileString** methods retrieve and set values in the registry, not a physical file.
- As the file format of the database journals is different on the PDA and desktop devices, any database recovery that needs to occur in a Compact JADE environment must occur on the device that needs the recovery.
- When the **File** class **open** or **openInput** method is called on Windows Mobile devices and the value of the **File** class **kind** property is set to **Kind_Unknown_Text** and the file does not have a Byte Order Mark (BOM) at the start of the file, if the file is zero (**0**) or **1** byte in length, the file is treated as **Kind_ANSI**; otherwise it is **Kind_Unicode**. No attempt is made to detect the file type by reading the first part of the file.

  Although you can read and write files of other kinds on Windows Mobile devices, you must explicitly specify the kind of file.
- Auditing of collection block deltas is disabled.

GUI Differences when Running Compact JADE

Notes You can run only a deployed JADE application on Compact JADE. The JADE development environment is not supported.

Compact JADE does *not* use JADE initialization files. For details about configuring JADE on a Windows Mobile device, see "Configuring Compact JADE", in Chapter 2 of the JADE *Installation and Configuration Guide*.

When you run Compact JADE:

- As you cannot override the default **SplashScreenFile** configuration parameter in single user mode, the default splash screen is always displayed when you run a Compact JADE application in that mode. (You can change the splash screen when the application is run in thin client mode.)
- A hand-held device generates **mouseMove** events only when the mouse is down.
- Printing is not implemented, and is ignored or it raises an exception.
- We recommend that the **Form** class **alwaysOnTop** method should not be used, as the display of new forms, including the message box, will probably be behind this form.
- Windows metafile (**.wmf**) and enhanced metafile (**.emf**) image files are not supported and their use generates a not available exception.
- The **JadeRichText** control class is not handled, and it generates an unavailable exception.
- The **ActiveXControl**, **JadeEditor**, **JadeTextEdit**, **JadeXamlControl**, **MultiMedia**, **Ocx**, and **OleControl** control classes and any associated methods on the **Application** class are not handled, and generate an unavailable exception.
- Calls to the **CMDFont** class generate an unavailable exception, as the dialog does not exist.
- Font substitution is not supported and is ignored.
- The interrupt icon support is not available.
- Accessibility support is not available.
- Windows Compact devices do not support images on submenu items. Images can be displayed only on top-level items that are displayed on the menu bar.
- The Multiple Document Interface (MDI) is not supported and is ignored.
- **Form**::**setFormParent** is not supported and its use generates a not available exception.
- The **Window** class **drawTextCharRotation** and **drawTextRotation** graphics properties are not available and their use generates a not available exception.
- The **Window** class **drawArc**, **drawChord**, **drawFloodFill**, and **drawPie** graphics methods are not available and their use generates a not available exception.
- Animated cursors are not available and their use generates a not available exception.
- Cursors other than the standard pointer and the busy (hourglass) cursor have no effect.
- Because the size of an image created using the **createPicture** method has a significant impact when using Compact JADE in thin client mode over a wireless network, the **Window** class **createPictureAsType** method enables you to convert the captured image to the specified type before returning it to the calling method.
- The **Binary**::**convertPicture** and **convertToFile** picture conversion methods are not available under Window CE 4.2 devices, and their use generates a *not available* exception.
- Minimizing a form is not available and is ignored.
- Help requests are ignored.
- The **Form** class **minimumHeight** and **minimumWidth** properties are ignored.

- Although docking controls can be floated, they can never be docked again because the required **mouseMove** events are not received.
- Use of hatch brushes during drawing is not supported. The fill is therefore always solid.
- Setting the **TextBox**::**maxLength** property is ignored, as this is not supported.
- **Table**::**partialTextIndication** is not available and is ignored.
- Because **mouseMove** events are never received, you cannot resize table rows and columns.
- Smart locale-based sorting of **ListBox** and **Table** controls occurs only if using a presentation client and the application server is a Windows environment.
- No **selected** event is generated for a submenu parent menu. The selected event is generated only for a top-level menu.
- The **Window**::**hwnd** method always returns zero (**0**).
- The **Form**::**scaleForm** property is always ignored.
- Standard Windows **Button** images are not available and are ignored.
- The hand-held device runs Unicode. In JADE thin client mode, a Unicode hand-held presentation client can connect to an ANSI application server. However, if non-ANSI characters are transmitted, unexpected failures may occur.
- **ListBox** images are not drawn transparently.
- Drawing lines are always solid, as other styles are not supported.
- The default font is Tahoma 8.25.

Note We recommend that your application uses only the Tahoma font, as that is usually the only font available.

- Although skins are supported, a form itself is never skinned (that is, there is no ability to draw the form borders). You should limit skin use, however, because they are likely to generate out-of-memory and resource issues.
- Maximizing a form adds the ☒ button to the hand-held top caption if the form has a **Close** button.
- Changing the **Form**::**maxButton** property value at run time requires the form to be hidden and then redisplayed, which is unattractive when redrawn.
- The hand-held device has a tendency to place a form at the top left of the available area. Positioning a non-maximized form does not always succeed.
- Form borders of resizable and fixed (that is, **BorderStyle_Sizable**, **BorderStyle_Double**, and **BorderStyle_None**) are always drawn as single; that is, you cannot resize a form, whose border is always created as **BorderStyle_Single**.
- **Folder** **TabsPosition_Left** and **TabsPosition_Right** values are treated as being at the top (that is, **TabsPosition_Top**).

- A form created under Compact JADE fills the work area of the Windows Mobile device, ignoring the **left**, **top**, **width**, and **height** property values of the form. This ensures that a form is visible in the device and that the screen work area is fully utilized.

  Although the Windows Mobile device does this automatically under some conditions if JADE does not do so, JADE always does so for consistency.

  In addition, if a form with a caption is dragged, the Windows Mobile device sometimes repositions the form to the top left work area. If required, your logic can reposition and resize a non-maximized form after it has been created.

In addition, the **TcpIpConnection** class **readUntil** and **readUntilAsynch** methods under Compact JADE raise exception 1068 (*Feature not available in this release*), because certain operating system features are not available on Windows Mobile devices.

See Appendix A of your JADE *Thin Client Guide* for details about:

- JADE Functionality Affected by Thin Client Operation
- JADE Thin Client Performance Considerations
- JADE Thin Client Restrictions

Chapter 6 Windows Vista Support

This chapter covers the differences when running JADE applications under the Microsoft Windows Vista.

Overview

When JADE is installed under the **Program File**s directory on Windows Vista, it conforms to the expected Windows Vista behavior.

The ability to return locations other than the JADE HOME directory for program and user data requests is dependent on where the JADE binaries have been installed.

If the JADE programs are not installed in the system-wide location for installed applications (in a folder in the **Program Files** directory), returning alternate addresses is not implemented. It is assumed that the programs installed in custom locations have sufficient security privileges to work in the style of earlier JADE 6.0 and JADE 6.1 releases.

Although the alternate names and locations are driven by Windows Vista requirements, the code is portable across Windows and UNIX-based implementations.

The **Node** and **Process** class methods summarized in this section provide access to various locations in the file system hierarchy, because security restrictions can deny appropriate access to previously available directories or the directories are a unique location for each user when they where previously shared. (For details about the differences of these methods called on nodes and processes, see "Directory Locations", in Chapter 5 of your JADE *Installation and Configuration Guide*.)

Method	Returns a string containing the …
getJadeInstallDirectory	Directory in which the executable of the current executing program is located.
getJadeHomeDirectory	Parent directory of the installation directory.

Method	Returns a string containing the …
getProgramDataDirectory	Program data directory, which is dependent on the value of the **ProgramDataDirectory** parameter in the [JadeEnvironment] section of the JADE initialization file and the location of the installation directory. Files that should be placed under the returned location are entities that should be shared across multiple users of these binaries; for example, the **jommsg.log** file or shared dictionary spelling files that are updated.
getUserDataDirectory	User data directory, which is dependent on the value of the **UserDataDirectory** parameter in the [JadeEnvironment] section of the JADE initialization file and the location of the installation directory. Files that should not be placed under the returned location are entities that should be shared across multiple users of these binaries; for example, if a presentation client installation occurs on a Windows machine running Citrix or Terminal Services and all users run the same thin client binaries, any data created on the client file system should be stored under this location (that is, dictionaries for each user).
getJadeWorkDirectory	Directory in which the JADE work files are created and which is dependent on the value of the **JadeWorkDirectory** parameter in the [JadeEnvironment] section of the JADE initialization file.

Returned directory strings are normalized; that is, they have the virgule, or forward slash, (/) character as directory separators and they end in the virgule (/) character. Directories are created if they do not exist.

For details about the JADE HOME directory, see "Relative Directory Handling in the Command Line path Parameter", earlier in this document, or "Specifying Parameters in the JADE Command Line", in Chapter 5 of the JADE *Installation and Configuration Guide*.

For more details, see "Directory Locations", in Chapter 5 of the JADE *Installation and Configuration Guide*, Volume 1 and Volume 2 of the JADE *Encyclopaedia of Classes*, and "JADE Environment [JadeEnvironment] Section", in the JADE *Initialization File Reference*.

JadeLocal Intra-Machine Transport Type

The **JadeLocal** intra-machine transport between the JADE database and application servers and standard clients is implemented by the use of shared memory and global events and semaphores on Windows Vista.

Vista does not allow objects to be created in the **Global** namespace without a specific privilege that is not available to a standard user.

The **JadeLocal** transport type uses the following configuration values in the respective [JadeServer] and [JadeClient] sections of the JADE initialization file.

```
NetworkSpecification<n>=JadeLocal,Enabled,basename

ServerNodeSpecifications=JadeLocal,basename
```

The **basename** value can have an optional **Local** or **Global** prefix. If the prefix is absent on Windows XP, it defaults to **Global**. On Vista, it defaults to **Local**, which is consistent with running with the least-privileges mode.

When running as a standard user on Vista, the value of the **basename** is created in the **Local** or session namespace, which means that all the JADE programs must be running as the same user logon and also in the same Windows session, to be able to connect to this RPC transport. For example, if the database is installed as a service, all application servers and standard clients wanting to connect to this database via **JadeLocal** transport must also be running as services and under the same user logon.

If you require the Windows XP behavior of **JadeLocal** on Windows Vista, change the following configuration parameters in the respective [JadeServer] and [JadeClient] sections of the JADE initialization file.

```
NetworkSpecification<n>=JadeLocal,Enabled,Global\basename

ServerNodeSpecifications=JadeLocal,Global\basename
```

In addition, your Windows administrator must configure the user logon to add the following Windows privilege **Create global objects** (**SeCreateGlobalPrivilege** at the programming API level), which can be done directly to the user logon or to a group of which that user is a member. On a machine that is not part of a Windows domain, this can be done by accessing the **Local Security Policy** in the Administrative tools directory and adding the **Create global objects** policy under **Local Policies** / **User Rights Assignment** to the desired group or user.

These changes allow JADE programs that need to connect via the **JadeLocal** intra-machine transport to work across multiple Windows sessions or user logons.

If a standard user attempts to create a **Global\basename** value in either of these initialization file parameters, it fails because of insufficient privileges.

Most Recently Used (MRU) List of Opened Database Paths

Because Windows Vista does not allow standard users to write to the **HKEY_LOCAL_MACHINE** (HKLM) registry hive in a global area, the path to the system directory is recorded in the **HKEY_CURRENT_USER** (HKCU) registry hive but it is limited to the database paths that are accessed by the current user logon.

Separation of Read-Only and Read-Write Files

To support the separation of read-only and read-write files at run time, the **jade.exe**, **jadepg**, and **jaddinst.exe** programs do not write any files into the JADE installation directory. (This affects standard clients and thin clients.)

The **JadeWorkDirectory** parameter in the [JadeEnvironment] section of the JADE initialization file defines the location of a directory that JADE uses for work files. This directory, which is required for internal use such as interlock files used during downloading, defaults to ***<jade-program-data-directory>*\temp** on Windows and ***<jade-program-data-directory>*\tmp** on Linux.

In addition, the thin client cache file is written into this directory if the location has not been specified by the **FormCacheFile** parameter in the [JadeThinClient] section of the JADE initialization file. The **download.log** file created by the download install process is output to the log directory specified in the **LogDirectory** parameter in the [JadeLog] section of the JADE initialization file.

Note To ensure total read-only and read-write file separation, the JADE initialization file cannot be located in the JADE HOME directory.

Service Management

Because standard Vista and Windows XP users do not have the necessary privileges to modify entries in the **HKEY_LOCAL_MACHINE** (HKLM) area of the registry when services are installed or removed, the menu options that enable the JADE Remote Node Access utility (**jadrap**) and a JADE application server (**jadapp**) to run as a service are disabled if you do not have the necessary privileges to install or remove an application as a service.

When a JADE application server has been installed as a service, it displays the correct state information but does not allow standard users to start, stop, or remove the service.

Vista Security Model

Because of the changes to the security model in Windows Vista, if you are upgrading to JADE under Vista, ensure that you have the appropriate privileges or capabilities to install applications.

The configuration of Vista's User Account Control (UAC) and your current user account privileges may affect the behavior of the upgrade to JADE 6.3. For details about Windows Vista UACs, standard user accounts, and administrator accounts, see:

http://technet2.microsoft.com/WindowsVista/en/library/00d04415-2b2f-422c-b70e-b18ff918c2811033.mspx?mfr=true

Index

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W