J2ME CLDC KJava/Palm "Add-On" Package

Release Notes / CLDC 1.0.2 FCS

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

CLDC 1.0.2 FCS
February 26, 2001

901 San Antonio Road, Palo Alto, CA 94303 USA

SUN MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. SUN SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THE PUBLICATION. SUN MICROSYSTEMS, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME.

Contents

1

Introduction

These release notes provide information about Sun's Palm implementation of the *Connected, Limited Device Configuration* (CLDC). The CLDC implementation runs on top of Sun's K Virtual Machine (KVM) that is provided as part of this release.

About This Release

This release package contains a supplementary, "add-on" release that is intended to complement the *J2ME CLDC Reference Implementation* release package. The *J2ME CLDC Reference Implementation* package is available separately, and it runs on Linux, Solaris and Win32 platforms.

This "add-on" package contains a Palm port of the K Virtual Machine, the associated Palm tools, as well as the Java library classes and native code for supporting the old `com.sun.kjava (Spotlet)` GUI libraries that are not officially supported by Sun. The official GUI libraries for Java 2 Micro Edition are currently being defined through Java Community Process (JCP).

Note – This release package is not a complete, standalone package. Even though the source code for the Palm-specific parts of the K Virtual Machine is provided as part of this release, you cannot build the Palm CLDC/KVM binaries from the source files provided in this package alone. If you want to build the Palm binaries from source code, please install this package on top of the *J2ME CLDC Reference Implementation*.

About CLDC

CLDC is the result of a Java Community Process effort (JSR-30) whose goal was to define a standard, portable Java™ platform for small, resource-constrained, connected devices. The CLDC specification effort was completed in collaboration with 18 companies representing different industries. Target devices for CLDC are characterized generally as follows:

- 160 to 512 kilobytes of total memory, including both RAM and flash or ROM, available for the Java platform.
- Limited power, often battery powered operation.
- Connectivity to some kind of network, often with a wireless, intermittent connection and with limited (often 9600 bps or less) bandwidth.
- User interfaces with varying degrees of sophistication down to and including none.

Cell phones, two-way pagers, personal digital assistants (PDAs), organizers, home appliances, and point of sale terminals are some, but not all, of the devices that might be supported by CLDC.

The *CLDC Specification* document is available for public downloading at `http://java.sun.com/aboutJava/communityprocess/final/jsr030/index.html`.

Note that CLDC is intended to serve as the "lowest common denominator" building block for various kinds of resource-constrained, Java Powered™ devices. As such, CLDC is not a complete, self-sufficient solution, but it needs to be complemented by *profiles*. For instance, all the user interface aspects are outside the scope of *CLDC Specification*. A parallel Java Community Process effort (JSR-37) called *Mobile Information Device Profile* (MIDP) has defined the necessary remaining Java platform features and libraries for two-way communication devices such as cell phones, while another effort (JSR-75) is currently focusing on PDA-type devices. Other profiles for other vertical markets or device categories may be defined later.

Supported Platforms

The *J2ME CLDC Reference Implementation* (available separately) runs on Solaris and Win32 platforms. A CLDC-compliant port for Linux is also available.

A CLDC-compliant port for the Palm Connected Organizer is provided as part of this release package. This release package can be installed on top of *J2ME CLDC Reference Implementation*. The Palm port is not considered to be a reference implementation of the CLDC.

Prerequisites and Dependencies

For more details on the Connected, Limited Device Configuration standardization effort, please refer to the *Connected, Limited Device Configuration Specification*, version 1.0, Sun Microsystems, Inc. (`http://java.sun.com/aboutJava/communityprocess/final/jsr030/index.html`).

Please refer to the *KVM Porting Guide*, Sun Microsystems, Inc. for information about porting the K Virtual Machine to new platforms.

2

Installation Notes

Unzipping the Distribution

Note – Please keep in mind that this release package in not a standalone release. It is intended that you install this package on top of the *J2ME CLDC Reference Implementation* package. The *J2ME CLDC Reference Implementation* package is available separately.

Unzip (using the *overwrite* option) the distribution into the same directory (`j2me_cldc`) in which you have previously unzipped the *J2ME CLDC Reference Implementation*. This will create the directory `j2me_cldc` with the following subdirectories:

- `api`
- `bin`
- `build`
- `docs`
- `jam`
- `kvm`
- `samples`
- `tools`

IMPORTANT: Make sure that the unzipped files go into the same directories as the files that you extracted earlier from the *J2ME CLDC Reference Implementation* package*!!*

Please refer to the *KVM Porting Guide*, Sun Microsystems, Inc. for further information on the contents of the directories listed above. The porting guide is provided as part the *J2ME CLDC Reference Implementation* package.

Building the Source Release

Refer to instructions provided in the release notes of the *J2ME CLDC Reference Implementation* package on how to build the source code release. In order to build the Palm version of CLDC/KVM, a Metrowerks CodeWarrior project file is provided.

Note that for building the “romized” version of the Palm KVM with all the system classes prelinked/preloaded into the KVM executable (this is the default build mechanism for CLDC 1.0.2 FCS), you need to run the JavaCodeCompact (JCC) tool on a desktop machine first. Enter the subdirectory `tools/jcc` and type `gnumake palm` before compiling the source files using Metrowerks CodeWarrior.

Running Sample Applications

The directory `samples` contains several demonstration programs that can be run on all target platforms.

The Palm-like GUI

The Connected, Limited Device Configuration itself does not define any GUI (graphical user interface) functionality. For testing purposes, this release includes a number of GUI classes that were originally written for the Palm implementation of KVM. These classes can be found in directory `api/classes/com/sun/kjava`.

Note – The GUI classes provided in package `com.sun.kjava` are *NOT* part of the Connected Limited Device Configuration (CLDC). Official GUI classes for Java 2 Micro Edition will be defined separately through the Java Community Process and included in *J2ME profiles*.

Running the demo apps on Linux, Solaris and Win32

The sample applications in directory `samples` utilize the `com.sun.kjava` GUI classes that were originally written for the Palm version of KVM. The sample applications can be run on the Unix (Solaris and Linux) and Win32 versions of KVM, because these versions contain the necessary native methods to provide a simulated "Palm-like" GUI environment. This GUI has the following features:

- Monochrome (black/white) input/output screen of 160 x 160 pixels with mouse support. When the mouse cursor is within the screen area, it behaves as the stylus (pen) on a Palm device. Clicking the left mouse button corresponds to tapping the screen with the pen. Moving the mouse while holding down the left mouse button corresponds to dragging the pen across the screen.
- Dummy silk-screened area. This is the large rectangle directly below the screen, which contains a smaller rectangle and two circles on either side. This area is NOT supported in the Unix and Windows versions of KVM. It corresponding to the silk-screened area of a Palm device, containing the Graffiti are and the Home/ Applications, Menu, Calculator, and Find buttons.
- Simulated "hard" buttons. Clicking the mouse on one of these rectangles or circles corresponds to pressing a "hard" button on a Palm device. These buttons ARE supported in the Unix and Windows versions of KVM. Left to right, they are: Power button, Calendar button. Address Book button, (upper) Scroll Up button, (lower) Scroll Down button, To Do List button, and Memo Pad button.

  Note: Most of the sample programs respond only to the Scrolling buttons. The Missiles game responds to all the "hard" buttons, but they are used to control the game (not to invoke the applications listed above).

  Note: Unlike in a real Palm device, holding down a hard button does NOT send repeated button events. You must repeatedly tap the button to produce multiple events.
- Simulated Graffiti input. Typing a key on the keyboard corresponds to entering that character into the Graffiti area on a Palm device.

  Note: Many of the sample programs do not respond to character input.

For convenience, two script files are provided in the samples directory for running the sample programs on Unix and Windows. "ku" is a Unix shell script, and "kw.bat" is a Windows batch file. Depending on the platform, the sample programs can be executed by typing either:

```
ku <classname>
```

or

```
kw <classname>
```

When the program quits, the KVM stops and the GUI screen disappears.

The following section lists each of the nine sample programs using an example of the command to start the program. A few notes on the use of each program are also provided.

"ku dragon.Dragon"

Tap anywhere on the screen to draw a dragon fractal at that point on the screen. Use the "+" and "-" buttons to increase/decrease the number of iterations or segment size of the next dragon to be drawn. Use the "?" to select a random number (within the allowable range for that control).

"kw manyballs.ManyBalls"

The program begins with a single ball bouncing around the screen (driven by its own Java thread within KVM). Press the "+" and "-" to add or decrease the number of ball threads.

"ku pong.Pong" and "ku starCruiser.StarCruiser"

Use the scroll bar slider, the scroll arrows, or the "hard" Scroll buttons to view all the game instructions. Press "Done" to go to the game screen.

"kw scribble.Scribble"

Read the game instructions and press "Done" to go to the game screen. This program responds to character input by displaying the character on the screen. Use the "hard" Scroll buttons to change the size of the ball. You can also "drag" the ball to another place on the screen.

"kw threeDLogo.ThreeDLogo"

Drag the pen to cause the 3D figure to rotate. Tap anywhere on the screen to return the figure to its starting position. The "Beam" button is NOT supported.

"ku ui.UITest"

A simple test program for some of the old UI classes.

"kw dots.DotGame"

Tap the "Help" button for instructions. Tap one of the four class names to select that algorithm for either the Host or the Guest player. Use PenTaps for a human player. Use one of the other 3 classes to play against that program. Either or both of Host and Guest can be human or program. Tap one of the "moves first" buttons to start the game. For the most compute-intensive game, play Average against Average on a grid of 11 rows x 13 columns.

"ku missiles.Missiles"

Tap the "Help" button for instructions. You don't need to gather any classes because they are all provided (in samples/missiles/*). Sound is not supported on the Unix and Windows versions. The Unix version displays the bitmaps somewhat incorrectly (green icons in a white box) but the game is still playable.

Running the demo apps on the Palm

If you have installed the additional Palm release package on top of this release, you can run the sample applications on a real Palm device. Demo applications are provided as ‘prc’ (Palm executable) files. Once you have installed the ‘KVM.prc’ and ‘KVMutil.prc’ executables on your Palm device, simply install these application files on your Palm device, and launch them from the Palm application launcher.

Note – This release package includes two different KVM executables for the Palm: ‘KVM.prc’ and ‘KVM_g.prc’. The difference between these two executables is that the latter executable contains the Java debugging facilities that are needed for plugging the KVM into a third party debugging environment. If you do not intend to use the debugging facilities, it is recommended that you install ‘KVM.prc’, because that executable is smaller and runs Java programs slightly faster.

IMPORTANT: Since ‘KVM.prc’ and ‘KVM_g.prc’ use the same Palm application creator ID, these executables cannot reside in a Palm device at the same time. To install ‘KVM_g.prc’ into your Palm device, you need to remove ‘KVM.prc’ first. Similarly, if you want to install ‘KVM.prc’, you need to make sure that ‘KVM_g.prc’ has been deleted first.

Building Sample Applications from Source Code

Application development for CLDC usually takes place on a desktop computer. At the high level, the procedure for building Java applications for CLDC is as follows:

1. compile the Java application using a Java compiler (not provided as part of this release.)
2. preverify the Java classfiles with the `preverify` tool provided in the *J2ME CLDC Reference Implementation* package.
3. use the `jar` tool to create a JAR file that contains all the Java classes belonging to your application (the `jar` tool is not provided as part of this release.)
4. finally, if you intend to run your application on the Palm platform, use the `MakePalmApp` tool to convert the Java classfiles or a JAR file into a 'prc'. This step is not necessary on other platforms.

Sample command line operations:

- Compilation:

```
javac -g:none -d tmp
 -classpath tmp;../bin/common/api/classes;
              ../bin/kjava/api/classes
 -bootclasspath ../bin/common/api/classes
 src/pong/Pong.java src/pong/PongBall.java
```

- Preverification:

```
../bin/win32/preverify -d classes -classpath
 ../bin/common/api/classes;../bin/kjava/api/classes tmp
```

- JAR creation:

```
cd classes
jar cvf ../Pong.jar pong
cd ..
```

- Building a Palm executable:

```
java
-classpath ../bin/kjava/tools/palm/classes
 palm.database.MakePalmApp
 -v -version "1.1" -icon icons/pong.bmp
 -bootclasspath ../bin/common/api/classes
 -classpath Pong.jar;../bin/kjava/api/classes
 pong.Pong
```

Refer to `docs/tools.html` for further instructions on how to build the sample applications.

3

Quality Assurance

Testing

The QA tests and CLDC TCK compatibility tests have been run on a regular basis on emulators and on the following platforms:

- Palm IIIx
- Palm V, Palm Vx
- Palm VII, VIIx

The Palm implementation passes all the test cases included in CLDC TCK. The TCK compatibility toolkit performs comprehensive regression testing of various Java language, virtual machine and library features supported by CLDC. The total number of test cases in CLDC TCK is approximately 4,500.

Components that are outside the scope of CLDC (such as packages `com.sun.cldc.io` and `com.sun.kjava`) have not undergone similar regression tests. Various demo applications have been used for testing those components.

Note – The Palm release includes support for wireless HTTP access for Palm VII. Wireless HTTP support is known to work under PalmOS version 3.2 on Palm VII.
No comprehensive testing of wireless networking has been performed on PalmOS 3.5 / Palm VIIx.

Known Bugs

A number of bugs have been dispatched for re-engineering but remain open at the time of this release. For a definitive reference on open bugs and feature requests, log in to the Java Developer Connection (JDC) web site: http://developer.java.sun.com/developer/.

A detailed list of bugs and feature requests related to the K Virtual Machine and CLDC can be found in: http://developer.java.sun.com/developer/bugParade/index.jshtml, under the bug category "K Virtual Machine".