

# J2ME CLDC/KVM Binary Release

---

*Release Notes / Beta Release*



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

Beta 2  
March 24, 2000

Copyright © 2000 Sun Microsystems, Inc.

901 San Antonio Road, Palo Alto, CA 94303 USA

All rights reserved. Copyright in this document is owned by Sun Microsystems, Inc.

Sun Microsystems, Inc. (SUN) hereby grants to you at no charge a nonexclusive, nontransferable, worldwide, limited license (without the right to sublicense) under SUN's intellectual property rights that are essential to practice the J2ME CLDC Reference Implementation technology to use this document for internal evaluation purposes only. Other than this limited license, you acquire no right, title, or interest in or to the document and you shall have no right to use the document for productive or commercial use.

#### RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-1(a).

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.

#### TRADEMARKS

Sun, Sun Microsystems, the Sun logo, Java, Java Card, SunDocs, and SunExpress are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. UNIX® is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

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 1**

Items Included in the Release 2

Prerequisites and Dependencies 2

## **2. Installation Notes 3**

Unzipping the Distribution 3

Running and Building Sample Applications on Palm 3

## **3. Quality Assurance 5**

Testing 5

Bugs Fixed in this Release 5

Known Bugs 12



# Introduction

---

These release notes provide information about Sun's reference implementation of the Connected, Limited Device Configuration (CLDC) and KVM.

CLDC is the result of a Java Community Process effort (JSR-30) whose goal is to define a standard Java platform for small, resource-constrained, connected devices. The CLDC specification effort has been done in collaboration with 19 companies representing different industries. 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 reference implementation runs on Sun's K Virtual Machine (KVM) implementation that is provided as part of this release.

Note that CLDC is intended to serve as the lowest common denominator for various kinds of resource-constrained, Java-enabled 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) is currently underway to define the necessary remaining Java platform features and libraries for a specific vertical market/device category. Other profiles for other vertical markets or device categories may be defined later.

The CLDC reference implementation and KVM run on Windows 98/NT, Solaris and Palm platforms. The version provided with this release contains binaries for the Palm platform only. Full version including the source code and binaries for all three platforms is available separately from Sun.

---

**Note** – The Java Community Process effort JSR-30 defining the Connected, Limited Device Configuration has not been completed yet, and therefore the features provided in this reference implementation are still subject to change.

---

---

## Items Included in the Release

This release includes:

- K Virtual Machine (KVM) for the Palm
- Preverifier tool
- JavaCodeCompact tool (for class prelinking/preloading)
- CLDC Java libraries (API)
- Additional Java libraries provided for testing purposes only (GUI)
- Sample applications provided for testing purposes

The release includes the following documentation:

- CLDC API documentation
- Palm-specific GUI API documentation (not part of CLDC)
- Overview of application development for the Palm
- *J2ME CLDC/KVM Binary Release Notes* (this document)

The CLDC Specification document, *Connected Limited Device Configuration Specification*, is available separately from the Java Community Process web site (<http://java.sun.com/aboutjava/communityprocess/review.html>).

---

**Note** – This release contains user interface and other libraries that are not officially part of the CLDC reference implementation. These libraries have been provided for testing purposes only, and may change or may be removed in future releases.

---

---

## Prerequisites and Dependencies

For more details on the Connected, Limited Device Configuration, please refer to the *Connected, Limited Device Configuration Specification*, Sun Microsystems, Inc.

Information about Java 2 Micro Edition (J2ME) configurations and profiles is provided in *Configurations and Profiles Architecture Specification, Java™ 2 Platform Micro Edition (J2ME)*, Sun Microsystems, Inc.

## Installation Notes

---

---

### Unzipping the Distribution

Unzip the distribution into any directory of your choice. It creates the directory `j2me_cldc` with the following subdirectories:

- `bin`
- `doc`
- `lib`
- `src`

---

### Running and Building Sample Applications on Palm

A number of sample application are provided for demonstration purposes. These applications are provided as 'prc' (Palm executable) files. Simply install these files on your Palm device, and launch them from the Palm application launcher. The KVM executable (KVM.prc) and KVM utility tool (KVMutil.prc) must be installed this way, too.

Source code for the sample applications is provided with this release. Refer to `doc/tools.html` for instructions on how to build the sample applications.

In general, application development takes place on your favorite desktop computer. At the high level, the procedure for building Java applications for the Palm is as follows:

- 1) compile the Java application using a Java compiler (not provided in this release)
- 2) preverify the Java classfiles with the `preverify` tool provided in this release
- 3) use the `MakePalmApp` tool to convert the Java classfiles into a 'prc' file

Sample command line operations:

- Compilation:

```
javac -g:none -d tmp -classpath tmp:../lib/classes  
-bootclasspath ../lib/classes src/Pong.java src/PongBall.java
```

- Preverification:

```
../bin/preverify -d classes -classpath ../lib/classes tmp
```

- Building a Palm executable:

```
java -classpath ../lib/classes palm.database.MakePalmApp -v  
-version "0.92" -icon icons/pong.bmp -bootclasspath  
../lib/classes -classpath classes Pong
```



## Quality Assurance

---

---

### Testing

JCK compatibility tests version 1.3, Tonga regression and stress tests and KVM Beta 2 VM and language compatibility tests have been run on a regular basis on emulators and on the following platforms:

- Palm IIIx
- Palm V

Various demo applications have been used for testing purposes.

---

### Bugs Fixed in this Release

Bug id numbers below refer to bug numbers in the BugTraq bug tracking database. This database is available to the public through the Java Developer Connection website (<http://developer.java.sun.com/developer/>).

**TABLE 3-1**

Bug Id	Synopsis
4297561	Better documentation needed on various aspects of <code>com.sun.kjava.*</code> classes.
4297513	Two open Sockets causes VM crash.
4302232	Palm: <Invalid format string: "%c">
4303012	<code>java.io.PrintStream</code> does not implement <code>flush()</code>

**TABLE 3-1**

Bug Id	Synopsis
4303008	Incorrect exception in java.util.Vector with negative init. size
4308017	private skip buffer in java.io.InputStream is too big
4308440	String constructors throw wrong exception for empty encoding
4309240	unexpected IllegalArgumentException in Hashtable(int)
4308432	String constructors throw wrong exception for null encoding
4309631	java.io.PrintStream must extend FilterOutputStream
4309956	System properties have incorrect default values
4309998	arraycopy method does not throw IndexOutOfBoundsException when it should
4310432	java.lang.Thread.toString() works incorrectly
4312327	Attempt to convert unknown character to bytes results in Exception
4316556	no IOException thrown when reading from closed InputStreamReader
4314772	Unexpected RuntimeException in Class.getResourceAsStream(String)
4313523	The Solaris build is broken: StrCompare in networkPrim.o
4323239	palm.database.MakePalmApp skips classes
4296919	adding standard Java classes is declared but not specified.
4300635	definition of KVM verification must be clarified
4305541	setDaemon, isDaemon omitted from CLDC Core
4309307	incomplete description of fields in javax.microedition.io.Connector
4309314	the size argument is not described for newDatagram(..) methods
4309304	Class.getResourceAsStream
4309595	The getProperty0(String key) method of java.lang.System is non-relevant
4309722	some exception should be thrown on timeouted receive call
4309725	should specify thrown exceptions if the length+offset is out of byte[].length
4310477	java.io.PrintStream semantics doesn't match those of the J2SE class
4310793	excessive description for javax.microedition.io.Connector
4310850	non-public method in the Connection interface
4312357	ConnectionNotFoundException should be thrown when open fails
4291538	kvm gets SEGV at isAssignable()
4291832	kvm gets SEGV at Interpret()

**TABLE 3-1**

Bug Id	Synopsis
4291827	kvm gets SEGV at ImplementsInterface()
4291840	kvm gets SEGV at Interpret()/PopStack
4291869	kvm gets SEGV in printVMstatus()
4292747	kvm hangs in MARK_AND_TAIL_RECURSE(subobject)
4293329	kvm gets SEGV at AlertUser
4293793	kvm hangs at SwitchThread
4293802	kvm gets BUS error in countClasses()
4293806	kvm gets BUS error in isAssignableTo()
4293714	kvm state is corrupted if args are present on command line
4294295	kvm gets SEGV at change_Key_to_Name()
4294315	kvm gets 'Heap is corrupted'
4294316	kvm gets SEGV at pushFrame()
4294333	kvm gets 'Uncaught java/lang/Error'
4294402	DR5: arraycopy is not working
4293799	kvm hangs at reschedule()
4294306	kvm gets 'Out of heap memory'
4294717	kvm gets 'The exception Integer overflow' on Win32
4295034	VM seems to crash on Socket or Database use with random errors
4293810	kvm gets SEGV at getParameterSlotsPointerMask()
4295472	Kvm instantiates java.lang.String with UTF8 characters
4296144	Error message "ALERT:Locals pointers corrupted"
4298252	kvm fails to handle some exceptions correctly
4297536	RFE: Avoid loaded classes using heap
4298549	kvm hangs at printFrame
4298758	kvm does not throw IllegalStateException
4298134	incorrect handling of exceptions
4299020	KVM memory errors or other problems in Database use?
4294301	kvm gets 'Bad call to notifyAll'
4299289	incorrect handling of exceptions

**TABLE 3-1**

Bug Id	Synopsis
4299485	Class.newInstance fails if the target class is not a public class
4294976	Globals need to be zeroed out after VM finishes running a class
4299560	The ArrayStoreException on an assignment operation to an array element
4299571	System.exit(0) returns 1
4299928	SEGV at line 3157 in Interpret()
4299934	The IllegalAccessException when create new instance of empty class
4299937	kvm does not throw ClassNotFoundException for invalid classnames.
4300318	Uncaught java/lang/IllegalMonitorStateException
4300329	kvm does not throw InstantiationException
4300447	kvm gets BUS error in printRegisterStatus at line 346 (interpreter.c)
4293825	kvm gets 'Bad call to monitorExit'
4293828	kvm gets 'bad stack pointer'
4300494	kvm does not throw Illegal*Exception
4300656	ROMIZER: SEGV in mallocObject at line 1124 in file "garbage.c"
4300657	ROMIZER: SEGV in markROMHashtable at line 263
4300451	kvm gets BUS error in getClassName_inBuffer
4300668	kvm gets 'Uncaught java/lang/NullPointerException'
4300933	SEGV in change_Key_to_MethodSignature_inBuffer at line 470
4300942	KVM does not work properly with array dimensions
4301222	SEGV in instantiateMultiArray at line 1252
4300925	SEGV in Interpret() at line 589 in file "interpret.c"
4293818	kvm gets SEGV at isAssignableTo()
4293816	kvm gets SEGV at fatalSlotError()
4299566	Uncaught java/lang/NullPointerException in a try-finally block
4301381	"bad
4301528	WIN32: "The memory could not be read", Interpret() line 2269
4301529	WIN32: Integer overflow, Interpret() line 1462
4301538	WIN32: 'bad stack pointer' or SEGV in printStackTrace() line 273
4301557	Incorrect work with a bytecode name of an externally-visible class

**TABLE 3-1**

Bug Id	Synopsis
4301558	The explicit creation of an array by an array creation expression failed
4301570	The test of a type of a hidden field failed.
4302230	Palm: "KVM" 0.5 reports "Field.c, Line;5198, Deleting dm handle"
4301561	The JVM Spec violation in 'daload'
4302958	Palm: the test arrays00401 gets "Fatal error"
4303003	KVM gets SEGV at null string argument
4303358	const declaration require for static variable initialization
4303360	Error handling
4303362	kVM function naming
4305141	printProfileInfo() fails when ROMIZER is enabled
4305143	ROMIZED data not properly re-initialized when KVM is restarted
4302053	The wrong initialization procedure with threads
4305555	preverifier gets 'Class not found' on a corrupted class, not ClassFormatError
4305556	preverifier gets SEGV in push_block at line 117 in file "inlinejsr.c"
4305557	preverifier throws "*** panic: exception table overflow"
4306494	minimize the use of recursive algorithm in vm code
4302960	Palm: the test cls16602 gets "Bad dynamic heap objects"
4302961	Palm: "Bad bit set"
4306561	preverifier always gets java/lang/VerifyError
4306784	small bug in frame.c (debugging output)
4307929	Preverifier: breaks exception table for adjoining try-catch blocks
4308219	loaderFile.c crashes if class is not found.
4306563	KVM throws VerifyError with multi-dim. (4 and more) arrays
4306564	incorrect handling of nested try-finally blocks
4306565	KVM skips the finally block
4306562	KVM gets SEGV in getRawClass at line 425 in file "class.c"
4301533	WIN32: "The memory could not be read", Interpret() line 2182
4308482	kvm gets VerifyError instead of NullPointerException
4308920	JAR: KVM gets SIGBUS at not existed class in JAR_ReadBytes

**TABLE 3-1**

Bug Id	Synopsis
4308430	String.startsWith(null) may cause SIGSEGV
4309634	JAR: kvm reads CLASSPATH up to the first entry of a jar-file
4309665	Preverifier: SEGV at line 283 in file "check_class.c"
4309639	correct class file causes KVM pre-verifier crash
4309641	Preverifier: corrupts class-files (Invalid constant pool entry)
4309988	Preverifier: corrupts classes with 'Bad class flag'
4309989	Preverifier: corrupts classes with 'bad method flags'
4309995	Preverify: 'Verify error' on previously passed tests
4310433	kvm gets SEGV in DismantleThread
4310782	arraycopy works incorrectly with arrays of different type elements
4310866	OutOfMemoryError is not caught when the heap is full
4310790	sleep(long) does not throw IllegalArgumentException for negative input
4310847	loop filling Vector with Strings crashes kvm
4311147	when heap is full and expecting a StackOverflowError, the kvm crashes
4311173	Object.wait(long) does not throw IllegalArgumentException for negative input
4311558	inserting objects to HashTable crashes kvm
4311717	Class.getResourceAsStream should return null if resource not found
4311968	KVM retrieves Java types long and double incorrectly with ROMIZING=true
4312253	notifyAll() crashes kvm when not in synchronized block
4310446	KVM pre-verifier can not handle multi-try statement
4310637	KVM calls fatalError if instantiateArray fails, should throw OOM instead.
4312604	circular dependency in Connector and System classes
4312807	KVM DR5.0- Windows (Bitmap class) bitmap image mangled
4312990	problem with the initial value of static final fields
4306998	SEGV in Interpret at line 2882 in file "interpret.c"
4313381	Preverifier: SEGV line 213 in "inlinejsr.c"
4313510	DR5 JavaCodeCompact generates C code which causes compile time errors
4313665	Optimization: AllCode in ROMjava.c should be marked as const
4313801	Optimization: nativeFunctionTables not needed in ROMIZER build

**TABLE 3-1**

Bug Id	Synopsis
4314097	JAR: unable to load classes from <xxx.jar> directory
4314149	Preverifier: Should not "panic"
4314223	verifier does not detect non-preverified "extending class"
4315209	Class.isInstance(null) call leads to SIGSEGV signal received
4314738	Class.forName(String) causes to out of heap memory if "[" is passed
4316407	Preverifier: corrupts class-files (multianewarray008)
4316517	socket protocol is absent in Win32 build b
4316399	unable to read classes from jar-files.
4316566	Class.isAssignableFrom(null) causes SIGSEGV received
4297527	instanceof is flakey
4297524	Multinewarray JVM opcode crash
4297526	Null pointers not always checked
4317198	unaligned long64 access in interpreter.c (Solaris + GCC)
4318541	Thread.start() doesn't throw IllegalThreadStateException
4319322	KVM hangup if any of the superclasses of C is C itself
4320481	Palm: javasoft.sqe.tests.vm.concepts.interfaces018.interfaces01801 fails
4320472	Palm: the first attempt to run application fails with "Not enough memory ..."
4320493	Palm:"Fatal exception" if try to input chars to the TextField via "keyboard"
4320741	Palm: javasoft/sqe/tests/vm/classfmt/atrcv1001/atrcv100101m1 failed
4321772	Palm: System.arraycopy does not work properly
4322753	KVM gets Marking object not in heap space at correct class
4322839	KVM Tool JCC outputs wrong code
4314010	SEGV in instantiateMultiArray at line 1267 in file "class.c"

---

## Known Bugs

The following 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, refer to the Java Developer Connection web site (<http://developer.java.sun.com/developer/>).

**TABLE 3-2**

Bug Id	Synopsis
4294892	System.currentTimeMillis() is incorrect after Palm powerdown
4300669	MakePalmApp with jdk1.1.8 causes panic: popSeen: corrupt seen class stack
4323666	Palm: VerticalScrollBar paint/restore bug
4302959	Palm: "no float remainder"
4320483	Palm: "Fatal exception" when try to run 'missiles.Missiles' demo
4323412	Palm: Memory leak: "KVM has just read directly from an unallocated chunk ..."
4323656	Palm: tests for 'Class name access' hangs on PalmOS 3.0
4323809	Palm: Beaming does not always work on PalmOS 3.5
4323816	Palm: makePalmApp -networking option kills apps when net not used