Nokia 9210 Communicator Java White Paper

TABLE OF CONTENTS

1. PREFACE

This document gives a short introduction to Java as a technology and then goes through the Nokia 9210 Communicator Java implementation.

2. INTRODUCTION TO JAVA

The term Java applies to two related but distinct concepts: it describes a programming language and a runtime environment in which programs written in the Java language can be executed. The Java language, from Sun Microsystems, Inc., is an object-oriented language with a syntax similar to C++, so C++ programmers can pick it up quite easily. Java programming language omits some C++ features making the language more effective and less susceptible to programmer errors. Unlike C++, Java source files are compiled for byte code (not object or machine code that can only be executed by a specific CPU, or run on a specific operating system). Byte code can be interpreted on any computer that has a Java Virtual Machine (JVM). The Java platform is designed to provide this "Write Once, Run Anywhere" capability.
The first and most widely known type of Java programs are applets, which run in the context of a web browser. Applets cannot be run without a browser or appletviewer application and applets cannot access local resources on the device. The second type of Java programs are standalone applications. Standalone applications perform like any other applications, except that Java Virtual Machine is needed for their execution. The third category of Java programs are the server applications, that do not run on terminals but on the server-side.

The Java Virtual Machine forms part of a Java runtime environment, which provides a self-contained execution environment for Java byte code. This code is independent of the operating system of the computer that it runs on. The runtime environment comprises:

- class loaders that load the byte codes from .class, .jar or .zip files
- the JVM interpreter that processes the byte codes to execute them
- class libraries that contain implementations of APIs that the Java applications can use
- application frameworks, including event handling, windowing and graphics support.

Security measures are an integral part of Java's design. The goal is to provide a secure, ready-built platform on which Java-enabled applications can be run in a secure fashion. There must also be security tools and services implemented in the Java programming language that enable a wider range of security-sensitive applications.

The Java platform has a large developer community. The fifth anniversary of the Java platform was held at the JavaOne 2000 developer conference. The conference had approximately 25,000 attendees, solidifying the JavaOne conference's title as the world's largest developer conference.

As a summary, the key benefits of Java language are:
- portability and platform independence
- easy and effective to develop
- large developing community
- safety

2.1 Java 1.0 and 1.1 Platform

Java is available in several different forms, described below. A Java Application Environment (JAE) comprises a Java Virtual Machine, APIs and support files, whereas a Java Development Kit (JDK) comprises an application environment, plus development tools.

Enterprise Java is available in several versions: 1.0.2, 1.1 and 1.2 (or Java 2). Sun Microsystems, Inc. provides a JAE and JDK for each of these. The different versions of JDK 1.1 running up to JDK 1.1.8 are compatible with one another: JDK 1.1 and JDK 1.2 are only backwards compatible, i.e. if a programmer uses features new to Java 2, the application will not run on JDK1.1. Some of these new features in Java 2 are also available as optional packages to JDK 1.1.

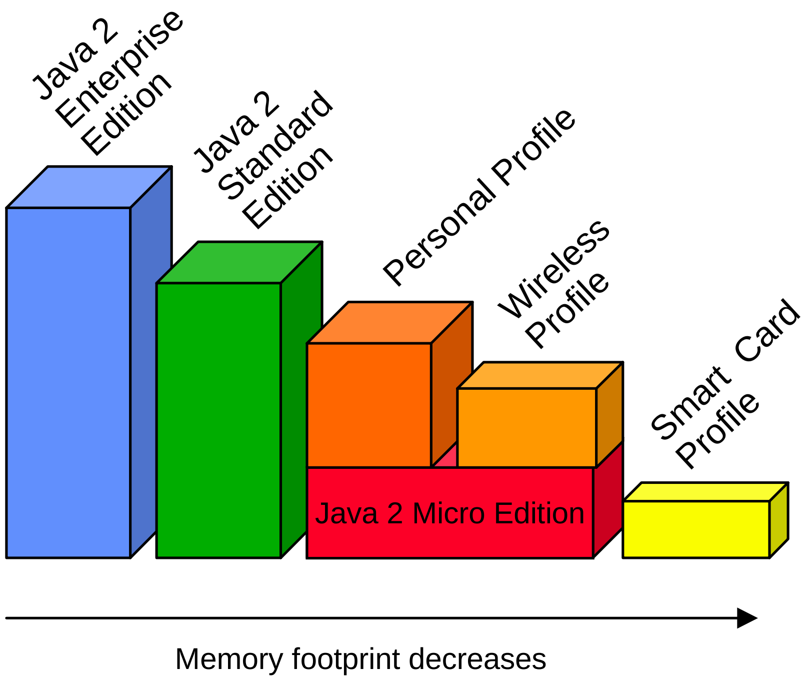
Personal Java is an application environment. It is distinguished from JDK 1.1 in that it includes only a subset of the JDK 1.1 APIs. The subset includes Abstract Windowing Toolkit (AWT) and other classes that allow applets and applications to be supported. The Personal Java Virtual Machine adheres to the same specification as the Enterprise Java Virtual Machine, although its implementation is tuned for resource-constrained devices. This is the Java version implemented in the Nokia 9210 Communicator.

Embedded Java is an application environment that is intended for devices having dedicated functionality and limitations on resources such as memory. The embedded Java APIs do not include support for applets. The intention is not to use Embedded Java as a platform for downloadable applications. Embedded Java is intended for cases where part of the functionality of the device is implemented in Java and the program to be run combined with the JVM and standard libraries is provided as a ROM image.

Java Card specifications enable Java technology to run on smart cards and other devices with limited memory. The Java Card API allows applications written for one smart card platform that is enabled with Java Card technology to run on any other such platform. The Java Card Application Environment (JCAE) is licensed on an OEM-basis to smart card manufacturers. The Java Card API is compatible with formal international standards, such as ISO7816, and industry-specific standards, such as Europay/Master Card/Visa (EMV).

2.2 Java 2 Platform

With the Java 2 Platform, Micro Edition, Sun Microsystems has firmly established that one size does not fit all. The Java technology platform has now been re-architected into three basic editions: the Standard Edition for the desktop and work station environments, the Enterprise Edition targeted at large, server-based systems, and the Micro Edition which covers a range of small consumer devices from set-top boxes to pagers.



The Java 2 Platform, Enterprise Edition is a platform that enables solutions for developing, deploying and managing multi-tier server-centric applications. J2EE utilizes the Java 2 Platform Standard Edition to extend a complete, stable, secure, fast Java platform to the enterprise level. It delivers value to the enterprise by enabling a platform, which significantly reduces the cost and complexity of developing multi-tier solutions, resulting in services that can be rapidly deployed and easily enhanced.

The Java 2 Platform, Standard Edition provides a complete, secure foundation for building and deploying network-centric enterprise applications ranging from the PC desktop computer up to the workgroup server. The J2SE is implemented by the Java 2 Software Development Kit (SDK), Standard Edition, and the Java 2 Runtime Environment, Standard Edition. For versions prior to 1.2, the Java 2 SDK is called the Java Development Kit; this applies to versions JDK 1.1 and JDK 1.0.

The Java 2 Platform, Micro Edition is a new edition of the Java 2 platform targeted at consumer electronics and embedded devices. J2ME consists of a virtual machine and a minimal layer of APIs targeted at providing only enough functionality to securely and safely download Java classes to a device and to configure the Java environment. The rest of the Java functionality needed to provide a complete Java runtime environment for a particular kind of device is provided within the context of an industry-defined profile. J2ME comes in two configurations, that are targeted at two broad categories of devices: (1) those with 128-512 K of memory available for the Java environment and applications (based on the K virtual machine because of the very limited resources available in its target devices) and (2) those with 512 K+ available for the Java environment and applications (based on the full virtual machine appropriate for more powerful environments). Configuration number one (1) is called Connected Limited Device Configuration and configuration number two (2) is called Connected Device Configuration. Their profiles specify both APIs and the underlying edition and configuration as well as any VM options supported by the appropriate configuration.

3. MAJOR JAVA TERMS AND ABBREVIATIONS

Terms and abbreviations	**Description**
Applet	Applets are programs that require a browser to run. The <applet> tag is embedded in a Web page and names the program to be run. When a user, either over the Internet or corporate Intranet accesses that page, the applet automatically downloads from the server and runs on the client machine.
Embedded Java	This is used for devices with intermittent or no network connection and often no GUI. It can be configured to include only those classes and VM features needed for a particular device.
Java Card	The Java Card specifications enable Java technology to run on smart cards and other devices which possess only limited memory. The Java Card API allows applications that are written for one smart card platform enabled with Java Card technology to run on any other such platform.
JavaPhone	The JavaPhone API provides access to functionality unique to telephony devices. These devices will include direct telephony control, datagram messaging, address book, calendar and power management. It enables the safe delivery of dynamic information services on telephony devices including wireless smartphones and Internet screenphones.
JavaScript	JavaScript is a totally different thing to Java. JavaScript was originally defined by Netscape and has now been standardized as ECMAScript by ECMA. It is a scripting language that is widely supported in Web browsers and other Web tools. (JavaScripts are not supported in the Nokia 9210 Communicator). It may be easier to use than Java, but it is not as powerful and deals mainly with elements on the Web page. On the client, JavaScript is maintained as a source code embedded into an HTML document. On the server, it is compiled into bytecode, similar to Java programs. JavaScript is also used to tie Java applets together.
Java 2 Platform, Micro Edition (J2ME)	J2ME is a new edition of the Java 2 platform targeted at consumer electronics and embedded devices. The KVM is suitable for 16/32-bit RISC/CISC microcontrollers with a total memory of no more than a few hundred kilobytes and sometimes less than 128 Kb of RAM. This typically applies to digital cellular phones, pagers, mainstream personal digital assistants, low-end analog set-top boxes, and small retail payment terminals. The K virtual machine (KVM) is a new Java runtime environment that was built from the ground up. Its purpose is to provide an extremely lean implementation of the Java virtual machine for use in devices that have a small memory footprint (cell phones, pagers,

K Virtual Machine (KVM)	PDAs, and set-top boxes). The KVM is the core of the Java 2 Micro Edition (J2ME).[1]
JDK	Java development toolkit (= JRE and simple development tools). A version of JDK is typically used to indicate the different versions of Java technology. With Java 2 the JDK was renamed as Java 2 SDK (Software development kit).
Jini	A set of APIs that may be incorporated as part of an optimal package for any Java 2 Platform Edition. The Jini APIs enable transparent networking of devices and eliminate the need for system or network administration intervention by a user. The Jini technology is currently an optional package available on J2SE and J2EE.
JRE	A subset of the Java Developer Kit or SDK for end-users and developers who want to redistribute the runtime environment. The Java runtime environment consists of the Java virtual machine, the java core classes, and supporting files, but not development utilities.
JTAPI	Java Telephone API for which mobile extensions are being developed.
MExE	Mobile Execution Environment – a standardization effort of ETSI. MExE builds on JavaPhone and WAP specifications. It is not being currently implemented in NMP products.
Personal Java	A Java runtime environment for network-connectable applications on personal consumer devices for home, office and mobile use. The Java environment is being implemented in the Nokia 9210 Communicator.
Sandbox	Sandbox comprises a number of cooperating system components, ranging from security managers that execute as part of the application, to security measures incorporated into the JVM and the language itself. The sandbox insures that an untrusted and possibly malicious application cannot gain access to the system recourses.
Standalone application	A Java application that can run without an HTML browser.

4. NOKIA 9210 COMMUNICATOR JAVA[2]

The Java platform in the Nokia 9210 Communicator is an implementation of the Sun Microsystems PersonalJava (PersonalJava Application Environment 1.1.1 Specification: http://java.sun.com/products/personaljava/spec-1-1-1/index.html). Additionally the Nokia 9210 Communicator supports JavaPhone specifications and some Symbian proprietary APIs. Currently there is no integration between the WAP browsers and Java. The Java platform is able to run third-party applications that are independent of browsers and are thus similar to any PDA application run in the device. These

[1] Sun Microsystems has made a KVM implementation for the Nokia 9110 Communicator.
[2] Nokia 9210 Communicator specifications are subject to change.

standalone applications are installed only at the user's request. The user should not see the distinction between native applications and those written in Java, if the Java applications are specially designed for the Nokia 9210 Communicator (considering the screen size, lack of mouse and pen).

Java applications have the same access rights to the functions of the Nokia 9210 Communicator as any other third-party software (EPOC) has. The phone applications use JavaPhone API. The Java applications have access to at least Calendar and Contact Manager data through standardized interfaces (defined in JavaPhone API).

4.1 Execution of Java programs

4.1.1 Loading and running standalone applications

The device shall allow downloading of new standalone applications via a GSM connection, via a serial or IR link, or in a Memory Card. New standalone applications are installed only at the users explicit request. Standalone Java applications are packaged, like native EPOC applications, in SIS files. Currently Java's own JAR packaging is not supported.

There is a standard installation scheme for standalone applications that store application-specific (configuration) data in a defined way. The standalone applications shall have access to application-specific files by standard convention without needing to know where the user chose to put them.

It is possible to uninstall an application so that all configuration data are also removed.

The standalone applications can be started from the Extras-menu of the Nokia 9210 Communicator PDA or from native applications.

4.2 PDA functions

The Java implementation in the Nokia 9210 Communicator is extendable so that new class libraries can be installed on the device. The user-installed classes and applications are removable and replaceable, but they are stored in non-volatile memory (i.e. flash or battery-backed RAM).

The standard set of libraries are pre-loaded on the system so that it is not necessary to download them when an application starts.

4.3 Backup and restore

It is possible to backup and restore the stored applications and application-specific data. The Nokia 9210 Communicator's general functionality is used. It is also possible to backup and restore installed classes.