1. **What is Java and History of Java**

Java is a [programming language](http://en.wikipedia.org/wiki/Programming_language) originally developed by [James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) (which has since [merged into Oracle Corporation](http://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](http://en.wikipedia.org/wiki/Java_(software_platform))

Java is a [high-level programming language](http://www.webopedia.com/TERM/H/high_level_language.html) was originally called OAK, and was designed for handheld devices and set-top boxes. Oak was unsuccessful so in 1995 Sun changed the name to Java and modified the language to take advantage of the burgeoning [World Wide Web](http://www.webopedia.com/TERM/W/World_Wide_Web.html).

It is the underlying technology that powers state-of-the-art programs including utilities, games, and business applications. Java runs on more than 850 million personal computers worldwide, and on billions of devices worldwide, including mobile and TV devices

Java is an [object-oriented language](http://www.webopedia.com/TERM/O/object_oriented_programming_OOP.html) similar to [C++](http://www.webopedia.com/TERM/C/C_plus_plus.html), but simplified to eliminate language features that cause common programming errors. Java [source code](http://www.webopedia.com/TERM/S/source_code.html) files (files with a .java extension) are [compiled](http://www.webopedia.com/TERM/C/compile.html) into a format called bytecode (files with a .class extension), which can then be executed by a Java [interpreter](http://www.webopedia.com/TERM/I/interpreter.html). Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (VMs), exist for most [operating systems](http://www.webopedia.com/TERM/O/operating_system.html), including [UNIX](http://www.webopedia.com/TERM/U/UNIX.html), the [Macintosh](http://www.webopedia.com/TERM/M/Macintosh_computer.html) OS, and [Windows](http://www.webopedia.com/TERM/W/Windows.html). Bytecode can also be converted directly into [machine language](http://www.webopedia.com/TERM/M/machine_language.html) instructions by a [just-in-time compiler (JIT)](http://www.webopedia.com/TERM/J/JIT.html).

1. **Terminology used in Java world.**

|  |
| --- |
|  |
| |  | | --- | | 1. **JDK (Java Development Kit)**   Java Developer Kit contains tools needed to develop the Java programs, and JRE to run the programs. The tools include compiler (javac.exe), Java application launcher (java.exe), Appletviewer, etc. The most current version is the J2SE.  If you use J2SE 1.2 and later to develop applications, you are using what's known as the [Java 2 Platform](http://www.programmersheaven.com/2/FAQ-JAVA-What-Is-JDK). The latest Java 5 or J2SE 1.5 is Sun’s most ambitious upgrade that has several enhancements to the [Java language](http://www.programmersheaven.com/2/FAQ-JAVA-What-Is-JDK).  Compiler converts java code into byte code. Java application launcher opens a JRE, loads the class, and invokes its main method.  You need JDK, if at all you want to write your own programs, and to compile them. For running java programs, JRE is sufficient.  JRE is targeted for execution of Java files  i.e. JRE = JVM + Java Packages Classes(like util, math, lang, awt,swing etc)+runtime libraries.  JDK is mainly targeted for java development. I.e. You can create a Java file (with the help of Java packages), compile a Java file and run a java file   1. **JRE (Java Runtime Environment)**   Java Runtime Environment contains JVM, class libraries, and other supporting files. It does not contain any development tools such as compiler, debugger, etc. Actually JVM runs the program, and it uses the class libraries, and other supporting files provided in JRE. If you want to run any java program, you need to have JRE installed in the system  The Java Virtual Machine provides a platform-independent way of executing code; programmers can concentrate on writing software, without having to be concerned with how or where it will run.  If u just want to run applets (ex: Online Yahoo games or puzzles), JRE needs to be installed on the machine.   1. **JVM (Java Virtual Machine)**   As we all aware when we compile a Java file, output is not an 'exe' but it's a '.class' file. '.class' file consists of Java byte codes which are understandable by JVM. Java Virtual Machine interprets the byte code into the machine code depending upon the underlying operating system and hardware combination. It is responsible for all the things like garbage collection, array bounds checking, etc… JVM is platform dependent.  The JVM is called "virtual" because it provides a machine interface that does not depend on the underlying operating system and machine hardware architecture. This independence from hardware and operating system is a cornerstone of the write-once run-anywhere value of Java programs.  There are different JVM implementations are there. These may differ in things like performance, reliability, speed, etc. These implementations will differ in those areas where Java specification doesn’t mention how to implement the features, like how the garbage collection process works is JVM dependent, Java spec doesn’t define any specific way to do this.   1. **JAR (Java Archive) Files**   A Java Archive File, commonly referred to as a Jar File is nothing more than a compressed archive file. The file has two important characteristics:   * The archive does not have to be compressed but if it is, the Zip compression algorithm is used. * The archive's filename must have the .jar extension.   Archives are used to bundle a number of files and directories into one single file, as such Java Archives or Jar Files are no different. Typically, Jar files are used to distribute Java software, these Jar files contains the actual class files of the programs archived in them.  Some benefits of using Jar files are:   * Security, A jar file can be digitally signed enabling users to verify the signature and then grant the program security privileges. * Decreased download time, since the archive is compressed it takes less time to download than it would to download each individual file. * Package Versioning, A Jar file may contain vendor and version information about the files it contains. * Portability, all Java Runtime Environments know how to handle Jar files.  1. **Environment variable PATH and CLASSPATH**   how to use the PATH and CLASSPATH environment variables on Microsoft Windows, Solaris, and Linux. Consult the installation instructions included with your installation of the Java Development Kit (JDK) software bundle for current information.  Also don't confuse Classpath with [PATH in Java](http://javarevisited.blogspot.sg/2011/10/how-to-set-path-for-java-unix-linux-and.html), which is another environment variable used to find java binaries located in JDK installation directory, also known as [JAVA\_HOME](http://javarevisited.blogspot.sg/2012/02/how-to-set-javahome-environment-in.html). Main difference between PATH and CLASSPATH is that former is used to locate Java commands while later is used to locate Java class files.  After installing the software, the JDK directory will have the structure shown below.    The bin directory contains both the compiler and the launcher.  **Updating the PATH Environment Variable (Microsoft Windows)**  We can run Java applications just fine without setting the PATH environment variable. Or, you can optionally set it as a convenience.  Set the PATH environment variable if you want to be able to conveniently run the executables (javac.exe, java.exe, javadoc.exe, and so on) from any directory without having to type the full path of the command. If you do not set the PATH variable, you need to specify the full path to the executable every time you run it, such as:  C:\Java\jdk1.7.0\bin\javac MyClass.java  The PATH environment variable is a series of directories separated by semicolons (;). Microsoft Windows looks for programs in the PATH directories in order, from left to right. You should have only one bin directory for the JDK in the path at a time (those following the first are ignored), so if one is already present, you can update that particular entry.  **The following is an example of a PATH environment variable:**  C:\Java\jdk1.7.0\bin;C:\Windows\System32\;C:\Windows\;C:\Windows\System32\Wbem  It is useful to set the PATH environment variable permanently so it will persist after rebooting. To make a permanent change to the PATH variable, use the System icon in the Control Panel. The precise procedure varies depending on the version of Windows:  **Windows XP**  Select Start, select Control Panel. double click System, and select the Advanced tab.  Click Environment Variables. In the section System Variables, find the PATH environment variable and select it. Click Edit. If the PATH environment variable does not exist, click New.  In the Edit System Variable (or New System Variable) window, specify the value of the PATH environment variable. Click OK. Close all remaining windows by clicking OK.  **Windows Vista:**  From the desktop, right click the My Computer icon.  Choose Properties from the context menu.  Click the Advanced tab (Advanced system settings link in Vista).  Click Environment Variables. In the section System Variables, find the PATH environment variable and select it. Click Edit. If the PATH environment variable does not exist, click New.  In the Edit System Variable (or New System Variable) window, specify the value of the PATH environment variable. Click OK. Close all remaining windows by clicking OK.  **Windows 7:**  From the desktop, right click the Computer icon.  Choose Properties from the context menu.  Click the Advanced system settings link.  Click Environment Variables. In the section System Variables, find the PATH environment variable and select it. Click Edit. If the PATH environment variable does not exist, click New.  In the Edit System Variable (or New System Variable) window, specify the value of the PATH environment variable. Click OK. Close all remaining windows by clicking OK.  Note: You may see a PATH environment variable similar to the following when editing it from the Control Panel:  %JAVA\_HOME%\bin;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem  Variables enclosed in percentage signs (%) are existing environment variables. If one of these variables is listed in the Environment Variables window from the Control Panel (such as JAVA\_HOME), then you can edit its value. If it does not appear, then it is a special environment variable that the operating system has defined. For example, SystemRoot is the location of the Microsoft Windows system folder. To obtain the value of a environment variable, enter the following at a command prompt. (This example obtains the value of the SystemRoot environment variable):  echo %SystemRoot% Setting Java Classpath in Windows Classpath in Java is path to directory or list of directory which is used by ClassLoaders to find and [load class in Java program](http://javarevisited.blogspot.sg/2012/07/when-class-loading-initialization-java-example.html). Classpath can be specified using CLASSPATH environment variable which is case insensitive, -cp or -classpath command line option or Class-Path attribute in manifest.mf file inside [JAR file in Java](http://javarevisited.blogspot.sg/2012/03/how-to-create-and-execute-jar-file-in.html).  CLASSPATH is an [environment variable](http://javarevisited.blogspot.sg/2012/08/how-to-get-environment-variables-in.html) which is used by [Java Virtual Machine](http://javarevisited.blogspot.sg/2011/11/hotspot-jvm-options-java-examples.html) to locate user defined classes. we will see How to setup classpath for java in windows and Linux , java -classpath example in different scenario and use of java -classpath or java -cp.  In order to set Classpath for Java in Windows (any version either Windows XP,  Windows 2000 or Windows 7) you need to specify value of environment variable CLASSPATH, name of this variable is not case sensitive and it doesn’t matter if name of your environment variable is [Classpath](http://java67.blogspot.sg/2012/08/what-is-path-and-classpath-in-java-difference.html), CLASSPATH or [classpath](http://javabuddy.hubpages.com/hub/What-is-PATH-and-CLASSPATH-in-Java-set-Windows-Unix-Linux) in Java.  Here is Step by Step guide for setting Java Classpath in Windows:  [How to se Java Classpath in windows and Unix Linux](http://javarevisited.blogspot.com/2011/01/how-classpath-work-in-java.html)  Go to Environment variable window in Windows by pressing "Windows + Pause “-->Advanced -->Environment variable " or you can go from right click on my computer than choosing properties and then Advanced and then Environment variable this will open Environment variable window in windows.  Now specify your environment variable CLASSPATH and put the value of your JAVA\_HOME\lib and also include [current directory](http://javarevisited.blogspot.sg/2012/04/how-to-find-current-directory-in-java.html) by including (dot or period sign).  Now to check the value of Java classpath in windows type "echo %CLASSPATH" in your DOS command prompt and it will show you the value of directory which are included in CLASSPATH.  You can also set classpath in windows by using DOS command like :  set CLASSPATH=%CLASSPATH%;JAVA\_HOME\lib; This way you can set classpath in Windows XP, windows 2000 or Windows 7 and 8, as they all come with command prompt.   1. **Running “Hello World ” Program in Java** | |  | |

**Reference :**

http://www.youtube.com/watch?v=GNTphyx-Q7o