Java Editions , JRE JVM JDK

Types of Java Application

Standalone applications are the application which runs on separate computer process without adding any file processes. The standalone application is also known as Java GUI Applications or **Desktop Applications** which uses some standard GUI components such as AWT(Abstract Windowing Toolkit), swing and JavaFX and this component are deployed to the desktop. These components have buttons, menu, tables, GUI widget toolkit, 3D graphics etc. using this component a traditional software is developed which can be installed in every machine.

### **2. Web Applications**

Web Applications are the client-server software application which is run by the client. Servlets, struts, JSP, Spring, hibernate etc. are used for the development of a client-server application. eCommerce application is also developed in java using eCommerce platform i.e Broadleaf.

# **Features of Java**

The prime reason behind creation of Java was to bring portability and security feature into a computer language. Beside these two major features, there were many other features that played an important role in moulding out the final form of this outstanding language. Those features are :

Object Oriented,Robust,Platform Independent,Secure

## Java Editions

1. Java Standard Edition

Java Standard edition is a computing platform which is used for development and deployment of portable code that is used in desktop and server environments. Java Standard Edition is also known as Java 2 Platform, Standard Edition (J2SE)

2. Java Enterprise Edition

Java Enterprise Edition is a set of specifications and extending Java SE 8 with features such as distributed computing and web services. The applications of Java Enterprise Edition run on reference runtimes. This reference runtime handle transactions, security, scalability, concurrency and the management of components to be deployed. Java Enterprise Edition is also known as Java 2 Platform Enterprise Edition (J2EE), and currently, it has been rebranded as Jakarta EE.

**Example:** e-commerce, accounting, banking information systems.

3 JavaFX

JavaFX is used for creating desktop applications and also rich internet applications(RIAs) which can be run on a wide variety of devices. JavaFX has almost replaced Swing as the standard GUI library for Java Standard Edition. JavaFX support for desktop computers and web browsers.

# [**Whats the main need of Environment variable?**](https://stackoverflow.com/questions/5284166/whats-the-main-need-of-environment-variable)

Environment variables are set to allow access to command line tools and to enable other tools to interact with SDKs more easily. For example, with Java on Windows, if the environment variable is not set on the PATH, running javac is much more cumbersome because you need to type in the full path to the command each time:

C:> \jdk<version>\bin\javac MyClass.java

In Java setting the environment variables isn't required; it's just easier. Other languages may be more stringent, though I haven't seen any specific examples I could cite. You can read the article [How Do I Set the Path System variable?](http://www.java.com/en/download/help/path.xml) for specifics on how to do this.

So whenever a program will require Java environment, it will look for the **java** environment variable which will give it the path to the execution directory.

## Is java compiled language or interpreted language

Programming languages are classified based on their levels of abstraction. We differentiate high-level languages (Java, Python, JavaScript, C++, Go), low-level (Assembler), and finally, machine code.

Every high-level language code, like Java, **needs to be translated to machine native code for execution.** This translation process can be either compilation or interpretation. However, there is also a third option. A combination that seeks to take advantage of both approaches.

Compiled languages (C++, Go) are converted directly into machine native code by a compiler program.

On the other hand, in interpreted languages (Python, JavaScript), there are no build steps. Instead, interpreters operate on the source code of the program while executing it.

[Java and the JVM](https://www.baeldung.com/jvm-languages) were designed with portability in mind. Therefore, most popular platforms today can run Java code.

This might sound like a hint that Java is a purely interpreted language. However, before execution, **Java source code needs to be compiled into**[**bytecode**](https://www.baeldung.com/java-class-view-bytecode)**. Bytecode is a special machine language native to the JVM*.***The JVM interprets and executes this code at runtime.

It is the JVM that is built and customized for each platform that supports Java, rather than our programs or libraries.

Modern JVMs also have a JIT compiler. **This means that the JVM optimizes our code at runtime** to gain similar performance benefits to a compiled language.

# **Java JVM, JDK and JRE**

JVM

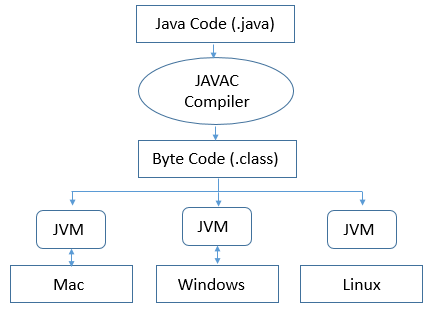
Java Virtual Machine (JVM) is a specification that provides runtime environment in which java **bytecode** can be executed. As the name implies, the JVM acts as a “virtual” machine or processor. Java's platform independence consists mostly of its **Java Virtual Machine (JVM)** .

The JVM performs following operation:

* Loads code
* Verifies code
* Executes code

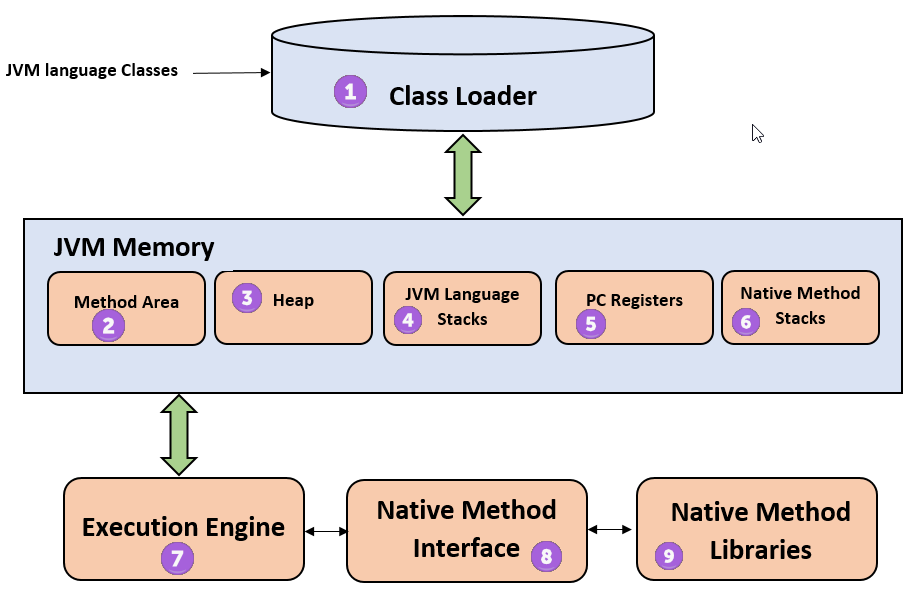
In most cases, other programming languages, the compiler produce code for a particular Operating System but the Java compiler produce Bytecode only for a **Java Virtual Machine** .

 When you run a Java program, it runs as a thread within the JVM process. It is the JVM's responsibility to load your class files, verify code, interpret them and execute them. When you issue a command like java , the JVM loads the class definition for that particular class and calls the main method of that class.



It is the JVMs responsibility that makes it possible for the same class file to run on any other **Operating Systems** . The JVM takes your compiled platform-neutral byte code and interprets it to run platform-specific machine code.

JVM Archiecture



Java Development Kit

The Java Development Kit (JDK) is a software development environment used for developing Java applications and applets.

 It includes the Java interpreter, Java classes, and Java development tools: compiler, debugger, disassembler, appletviewer, stub file generator, and documentation generator.

The JDK enables you to write applications that are developed once and run anywhere on any Java virtual machine. Java applications that are developed with the JDK on one system can be used on another system without changing or recompiling the code. The Java class files are portable to any standard Java virtual machine.

Java Run time environment

**What is a runtime environment**

A software program needs to execute, and to do that, it needs an environment to run in. In the past, most software used the operating system (OS) as the runtime environment. The program ran inside whatever computer it was on and relied directly on operating system settings for resource access; resources like memory, disk access, and network access. The Java Runtime Environment changed all that, at least for Java programs. In the case of Java and other JVM-based languages, the JRE creates an intermediary between the operating system and the actual program. The JRE loads class files and starts a virtual machine (the JVM) that ensures there is access to memory and other system resources in a consistent form across many operating systems.

JRE

