What is Java?

- Java is a true, secure, robust object oriented programming language, developed by Sun Microsystems.
- It is the world's most important and widely used computer language.
- Java originally designed to develop software for consumer electronic devices like TV, VCR, Toaster, Ovens etc.
- It was developed by James Ghosling, Patrick Naughton, Mike Sheridan at Sun Microsystems Inc as a team named as Green Project Team, headed by James Ghosling.
- This team demonstrated this language for electronic devices by making use of small touch sensitive Screens to control the working of these electronic devices.
- Java is a platform-independent language because it has runtime environment. Platform means a hardware or software environment in which an application runs.
- Java codes are compiled into byte code or machine-independent code. This byte code is run on JVM (Java Virtual Machine).
- The syntax is Java is almost the same as C/C++. But java does not support low-level programming functions like pointers. The codes in Java are always written in the form of Classes and Objects.
- The trouble with C and C++(and most other languages) is that they are designed to be compiled for specific target. Although it is possible to combine these programs for just about any type of the CPU.

  Java is related to C++ which is directly descendent of C. Much of the character of Java is inherited from these two languages. From C, Java derived its syntax. Many of java's Object Oriented features influenced by C++.

History of Java

- Java was developed by a team of programmer of Sun Microsystems of the USA known as Green Project Team.
- This team consisted of James Ghosling, Bill Joy, Patrick Naughton, Mike Sheridan, headed by James Ghosling.
- The version of java which was appeared in 1991, was written in 18 months at Sun Microsystems.
- Initially, Java was called **"Greentalk"** by James Gosling.
- Later it was called Oak. It was used internal at Sun.
- Oak is a symbol of strength and chosen as a national tree of many countries
- The public announcement of Java was in the spring of 1995.

- Java is an island in Indonesia, here the first coffee was produced or we call Java coffee. James Ghosling chose this name while having coffee near his office.
- The word JAVA does not have an acronym. It is just a name.
- Many popular companies like Netscape, Microsoft announce their support for Java.

Java Version History

Version Name	**Coad Name**	**Release Date**	**Description**
Java Alpha and Beta		1995	• It was the 1st version but was having unstable APIs and ABIs. • It was the 1st version but was having unstable APIs and ABIs.
JDK 1.0	Oak	January 1996	• 1st version
JDK 1.1		February 1997	• AWT Event modelling retooling. • Added Inner class, Java Beans, JDBC, RMI, Reflection, JIT • Added Inner class, Java Beans, JDBC, RMI, Reflection, JIT
J2SE 1.2	Playground	December 1998	• JDK replaced by J2SE.

			• Support strictfp keyword. • Swing API integrated with core classes. • Collection framework.
J2SE 1.3	Kestrel	May 2000	• HotSPot JVM included • RMI Modified. • JNDI(Java Naming and Directory Interface) Supported • JPDA(Java Platform Debugger Architecture). • Included Proxy Classes.
J2SE 1.4	Merin	February 2002	• Support assert Keyword. • Improvement in libraries. • Support Regular expression. • Support Exception Chaining. • Support Exception Chaining. • Included Java Web Start. • Support API Preferences (java.util.prefs).
J2SE 5.0	Tiger	September 2004	• Included Generics,

			Metadata, Autoboxing/Unboxing, Enumerations, Varargs. • Enhanced for each loop. • Support static imports.
Java SE 6	Mustang	December 2006	• Support Win9x version. • Support Scripting languages. • Improved Swing performance. • Support JDBC 4.0 • Upgrade of JAXB to 2.0. • Improvement in GUI and JVM.
Java SE 7	Dolphine	July 2011	• Support of dynamic language in JVM. • Included 64-bit pointers. • Support string in the switch. • Support resource management in the try block. • Support binary integer literals. • Support underscore in numeric literals. • Support multiple

			exceptions. • Included I/O library.
Java SE 8(LTS)		March 2014	• Support of JSR 335 and JEP 126. • Support unsigned integer. • Support Date and time API. • Included JavaFX. • Support Windows XP.
Java SE 9		September 2017	• Support multiple gigabyte heaps. • Included garbage collector.
Java SE 10		March 2018	• Support local variables type inference. • Support local variables type inference. • Included Application class.
Java SE 11(LTS)		September 2018	• Support bug fixes. • Include long term support(LTS). • Support transport layer security.
Java SE 12		March 2019	• Support JVM Constant API.

			• Include CDS Archives.
Java SE 13		September 2019	• Updated Switch Expressions. • Include Text Blocks. • Support Legacy socket API.
Java SE 14		March 2020	• Support Event Streaming. • Improved NullPointerException. • Improved NullPointerException. • Removal of the Concurrent Mark Sweep (CMS) in the garbage collector.

Application of Java

Java is widely used in every corner of world and of human life. Java is not only used in softwares but is also widely used in designing hardware controlling software components. .

Following are some other usage of Java

1. Developing Desktop Applications
2. Web Applications like Linkedin.com, Snapdeal.com etc
3. Mobile Operating System like Android
4. Embedded Systems
5. Robotics and games etc.

Types of Java Application

Java application can be classified as follows:

1. Standalone Applications

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 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.

Example: Media player, antivirus, Paint etc.

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.

Example: mail, e-commerce website, bank website etc.

3. Enterprise Application

Enterprise application is middleware applications. To use software and hardware systems technologies and services across the enterprises. It is designed for the corporate area such as banking business systems.

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

4. Mobile Application

For mobile applications, java uses ME or J2ME framework. This framework are the cross platform that runs applications across phones and smartphones. Java provides a platform for application development in android.

Example: WhatsApp, Xender etc.

Features of Java

The inventors of java wanted to design language which offers to the solution to the problem encounter programming language. It is designed to be not only reliable, portable and distributed but also simple, secure and interactive.

The most striking features of the language is that, it is platform neutral. Java is the first programming language that is not tied to any particular hardware or OS. Programs developed in java can be run anywhere on any system.

The most important features are….

- Simple
- Secure
- Portable
- Object Oriented
- Robust
- Multithreading
- Architecture neutral
- Compiled and Interpreted
- Distributed
- High performance
- Dynamic

1) Simple

Java is easy to learn and its syntax is quite simple, clean and easy to understand.The confusing and ambiguous concepts of C++ are either left out in Java or they have been re-implemented in a cleaner way.

Eg : Pointers and Operator Overloading are not there in java but were an important part of C++.

2) Secure

When it comes to security, Java is always the first choice. With java secure features it enable us to develop virus free, temper free system. Java program always runs in Java runtime environment with almost null interaction with system OS, hence it is more secure.

3) Portable

Java Byte code can be carried to any platform. No implementation dependent features. Everything related to storage is predefined, example: size of primitive data types

4) Object Oriented

Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.

Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

Basic concepts of OOPs are:

1. Object
2. Class
3. Inheritance
4. Polymorphism

5. Abstraction
6. Encapsulation

5) Robust

Robust simply means strong. Java is robust because:

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- There is automatic garbage collection in java which runs on the Java
- Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are exception handling and the type checking mechanism in Java. All these points make Java robust.

6) Multithreading

Java multithreading feature makes it possible to write program that can do many tasks simultaneously. Benefit of multithreading is that it utilizes same memory and other resources to execute multiple threads at the same time, like While typing, grammatical errors are checked along.

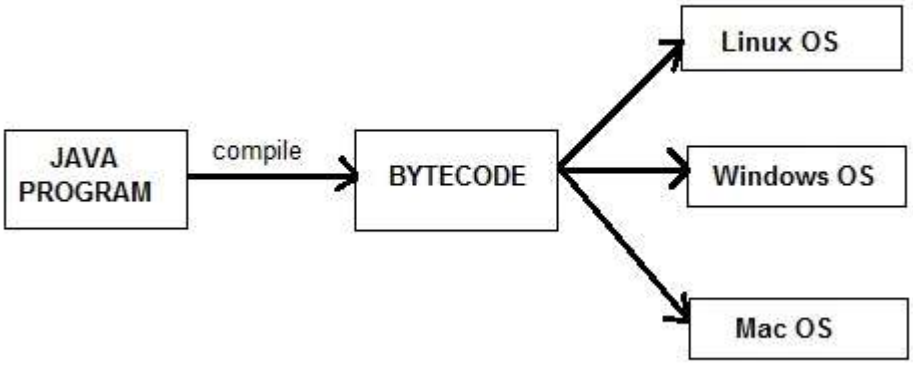
7) Architecture Neutral

Compiler generates byte codes, which have nothing to do with particular computer architecture; hence a Java program is easy to interpret on any machine.

8) Compile and interpreted

Unlike other programming languages such as C, C++ etc which are compiled into platform specific machines. Java is guaranteed to be write-once, run-anywhere language.

On compilation Java program is compiled into bytecode. This bytecode is platform independent and can be run on any machine, plus this bytecode format also provide security. Any machine with Java Runtime Environment can run Java Programs.



9) High performance

Java is faster than other traditional interpreted programming languages because Java bytecode is "close" to native code. It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

10) Distributed

Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

11.) Dynamic

Java is a dynamic language. It supports dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++.

Java supports dynamic compilation and automatic memory management (garbage collection).

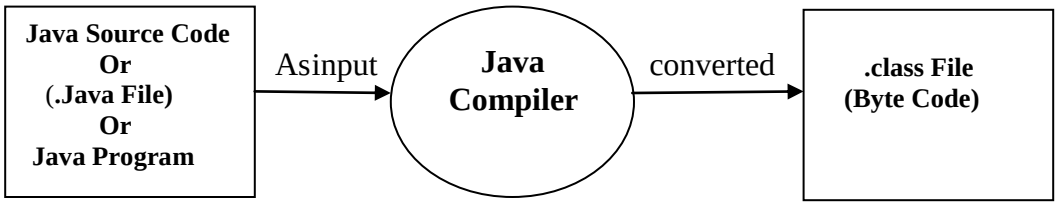
Platform

A platform is the basic hardware (computer) and software (operating system) on which software applications can be run.

Platform is also a group of technologies that are used as a base upon which other applications, processes are developed.

Byte Code

When a Java program (Source Code) is compiled through the java compiler, **.class** file is generated. This Code is known as Byte code. It is not in human readable form. It works as input for JVM (Java Virtual Machine) and then JVM convert this file for related Machine.

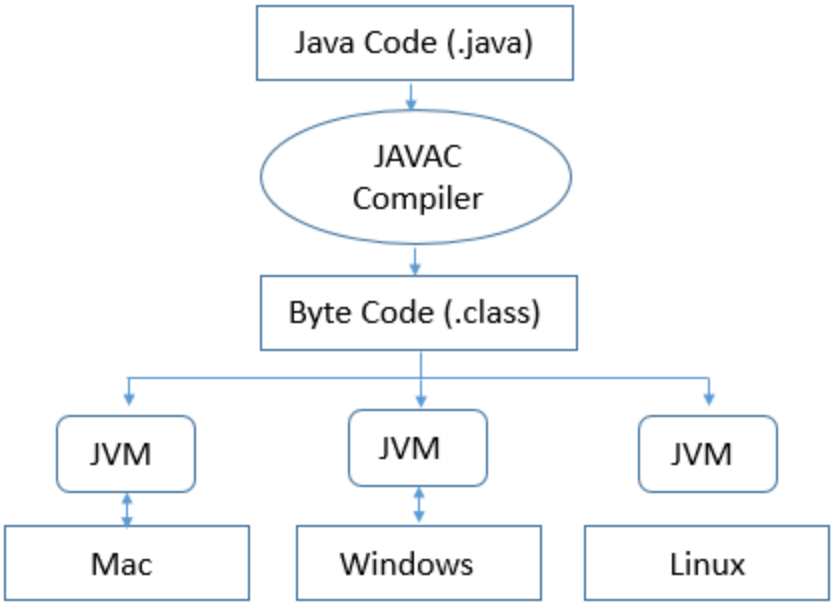


JVM

- JVM stands for Java Virtual Machine.
- It converts .class file i.e. Byte code into related machine code for the execution.
- Or Java Virtual Machine (JVM) is a specification that provides runtime environment in which java byte code can be executed.
- 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 Byte code only for a Java Virtual Machine .
- When we run a Java program,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 the compiled platform-neutral byte code and interprets it to run platform-specific machine code.

Java Vs. C++

- Java is platform-independent. Once compiled into byte code, it can be executed on any platform.
  C++ is a platform dependent language. The source code written in C++ needs to be compiled on every platform.
- Java is a compiled as well as an interpreted language.
  The compiled output of a Java source code is a byte code which is platform-independent.

  C++ is a compiled language.The source program written in C++ is compiled into an object code which can then be executed to produce an output.

- Java, however, translates the code into byte code.This byte code is portable and can be executed on any platform.
  C++ code is not portable.It must be compiled for each platform.
- In Java the memory management is system-controlled.

Memory management in C++ is manual.We need to allocate/deallocate memory manually using the new/delete operators.

- Java does not Support Multiple Inheritance. Effects of multiple inheritance can be achieved using the interfaces in Java.
  C++ Supports multiple inheritance.
- In Java, only method overloading is allowed.It does not allow operator overloading
  In C++, methods and operators can be overloaded. This is static polymorphism.
- In Java, the virtual keyword is absent. However, in Java, all non-static methods by default can be overridden.
  Or in simple terms, all non-static methods in Java are virtual by default.
  As a part of dynamic polymorphism, in C++, the virtual keyword is used with a function to indicate the function that can be overridden in the derived class. This way we can achieve polymorphism.
- We cannot use pointers in Java as leisurely as we can use in C++.
  C++ is all about pointers. C++ has strong support for pointers and we can do a lot of useful programming using pointers.