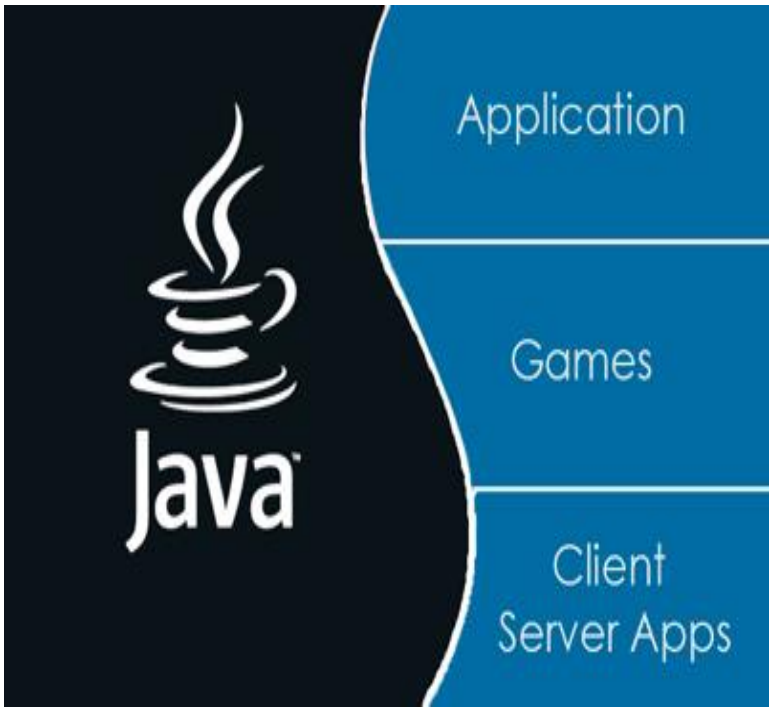
Unit-I Introduction to Java



*Java is an **object-oriented language** derived from C++ with strong support for networking, multithreading, and component-oriented development.*

*The first version of JDK (Java **Development Kit**) 1.0, released in 1995, established the first major public Java standard, with a C++ like syntax.*

A hierarchy of classes including support for networking, streams, event handling, exception handling, multithreading, and GUI development with the Abstract Windowing Toolkit (AWT).

Soon afterwards the Java Database Connectivity (JDBC) classes were introduced as a means of communicating with databases.

*Unlike other programming languages like **C** or **C++**, though, Java programs are not compiled into machine code; instead, they are converted into an architecture-neutral bytecode format. This collection of bytes represents the code for an abstract **Java virtual machine** (JVM).*

In order for these bytes to execute on a physical machine, a Java interpreter running on that physical machine must translate those bytes into local actions, such as printing a string or drawing a button.

➢ Introduction

Java was originally evolving by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' **Java** platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The history of Java is very interesting. Java was originally designed for interactive television, but it was too advanced technology for the digital cable television industry at the time. The history of java starts from Green Team. Java team members (also known as Green Team), initiated this project to develop a language for digital devices such as set-top boxes, televisions etc. But, it was suited for internet programming. Later, Java technology was incorporated by Netscape.Currently, Java is used in internet programming, mobile devices, games, e-business solutions etc.

• Why Java named as "Java"…..

*The team gathered to choose a new name. The suggested words were "dynamic", "revolutionary", "Silk", "jolt", "DNA" etc. They wanted something that reflected the essence of the technology: revolutionary, dynamic, lively, cool, unique, and easy to spell and fun to say.According to James Gosling "Java was one of the top choices along with **Silk**". Since java was so unique, most of the team members preferred java. Java is an island of Indonesia where first coffee was produced (called java coffee). Notice that Java is just a name not an acronym.Originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.* In 1995, *Time magazine called Java one of the Ten Best Products of 1995. JDK 1.0 released in(January 23, 199*6).

• Java Version History

There are many java versions that has been released. Current stable release of Java is Java SE 9.

1	JDK Alpha and Beta (1995)
2	JDK 1.0 (23rd Jan, 1996)
3	JDK 1.1 (19th Feb, 1997)
4	J2SE 1.2 (8th Dec, 1998)
5	J2SE 1.3 (8th May, 2000)
6	J2SE 1.4 (6th Feb, 2002)
7	J2SE 5.0 (30th Sep, 2004)
8	Java SE 6 (11th Dec, 2006)
9	Java SE 7 (28th July, 2011)
10	Java SE 8 (18th March, 2014)
11	Java SE 9 (21st Sep, 2017)
12	And Many More….

Table 1.1: Java Tools

1.1 What is Java?

Java is a programming language and computing platform first released by Sun Microsystems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is a programming language which having several features like fast, secure, and reliable. It is does not depends on platform again Java is a high level, robust, secured and object-oriented programming language. Any hardware or software environment in which a program runs, is known as a platform. Since Java has its own runtime environment (JRE) and API, it is called platform.

1.2Java Tools and its Environment

It includes four main part like JDK (Java Development Tool Kit), JSL (Java Standard Library) or API (Application Programming Interface), JVM (Java Virtual Machine), and JRE (Java Runtime Environment). Java environment includes a large number of development tools and thousands of classes and Methods.

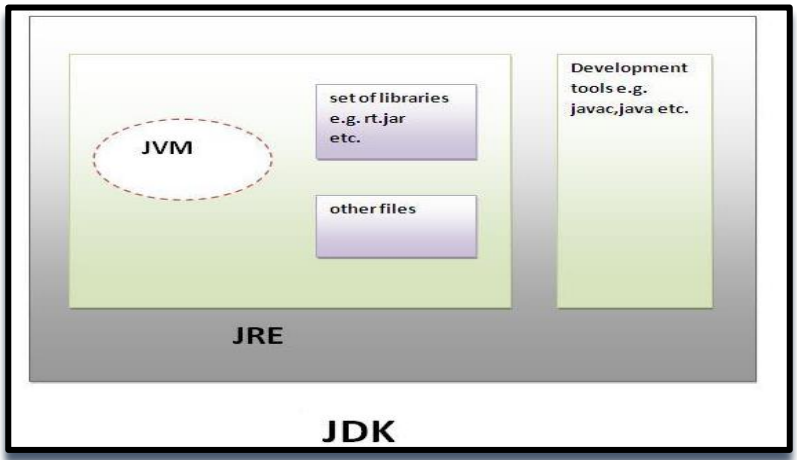


Fig1.1: Java Environment

1.2.1 SDK

A software developer's kit (**SDK**) is a set of programs used by a computer programmer to write application programs. Typically, an **SDK** includes a visual screen builder, an editor, a compiler, a linker, and sometimes other facilities. The term is used by Microsoft, Sun Microsystems, and a number of other companies.

1.2.2 JDK

The Java Development Kit (JDK) is a software development environment used for developing Java applications and applets. It includes the Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc) and other tools needed in Java development.

JDK Tools

1. **Javac: -** The java compiler which translate java source code to byte code so that the interpreter can understand the file. (.java file Converted into .class file)

2. **Java: -** Java interpreter which runs applet and application by reading and interpreting bytecode files.
3. **Appletviewer: -** It is use to run java applets
4. **Javadoc: -**Creates HTML format documentation from java source code files.
5. **Javah: -**Produces header files for use with native methods.
6. **Javap: -** Java Disassemblers, which enables us to convert bytecode files into a program description.
7. **Jdb: -** Java debugger, which helps us to find error in our programs.

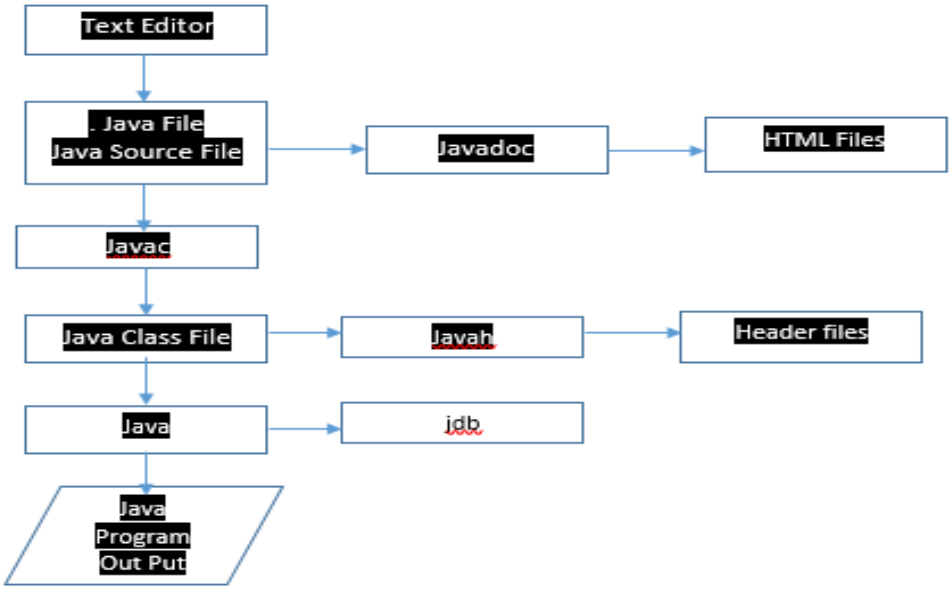


Fig1.2 Java Program Creation and Execution

1.2.3 JVM

JVM (Java Virtual Machine) is an abstract machine. Java compiler produce an intermediate code kwon as byte code for a machine that does not exists. This machine is called as JVM (Java Virtual Machine) and it exists only inside the computer memory.

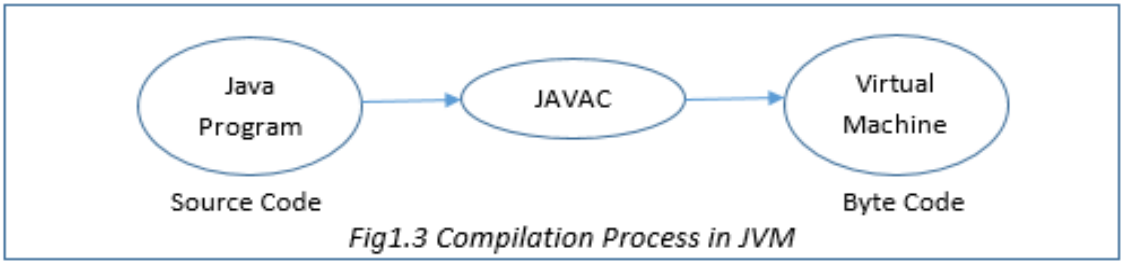


Fig1.3 Compilation Process in JVM

The Virtual Machine code is not Machine Specific. The Machine Specific code generated by Java Interpreter by acting an intermediate stage between Virtual Machine and the Real Machine.

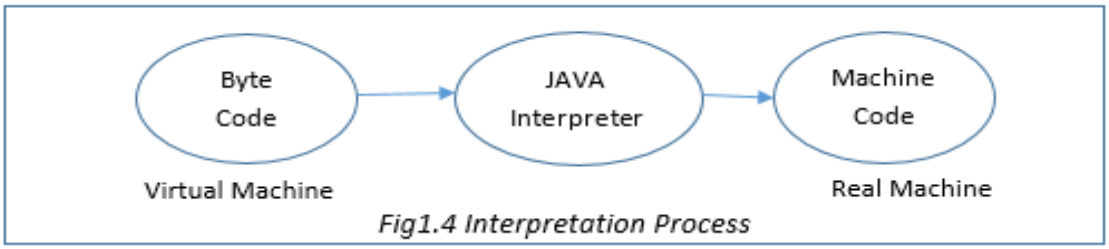


Fig1.4 Interpretation Process

It is a specification that provides runtime environment in which java bytecode can be executed. JVMs are available for many hardware and software platforms. JVM, JRE and JDK are platform dependent because configuration of each OS differs. But, Java is platform independent. There are three notions of the JVM: specification, implementation, and instance. JVM is changed according to its SE versions.

1. **A specification** where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Sun and other companies.
2. **An implementation** Its implementation is known as JRE (Java Runtime Environment).
3. **Runtime Instance** Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.

- The JVM performs following main tasks:
  - Loads code
  - Verifies code
  - Executes code
  - Provides runtime environment
- JVM provides definitions for the:
  - Memory area
  - Class file format
  - Register set
  - Garbage-collected heap
  - Fatal error reporting etc.

1.2.4 JRE

JRE is an acronym for **Java Runtime Environment**. It is used to provide runtime environment. It is the implementation of JVM. It physically exists. It contains set of libraries + other files that JVM uses at runtime.

1.2.5 JSL or API

API stands for application program interface. A programmer writing an application program can make a request to the Operating System using API (using graphical user interface or command**. API in the context of Java**, is a collection of prewritten packages, classes, and interfaces with their respective methods, fields and constructors.The Java API contents many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.

1.3 Concepts of OOPs

Basically Object Orientated Programming is having four pillarsData Abstraction, Encapsulation, Polymorphism and Inheritance which included in Classes and Objects. An object-based application in Java is based on declaring classes, creating objects from them and interacting between these objects.

Object Oriented Programming is a programming concept that works on the principle that objects are the most important part of your program. It allows users create the objects that they want and then create methods to handle those objects. Manipulating these objects to get results is the goal of Object Oriented Programming. Object Oriented Programming popularly known as OOP, is used in a modern programming language like Java.

1.3.1 Object-Oriented Programming is a methodology or paradigm

This is to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:

- Object
- Class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation
- Dynamic Binding
- Messes Passing

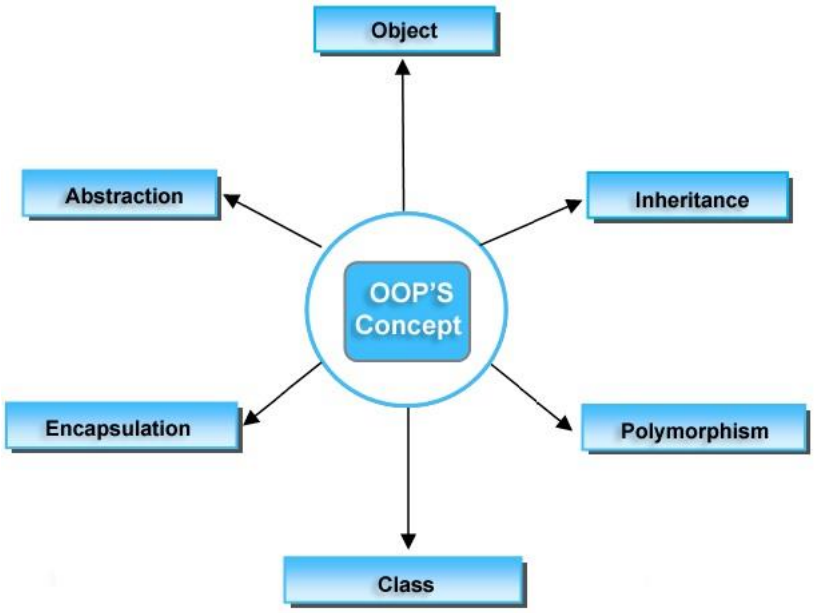


Fig1.5 Basic Concepts of OOPs

- **Object**

  An object can be defined as an instance of a class, and there can be multiple instances of an object in a program. An Object contains both the data and the function, which operates on the data. For example - chair, bike, marker, pen, table, car, etc.

- **Class**

  The class is a group of similar entities. It is only an logical component and not the physical entity. For example, if you had a class called “Expensive Cars” it could have objects like Mercedes, BMW, Toyota, etc. Its properties(data) can be price or speed of these cars. While the methods may be performed with these cars are driving, reverse, braking etc.

- **Inheritance**

  Inheritance is an OOPS concept in which one object acquires the properties and behaviors of the parent object. It’s creating a parent-child relationship between two classes. It offers robust and natural mechanism for organizing and structure of any software.

- **Polymorphism**

  Polymorphism refers to the ability of a variable, object or function to take on multiple forms. For example, in English, the verb “run” has a different meaning if you use it with “a laptop,” “a foot race”, and “business”, Here, we understand the meaning of “run” based on the other words used along with it.The same also applied to Polymorphism.

- **Abstraction**

  An abstraction is an act of representing essential features without including background details. It is a technique of creating a new data type that is suited for a specific application. For example, while driving a car, you do not have to be concerned with its internal working. Here you just need to concern about parts like steering wheel, Gears, accelerator, etc.

- **Encapsulation**

  Encapsulation is an OOP technique of wrapping the data and code. In this OOPS concept, the variables of a class are always hidden from other classes. It can only be accessed using the methods of their current class. For example - in school, a student cannot exist without a class.

- **Dynamic Binding**

  Dynamic binding is an object oriented programming concept and it is related with polymorphism and inheritance. Dynamic binding definition. Dynamic binding(dispatch) means that a block of code executed with reference to a procedure(method) call is determined at run time.

- **Message Passing**

  Message passing is a type of communication between processes. Message passing is a form of communication used in parallel programming and object-oriented programming. Communications are completed by the sending of messages (functions, signals and data packets) to recipients.

1.3.2**Advantages of OOPS:**

- OOP offers easy to understand and a clear modular structure for programs.

- Objects created for Object-Oriented Programs can be reused in other programs. Thus it saves significant development cost.
- Large programs are difficult to write, but if the development and designing team follow OOPS concept then they can better design with minimum flaws. So it is Suitable for large projects
- It also enhances program modularity because every object exists independently.
- Better memory management
- Fairly efficient languages
- It implements real life scenario
- It is easy to maintain and modify existing code
- Implementation details are hidden from other modules

1.3.3 Areas of Application of OOP concept:

The promising areas includes the followings,

1. Real Time Systems Design
2. Simulation and Modeling System
3. Object Oriented Database
4. Object Oriented Distributed Database
5. Client-Server System
6. Hypertext, Hypermedia
7. Neural Networking and Parallel Programming
8. Decision Support and Office Automation Systems

1.3.4 Difference between Procedure Oriented Programming (POP) and Object Oriented Programming (OOP)

Sr. no.	*Subject of Difference*	Procedure Oriented Programming (POP)	Object Oriented Programming (OOP)
1	Problem decomposition	Decompose the main problem in small parts called functions.	Decompose the main problem in small parts called objects.
2	Connections of parts	Connects small parts of the program by passing parameters & using operating system.	Connects small parts of the program by passing messages.
3	Emphasizing	Emphasizes on functions.	Emphasizes on data.
4	Use of data	In large programs, most functions use global data.	Each object controls data under it.
5	Passing of data	Data may get passed from one function to another.	Data never get passed from one object to another.
6	Security of data	Appropriate & effective techniques are unavailable to secure the data.	Data stay secured as no external function can use data of an object.
7	Modification of program	Modification of a completed program is very difficult and it may affect the	Modifications are easy as objects stay independent to declare and

		whole program.	define.
8	Designing approach	Employs top-down approach for designing programs.	Employs bottom-up approach for designing.
9	Data identification	In large programs, it is very difficult to find what data has been used by which function.	As data and functions stay close, it is easy to identify data.
10	Used languages	Languages like C, FORTRAN, COBOL etc. use POP.	Languages like C++, JAVA etc. use OOP.

Table1.2: Difference Between POP and OOP

1.4 Features of Java

The main objective of Java programming language creation was to make it portable, simple and secure programming language. Apart from this, there are also some awesome features which play important role in the popularity of this language. The features of Java are also known as java buzzwords. Following is a list of most important features of Java language. The Java Features given below are simple and easy to understand.Features of a language are nothing but the set of services or facilities provided by the language vendors to the industry programmers. Some important **features of java** are;

Important Features of Java

- Simple
- Object-Oriented
- Portable
- Platform independent
- Secured
- Robust
- Architecture neutral
- Dynamic
- Compiled Interpreted
- High Performance
- Multithreaded
- Distributed

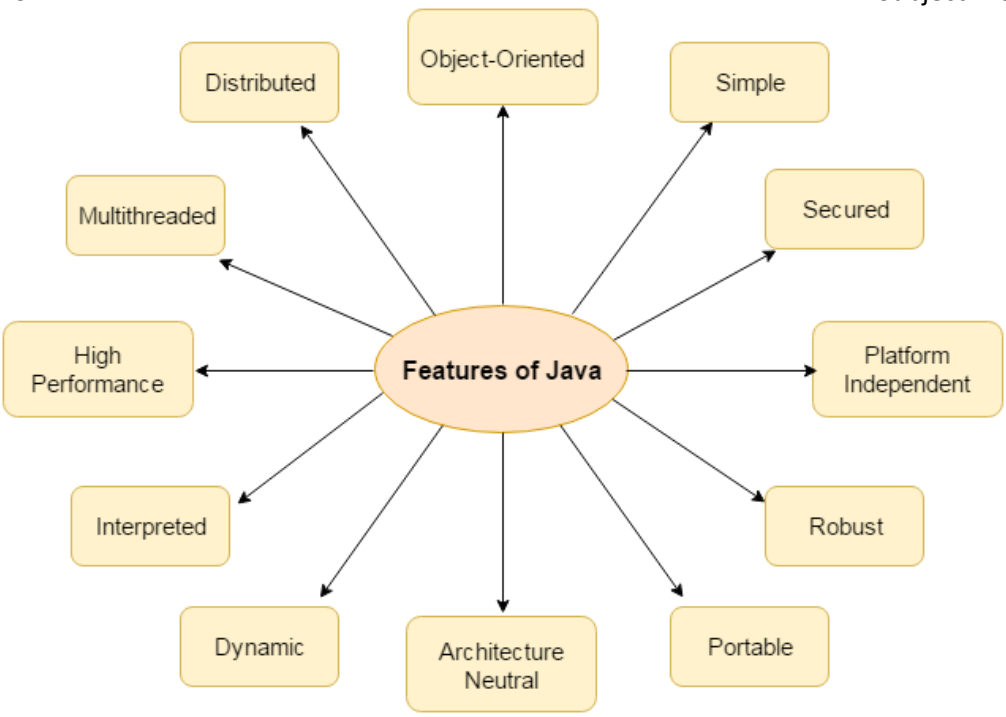


Fig1.6 Features of Java

- **Simple**

  It is simple because of the following factors:

  1. Looks familiar to existing programmers: related to C and C++:
  2. Omits many rarely used, poorly understood, confusing features of C++, like operator overloading, multiple inheritance, automatic coercions, etc.
  3. It does not Contains goto statement, but break and continue is available in java.
  4. Has no header files and eliminated C preprocessor.
  5. Eliminates much redundancy (e.g. no structs, unions, or functions)
  6. It is **free from pointer** due to this execution time of application is improved. [Whenever we write a Java program without pointers then internally it is converted into the equivalent pointer program].
  7. It has **Rich set of API** (application protocol interface).
  8. It has **Garbage Collector** which is always used to collect un-Referenced (unused) Memory location for improving performance of a Java program
  9. It contains user friendly syntax for developing any applications.

- **Object-Oriented**

  Java is 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.It supports OOP's conceptsbecause of this it is most secure language, for this topic you have read our OOPs concepts in detail.

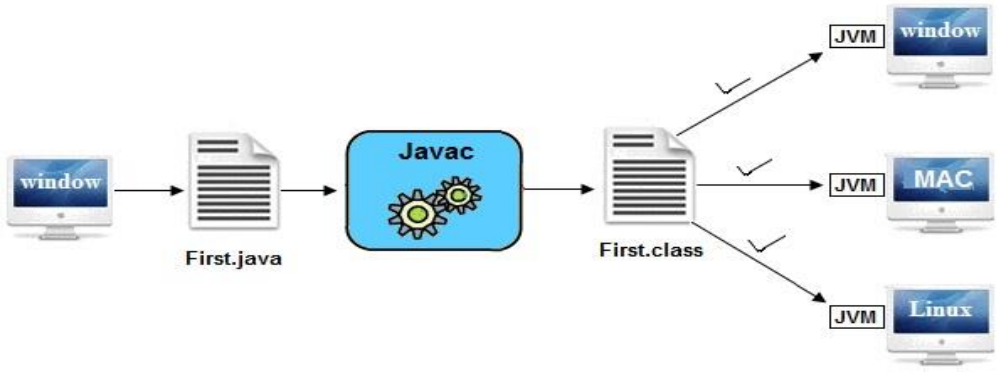
- **Portable**

  If any language supports platform independent and architectural neutral feature known as portable. The languages like C, CPP, Pascal are treated as non-portable

language. Java is portable because it facilitates you to carry the java bytecode (.class file) to any platform.

- **Platform Independent**

A program or technology is said to be platform independent if and only if which can run on all available operating systems with respect to its development and compilation. (Platform represents O.S). Java is platform independent because it is different from other languages like C, C++ etc. which are compiled into platform specific machines while Java is a write once, run anywhere language. A platform is the hardware or software environment in which a program runs.



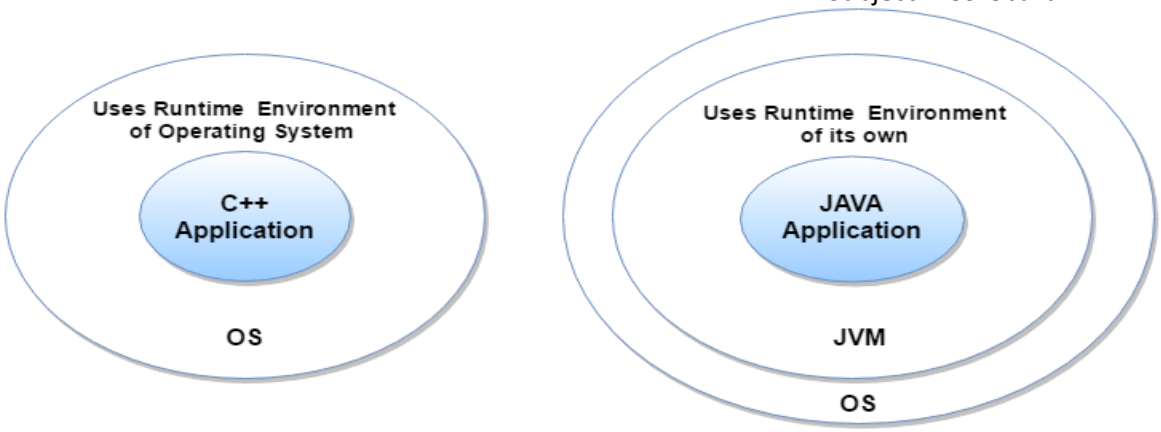
There are two types of platforms **software-based and hardware-based.** Java provides software-based platform. The Java platform differs from most other platforms in the sense that it is a software-based platform that runs on the top of other hardware-based platforms. It has two components:

1. Runtime Environment
2. API(Application Programming Interface)

Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, Mac/OS etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere(WORA).

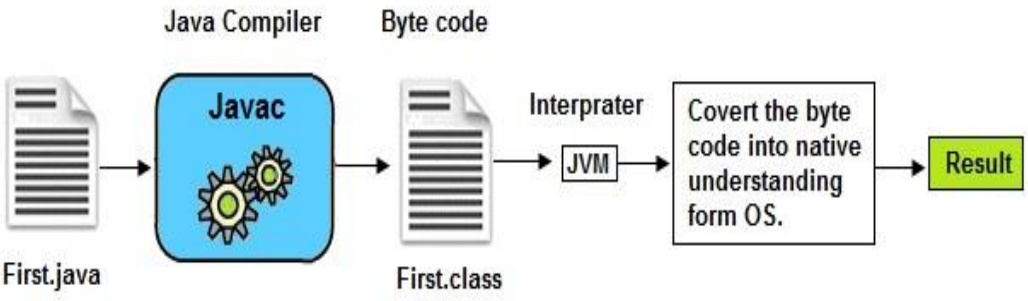
- **Secured**

Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:**No explicit pointer,Java Programs run inside JVM.** In this language, all code is covered in byte code after compilation which is not readable by human.JVM is an interpreter which is installed in each client machine that is updated with latest security updates by internet. When this byte codes are executed, the JVM can take care of the security. So, **java** is said to be more **secure** than other **programming languages**



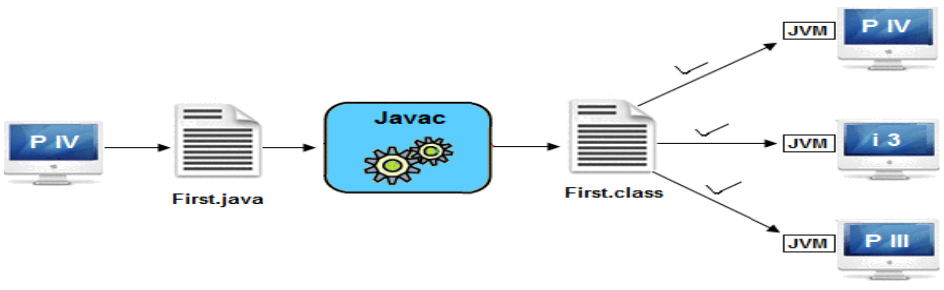
- **Robust**

Java is called as robust because It is portable across many Operating systems.It uses strong memory management. It doesn't have the concept of pointersso that we can avoids the security problem. There is automatic garbage collection in java. Also it is support the exception handling and type checking mechanism in so that Bugs, especially system crashing bugs, are very rare in Java. All these points make java robust



- **Architecture neutral**

1. A Language or Technology is said to be Architectural neutral which can run on any available processors in the real world without considering their development and compilation. compiler generates bytecodes, which have nothing to do with a particular computer architecture easy to interpret on any machine.
2. It is a



Software that is designed without regard to the target platform. Software is often written to maximize the performance of a specific hardware platform, but such software must be modified to make it run on other hardware.

3. Java is architecture neutral because there is no implementation dependent features e.g. size of primitive types is fixed.
4. In C programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. But in java, it occupies 4 bytes of memory for both 32 and 64 bit architectures.

- **Dynamic**

  It supports Dynamic memory allocation due to this memory wastage is reduce and improve performance of the application. The process of allocating the memory space to the input of the program at a run-time is known as dynamic memory allocation, to programming to allocate memory space by dynamically we use an operator called 'new'. A 'new' operator is known as dynamic memory allocation operator.

- **Compiled Interpreted**

  Unlike most of the programming languages which are either complied or interpreted, Java is both complied and interpreted. The Java compiler translates a java source file to bytecodes and the Java interpreter executes the translated byte codes directly on the system that implements the Java Virtual Machine. These two steps of compilation and interpretation allow extensive code checking and improved security.Java byte-codes provide an architecture-neutral object file format. The code is designed to transport programs efficiently to multiple platforms.

- **High Performance**

  Java is an interpreted language, so it will never be as fast as a compiled language as C or C++. In fact, it is about 20 times as slow as C. However, this speed is more than enough to run interactive, GUI and network-based applications, where the application is often idle, waiting for the user to do something, or waiting for data from the network. Although in the early releases of Java, the interpretation of by bytecode resulted in slow performance but the advance version of JVM uses the adaptive and Just in time (JIT) compilation technique that improves performance by converting Java bytecodes to native machine instructions on the fly.

- **Multithreaded**

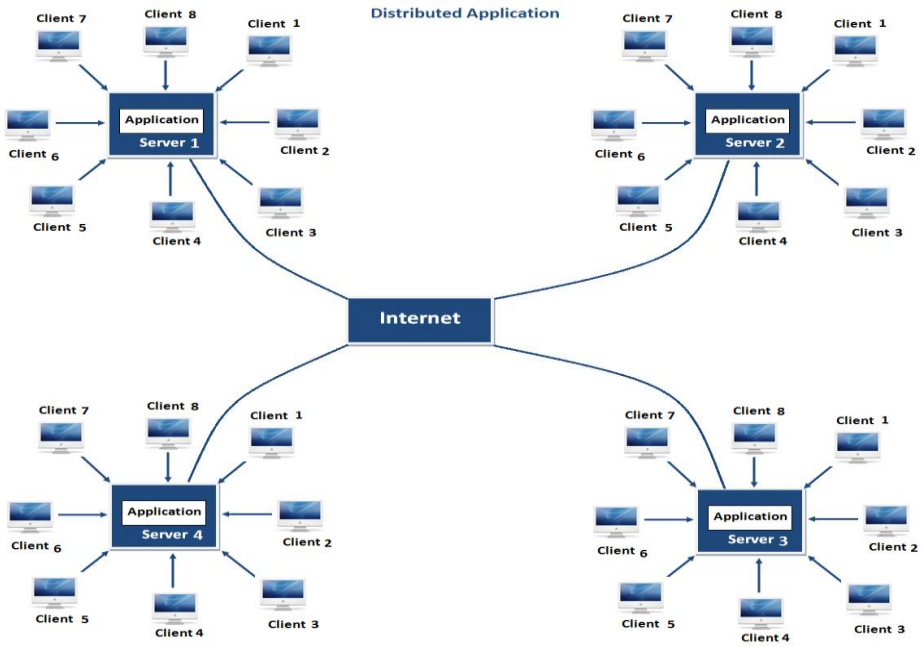
  Java supports Multithreading Programming. Multithreading means handling more than one job at a time like multitasking, so get more process get done in less time than it could with just one thread.A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area. Threads are important for multi-media, Web applications etc.

- **Distributed**

  Java is distributed because it facilitates us to create distributed applications in java. Using this language, we can create distributed applications. In distributed application multiple client system depends on multiple server systems so that even

problem occurred in one server will never be reflected on any client system. Java was designed with the distributed environment.

It has networking facilities, so it can be transmitting, run over internet. RMI and EJB are used for creating distributed applications. We may access files by calling the methods from any machine on the internet. Java is a distributed language which means that the program can be design to run on computer networks. Java provides an extensive library of classes for communicating ,using TCP/IP protocols such as HTTP and FTP. This makes creating network connections much easier than in C/C++. You can read and write objects on the remote sites via URL with the same ease that programmers are used to when read and write data from and to a file. This helps the programmers at remote locations to work together on the same project.



1.5 Application of Java

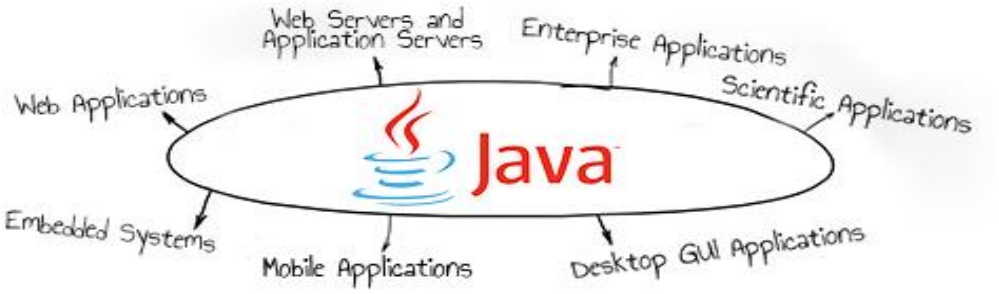


Fig1.7 Application of Java

1.5.1 **Desktop GUI Applications**:

Java provides GUI development through various means like Abstract Windowing Toolkit (AWT), Swing and JavaFX. While AWT contains a number of pre-constructed components such as menu, button, list, and numerous third-party components, Swing, a GUI widget toolkit, additionally provides certain advanced components like trees, tables, scroll panes, tabbed panel and lists. JavaFX, a set of graphics and media packages, provides Swing interoperability, 3D graphic features and self-contained deployment model which facilitates quick scripting of Java applets and applications.

1.5.2**Mobile Applications:**

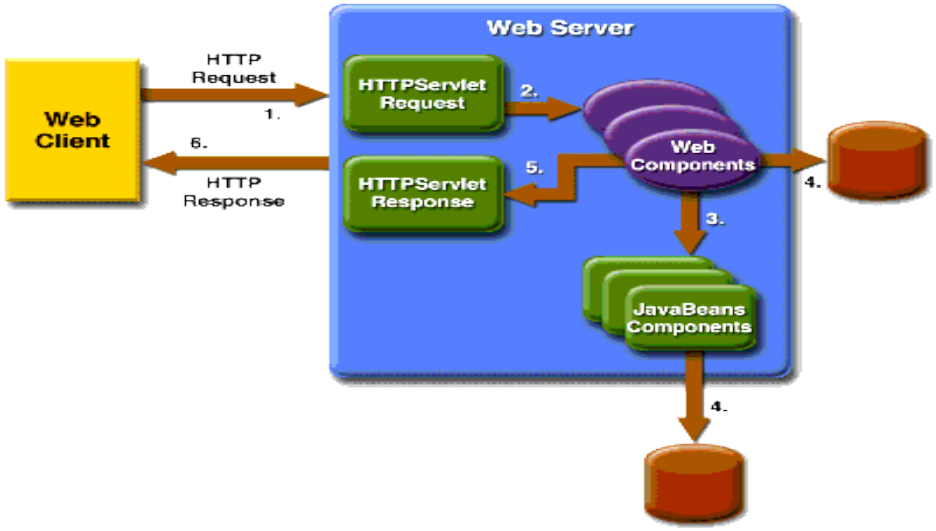
Java Platform, Micro Edition (Java ME or J2ME) is a cross-platform framework to build applications that run across all Java supported devices, including feature phones and smart phones. Further, applications for Android, one of the most popular mobile operating systems, are usually scripted in Java using the Android Software Development Kit (SDK) or other environments.

1.5.3 **Embedded Systems:**

Embedded systems, ranging from tiny chips to specialized computers, are components of larger electromechanical systems performing dedicated tasks. Several devices, such as SIM cards, blue-ray disk players, utility meters and televisions, use embedded Java technologies. According to Oracle, 100% of Blu-ray Disc Players and 125 million TV devices employ Java.

1.5.4 **Web Applications:**

Java provides support for web applications through Servlets, Struts or JSPs. The easy programming and higher security offered by the programming language has allowed a large number of government applications for health, social security, education and insurance to be based on Java. Java also finds application in development of eCommerce web applications using open-source eCommerce platforms, such as Broadleaf.

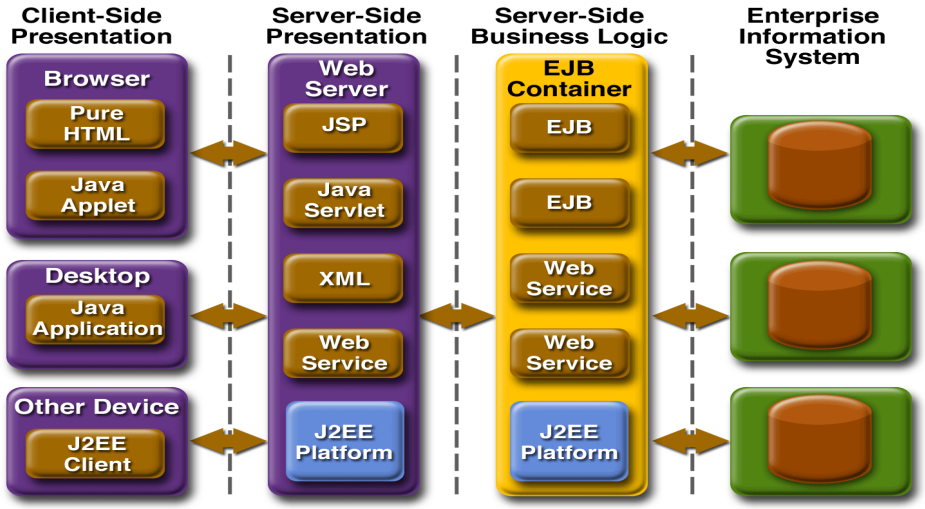


1.5.5 **Web Servers and Application Servers:**

The Java ecosystem today contains multiple Java web servers and application servers. While Apache Tomcat, Simple, Jo!, Rimfaxe Web Server (RWS) and Project Jigsaw dominate the web server space, WebLogic, WebSphere, and Jboss EAP dominate commercial application server space.

1.5.6 **Enterprise Applications:**

Java Enterprise Edition (Java EE) is a popular platform that provides API and runtime environment for scripting and running enterprise software, including network applications and web-services. Oracle claims Java is running in 97% of enterprise computers. The higher performance guarantees and faster computing in Java has resulted in high frequency trading systems like Murex to be scripted in the language. It is also the backbone for a variety of banking applications which have Java running from front user end to back server end.



1.5.7**Scientific Applications:**

Java is the choice of many software developers for writing applications involving scientific calculations and mathematical operations. These programs are generally considered to be fast and secure, have a higher degree of portability and low maintenance. Applications like MATLAB use Java both for interacting user interface and as part of the core system.

1.6 How Java differ from C++

There are many differences and similarities between C++ programming language and Java. A list of top differences between C++ and Java are given below:

Sr. No.	Java	C++
1	Java is platform-independent.	C++ is platform-dependent.
2	Java is mainly used for application programming. It is widely used in window, web-based, enterprise and mobile applications.	C++ is mainly used for system programming.
3	Java doesn't support goto statement.	C++ supports goto statement.
4	Java doesn't support multiple inheritance through class. It can be achieved by interfaces in java.	C++ supports multiple inheritance.

5	Java doesn't support operator overloading.	C++ supports operator overloading.
6	Java supports pointer internally. But you can't write the pointer program in java. It means java has restricted pointer support in java.	C++ supports pointers. You can write pointer program in C++.
7	Java uses compiler and interpreter both.	C++ uses compiler only.
8	Java supports call by value only. There is no call by reference in java.	C++ supports both call by value and call by reference.
9	Java doesn't support structures and unions.	C++ supports structures and unions.
10	Java has built-in thread support.	C++ doesn't have built-in support for threads. It relies on third-party libraries for thread support.
11	Java supports documentation comment (/** ... */) to create documentation for java source code.	C++ doesn't support documentation comment.
12	Java has no virtual keyword. We can override all non-static methods by default. In other words, non-static methods are virtual by default.	C++ supports virtual keyword so that we can decide whether or not override a function.
13	Java supports unsigned right shift >>> operator that fills zero at the top for the negative numbers. For positive numbers, it works same like >> operator.	C++ doesn't support >>> operator.
14	Java uses single inheritance tree always because all classes are the child of Object class in java. Object class is the root of inheritance tree in java.	C++ creates a new inheritance tree always.

Table1.3: Difference Between Java and C++