Date : 13th feb 2024

## **What is Java?**

* Developed By James Gosling at Sun Microsystems in 1995
* It is a Programming language and a Platform.
* Platform: any hardware or software env. Where program runs.
* Since Java has JRE (Java runtime env) and API . so it is a platform

##### **Types of Java Applications :**

1. **Standalone application :**

* Also known as Desktop app or window based app
* Traditional software that we need to install on our system
* Ex. : media player, antivirus

1. **Web application**

* Application that runs on server side and created dynamic page
* Ex. Servlets , JSP , Spring

1. **Enterprise Application**

* Application that is distributed in nature such as bank applications

1. **Mobile Application**

* Application created for mobile device
* Android and JAVA ME are used to create mobile applications

##### **Java Platforms / Editions :**

1. **Java SE (Java standard edition) :**

* Include java programming APIs such as java.lang, java.io, java.net , java.util etc
* Includes core topics like OOPs, String . Regex , Exception

1. **Java EE (Enterprise edition) :**

* Used to develop web and enterprise applications
* Build on top of Java SE platform
* Includes topics like Servlets , JSP, Web services etc.

1. **Java ME (Micro edition) :**

* Micro platform that is dedicated to mobile applications

1. **Java FX :**

* Used to develop rich internet applications
* Uses lightweight user interface API

## Features of Java :

1. **Simple :**

* Easy to learn
* Simple syntax . based on C++
* Clean and easy to understand
* Removed many complicated features like - explicit pointers and operator overloading
* Have automatic garbage collector

1. **Object-Oriented :**

* Object-oriented programming language i.e everything is an object
* Object-oriented means we organize our software as a combination of different types of objects that incorporate 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:
  + Object
  + Class
  + Inheritance
  + Polymorphism
  + Abstraction
  + Encapsulation

1. **Portable :**

* Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

1. **Platform independent :**

* Java is write once , run anywhere so it is not platform dependent
* C, C++ are complied into platform specific machines
* Java is software-based platform which runs on top of other hardware based platforms .
* Java has two components :
  + Runtime env.
  + API
* Java code is complied by compiler and converted into bytecode . this bytecode is platform-independent code

1. **Secured :**

* Java is best known for security
* Java can be used to developed virus free system
* Java is secured by bcz :
  + No explicit pointer
  + Java Programs run inside a virtual machine sandbox
  + **Classloader:** Classloader in Java is a part of the Java Runtime Environment (JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.
  + **Bytecode Verifier:** It checks the code fragments for illegal code that can violate access rights to objects.
  + **Security Manager:** It determines what resources a class can access such as reading and writing to the local disk.

1. **Robust**

* Robust means strong
* It is robust bcz :
  + Uses string memory management
  + Lack of pointers
  + Automatic garbage collection which runs on JVM
  + Exception handling
  + Type checking mechanism

1. **Architecture neutral :**

* There is no implementation-dependent feature .
* Ex: size of primitive type is fixed
* In c int occupies 2 bytes incase of 32 bit architecture and 4 bytes in case of 64 bit architecture but Java occupies 4 bytes for both 32 and 64 bit

1. **Interpreted**
2. **High Performance :**

* Java is faster as java bytecode is close to native code
* Still little slower than compiler language like C++
* Since java is an interpreted language so it is slower than the compiler language

1. **Multithreaded :**

* Thread is 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.

1. **Distributed :**

* Distributed because it facilitates user to create distributed applications
* RMI and EJB is used to crete distributed applications

1. **Dynamic :**

* Java supports dynamic loading of classes i.e clases are loaded on demand
* Java supports dynamic compilation and automatic memory management (garbage collection).

## **C++ VS Java**

| **Comparison Index** | **C++** | **Java** |
| --- | --- | --- |
| Platform independent | Platform dependent | Platform independent |
| Used for | For system programming | Application programming, window based , web-based, enterprise , mobile applications |
| Goto statement | Support Goto statement | Does not support Goto |
| Multiple inheritance | Supports Multiple inheritance | Does not support Multiple Inheritance |
| Operator overloading | Supports Operator overloading | Doesnot supports operator overloading |
| Pointers | Supports Pointers | Supports pointers internally . but cant write pointers in program explicitly Java restrict pointer support |
| Compiler & interpreter | Uses compiler only | Uses both compiler and interpreter |
| Flow | C++ is compiled and run using the compiler which converts source cod into machine code so it is platform dependent | Java source code is converted into bytecode at compilation time . The interpreter executes this bytecode at runtime and produces output. It is interpreted that is why it is platform-independent |
| Call by value and call by reference | Supports both | Supports only call by value |
| Structure and Union | Supports both | Supports NOne |
| Thread Support | Relies on Third-party libraries for thread support | Have build in thread support |
| Multi-comment / Documentation comment | Does not support | Support documentation comment (/\* \*/) |
| Virtual keyword | Supports virtual keyword so that we can decide whether or not to override a function | Has no virtual keyword . We can override all non static method by default , in other word , non-static methods are virtual by default |

## **First Java Program | Hello World Example**

##### 