JAVA

**HISTORY AND INTRODUCTION TO JAVA**

**History**

Java is related to C++, which is a direct descendent of C Much of the features of Java are inherited from these two languages. Each innovation in language design was driven by the need to solve a fundamental problem that the preceding languages could not solve. Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991. This language was initially called "Oak" but was renamed "Java" in 1995.

The primary motivation was the need for a platform independent language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls. Java derives much of its character from C and C++. This is by intent. The Java designers knew that using the familiar syntax of C and echoing the object oriented features of C++ would make their language appealing to the C/C++ programmers.

**Introduction to Java**

Java is a modern, evolutionary computing language that combines an elegant language design with powerful features that were previously available primarily in specialty languages. In addition to the core language components, Java software Distributions include many powerful, supporting software libraries for tasks such as database, network, and graphical user interface(GUI) programming. In this chapter, we focus on the core Java language features.

Java is a true object-oriented (OO) programming language.

The main implication of this statement is that in order to write programs with Java, you must work within its object-oriented structure. Object-oriented languages provide a framework for designing programs that represent real-world entities such as cars,

employees, insurance policies, and so on. Representing real-world entities with non object-oriented languages is difficult because it's necessary to describe entities such as a truck with rather primitive

language constructs such as Pascal's record, C's struct, and others that represent data only.

The behaviour of an entity must be handled separately with language constructs such as procedures and/or functions, hence, the term procedural programming languages. Given this separation,

the programmer must manually associate a data structure with the appropriate procedures that operate on, that is, manipulate, the data.

Java can be used to create two types of programs: applications and applets. An application is a program that runs on your computer, under the operating system of that computer. An applet is an application designed to be transmitted over the Internet and executed by a Java-compatible Web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image, sound file, or video clip.

**JAVA FEATURES**

The Java programming language is a high-level language that can be characterized by:

**Simple -** Java was designed to be easy for the professional programmer to learn and use effectively. Because Java inherits the C/C++ syntax and many of the object-oriented features of

C++, most programmers have little trouble learning Java.

**Secure -** Java does not use memory pointers explicitly. All the programs in java are run under an area known as the sand box. Security manager determines the accessibility options of a class like reading and writing a file to the local disk.

**Portable -** The feature Write-once-run-anywhere makes the java language portable provided that the system must have interpreter for the JVM. Java also has the standard data size irrespective of operating system or the processor. These features make the java as a portable language.

**Object-oriented -** The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high performance non-objects. It is a fully Object Oriented language

becaue object is at the outer most level of data structure in java. No stand alone methods, constants, and variables are there in java. Everything in java is object even the primitive data

types can also be converted into object by using the wrapper class.

**Robust -** Java restricts you in a few key areas, to force you to find your mistakes early in program development. At the same time, Java frees you from having to worry about many of the most common causes of programming errors. Java has the strong memory allocation and automatic garbage collection mechanism. It provides the powerful exception handling and type checking mechanism as compare to other programming languages. All of the above features make the java language robust.

**Multithreaded -** Java supports multithreaded programming, which allows you to write programs that do many things simultaneously. Multiple threads execute instructions according to the program code in a process or a program. Multithreading works the similar way as multiple processes run on one computer.

**Architecture-neutral -** The Java compiler supports this feature by generating byte code instructions, to be easily interpreted on any machine and to be easily translated into native machine code on the fly. The compiler generates an architecture-neutral object file format to enable a Java application to execute anywhere on the network and then the compiled code is executed on many processors, given the presence of the Java runtime system.

 **Interpreted -** Java enables the creation of cross-platform programs by compiling into an intermediate representation called Java byte code. This code can be interpreted on any system that provides a Java Virtual Machine.

**High performance -** Java uses native code usage, and lightweight process called threads. In the beginning interpretation of byte code resulted the performance slow but the advance version of JVM uses the adaptive and just in time compilation technique that improves the performance.

**Distributed -** Java is designed for the distributed environment of the Internet, because it handles TCP/IP protocols. Internet programmers can call functions on these protocols and can get access the files from any remote machine on the internet rather than writing codes on their local system.

**Dynamic -** Java programs carry with them substantial amounts of run-time type information that is used to verify and resolve accesses to objects at run time. While executing the java program the user can get the required files dynamically from a local drive or from a computer thousands of miles away from the user just by connecting with the Internet.

**DIFFERENT TYPES OF JAVA PROGRAMS**

Java is a programming language that’s used to build programs that can work on the local machine and on the internet as well. So there are various categories of programs that can be developed in Java.

 **STAND-ALONE APPLICATIONS -** Console Applications – An application is a program that runs on the computer under the operating system of your computer. Creating an application in

java is similar to doing so in any other computer language. The application can either be **GUI based** or **console based**.

 **WEB APPLICATIONS -** These are the applications which are web-based in nature and require a web browser for excecution. The Web applications makes use of a Server to store the data and every time a user requests to execute that application, the request is passed on to the server for suitable reply. E.g. Applet and Servlet. Applets are Java programs that are created specially to work on the internet. In Servlets, the client sends a request to a server. The server processes the request and

sends a response back to the client.

**DISTRIBUTED APPLICATIONS -** It requires a server to run these applications. A number of servers are used simultaneously for backup to prevent any data losses.

**CLIENT SERVER APPLICATIONS -** These applications too make use of web technology for their execution. They follow simple Client-Server model, where a client makes requests directly to the server.

**DIFFERENTIATE JAVA WITH C AND C++**

**Major differences between C and JAVA are**

* JAVA is Object-Oriented while C is procedural – Most differences between the features of the two languages arise due to the use of different programming paradigms. C is more procedure-oriented while JAVA is data-oriented.
* Java is an Interpreted language while C is a compiled language-
* A C compiler takes your code & translates it into something the machine can understand. While with JAVA, the code is first transformed to what is called the bytecode. This bytecode is then executed by the JVM(Java Virtual Machine). For the same reason, JAVA code is more portable.
* C is a low-level language while JAVA is a high-level language.
* C uses the top-down approach while JAVA uses the bottom-up approach - In C, formulating the program begins by defining the whole and then splitting them into smaller elements. JAVA follows the bottom-up approach where the smaller elements together to form the whole.
* Pointer go backstage in JAVA while C requires explicit handling of pointers - When it comes to JAVA, we don't need the \*'s & &'s to deal with pointers & their addressing. More formally, there is no pointer syntax required in JAVA. It does what it needs to do.
* While in JAVA, we do create references for objects.
* JAVA supports Method Overloading while C does not support overloading at all - JAVA supports function or method overloading-that is we can have two or more functions with the same name.
* The standard Input & Output Functions - Although this difference might not hold any conceptual significance, but it's maybe just the tradition. C uses the printf & scanf functions as its standard input & output while JAVA uses the System. out. print & System. In read functions.
* Exception Handling in JAVA and the errors & crashes in C -
* When an error occurs in a Java program it results in an exception being thrown. It can then be handled using various exception handling techniques. While in C, if there's an error, there is an error.

**Major differences between C++ and JAVA are**

* C++ was mainly designed for systems programming and Java was created initially to support network computing.
* C++ supports pointers whereas Java does not pointers.
* At compilation time Java Source code converts into byte code
* .The interpreter execute this byte code at run time and gives output. C++ run and compile using compiler which converts source code into machine level languages so C++ is plat from dependents
* Java is platform independent language but C++ is depends upon operating system machine etc.
* Java uses compiler and interpreter both and in C++ their is only compiler
* C++ supports operator overloading multiple inheritance but java does not.
* Java does is a similar to C++ but not have all the complicated aspects of C++ (ex: Pointers, templates, unions, operator overloading, structures etc..)
* Thread support is built-in Java but not in C++.
* Internet support is built-in Java but not in C++.
* Java does not support header file, include library files just like
* C++ .Java use import to include different Classes and methods.
* Java does not support default arguments like C++.
* Exception and Auto Garbage Collector handling in Java is different because there are no destructors into Java.
* Java has method overloading, but no operator overloading just like C++.

**SAMPLE JAVA PROGRAM**

Let us start Java programming with a small example. This program will show the output “Hello World”

class HelloWorld

{

public static void main(String args[])

{

System.out.println(“Hello World”);

}

}

Now let us understand the program line by line.

* **Class Declaration** – This line declares a class. class is an object-oriented construct and a keyword which states that the class declaration follows. HelloWorld is the name of the class.
* **Opening and Closing Brace** - The entire class definition, including all of its members, will be between the opening curly brace ({) and the closing curly brace (}). The use of the curly braces in Java is identical to the way they are used in C and
* C++.
* **Main Line** – The main() function is similar to the the main() in C/C++. Every Java application program must include the main() method. The keyword **public** is an access specifier that declares the main method accessible to all other classes. The next keyword static states that this method belongs to the entire class. The keyword **static** allows main() to be called without having to instantiate a particular instance of the class. main() is method called when a Java application begins. In main(), is only one parameter, String args[ ] declares a parameter named **args**, which is an array of instances of the class String. The type modifier **void** states that the main() method does not
* return any value.
* **Output Line** – This line is similar to the printf() of C or cout << of C++. The println() method is a member of the out object, which is a static data member of System class. System is a predefined class that provides access to the system, and out is the output stream that is connected to the console.

**Output:**

Hello World.