Java - Features

Java programming language was initially developed to work on embedded systems, settop boxes, television. So by requirements, it was initially designed to work on varied platforms. Over the period of multiple years, Java evolved to become one of the most popular language used to develop internet based applications.

Java is a feature rich language and with every new version, it is continously evolving. It is widely used across billions of devices. Following are the main features of the Java language -

- Object Oriented
- Platform Independent
- Simple
- Secure
- Architecture-neutral
- Portable
- Robust
- Multithreaded
- Interpreted
- High Performance
- Distributed
- Dynamic

Object Oriented

In Java, everything is an Object. Java can be easily extended since it is based on the Object model. As a language that has the Object-Oriented feature, Java supports the following fundamental concepts of OOPs –

- Polymorphism
- Inheritance
- Encapsulation
- Abstraction
- Classes
- Objects
- Instance
- Method

- Message Passing

Platform Independent

Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Java Virtual Machine (JVM) on whichever platform it is being run on.

Java is designed in **Write Once, Run Anywhere** (WORA) way. Code written in Java is not directly dependent on the type of machine it is running. A code is Java is compiled in ByteCode which is platform independent. Java Virtual Machine, JVM can understand the byte code. Java provides platform specific JVMs. It is the responsibility of platform specific JVM to interpret the byte code correctly thus developers are free to write code without worrying about platforms like windows, linux, unix, Mac etc. This feature makes Java a platform neutral language.

As byte code can be distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on. It makes java code highly portable and useful for application running on multiple platforms.

Learn **Java** in-depth with real-world projects through our **Java certification course**. Enroll and become a certified expert to boost your career.

Simple

Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.

Java is very easy to learn. It inherits many features from C, C++ and removes complex features like pointers, operator overloading, multiple inheritance, explicit memory allocation etc. It provides automatic garbage collection. With a rich set of libraries with thousands of useful functions, Java makes developers life easy.

Secure

With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.

Java is by design highly secure as it is not asking developers to interact with underlying system memory or operation system. Bytecode is secure and several security flaws like buffer overflow, memory leak are very rare. Java exception handling mechanism allows developers to handle almost all type of error/exceptions which can happen during program execution. Automatic garbage collection helps in maintaining the system memory space utilization in check.

Architecture-neutral

Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.

Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system. With advancement in processor architectures or machine specific processors, java code remains independent of any specific requirement of a processor. As java is an open standard, even a specific JVM can be prepared for a custom architecture. As in today's time, we've JVM available for almost all popular platforms, architectures, java code is completely independent. For example, a java program created in Windows machine can run on linux machine without any code modification.

Portable

Being architecture-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary, which is a POSIX subset.

Due to this portability, java was an instant hit since inception. It was particulary useful for internet based application where platforms varied from place to place and same code base can be used across multiple platform. So collaboration between developers was easy across multiple locations.

Robust

Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking. Automatic garbage collection, strong memory management, no pointers, no direct access to system memory, exception handling, error handling are some of the key features which makes Java a Robust, strong language to rely on.

Multithreaded

With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.

A multi-threaded program contains two or more parts that can run concurrently and each part can handle a different task at the same time making optimal use of the available resources specially when your computer has multiple CPUs.

By definition, multitasking is when multiple processes share common processing resources such as a CPU. Multithreading extends the idea of multitasking into applications where you can subdivide specific operations within a single application into individual threads. Each of

the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application.

Multi-threading enables you to write in a way where multiple activities can proceed concurrently in the same program.

Interpreted

Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.

JVM sits in between the javac compiler and the underlying hardware, the javac (or any other compiler) compiler compiles Java code in the Bytecode, which is understood by a platform specific JVM. The JVM then compiles the Bytecode in binary using JIT (Just-in-time) compilation, as the code executes.

High Performance

With the use of Just-In-Time compilers, Java enables high performance. JVM uses JIT compiler to improves the execution time of the program. Below are some general optimizations that are done by the JIT compilers

- Method inlining
- Dead code elimination
- Heuristics for optimizing call sites
- Constant folding

Distributed

Java is designed for the distributed environment of the internet.

Dynamic

Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment. Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.