# **Java**

# Java was developed by ***Sun Microsystems*** (which is now the subsidiary of **Oracle**) in the year **1995**. ***James Gosling*** is known as the father of Java. Before Java, its name was *Oak*. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.

# Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented , portable, simple and secure programming language.

# **Simple**

# Java is very easy to learn, and its syntax is simple, clean and easy to understand.

# Java has **removed** many complicated and **rarely-used features**, for example, explicit **pointers**, operator overloading, etc.

# There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

# Java is an [object-oriented](https://www.javatpoint.com/java-oops-concepts) programming language. Everything in Java 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:

1. [Object](https://www.javatpoint.com/object-and-class-in-java)
2. [Class](https://www.javatpoint.com/object-and-class-in-java#class)
3. [Inheritance](https://www.javatpoint.com/inheritance-in-java)
4. [Polymorphism](https://www.javatpoint.com/runtime-polymorphism-in-java)
5. [Abstraction](https://www.javatpoint.com/abstract-class-in-java)
6. [Encapsulation](https://www.javatpoint.com/encapsulation)

### 3. Platform Independent

### The design objective of javasoft people is to develop a language that must work on any platform. Here platform means a type of operating system and hardware technology. Java allows programmers to write their program on any machine with any configuration and to execute it on any other machine having different configurations. In Java, Java source code is compiled to bytecode and this bytecode is not bound to any platform. In fact, this bytecode is only understandable by the Java Virtual Machine which is installed in our system. What I meant to say is that every operating system has its own version of JVM, which is capable of reading and converting bytecode to an equivalent machine’s native language. This reduces the overhead of programmers writing system-specific code. Now programmers write programs only once, compile them, generate the bytecode and then export it anywhere. 4.

### Portable

The WORA (Write Once Run Anywhere) concept and platform-independent feature make Java portable. Now using the Java programming language, developers can yield the same result on any machine, by writing code only once. The reason behind this is JVM and **bytecode**. Suppose you wrote any code in Java, then that code is first converted to equivalent bytecode which is only readable by JVM. We have different versions of JVM for different platforms. Windows machines have their own version of JVM, Linux has its own and macOS has its own version of JVM. So if you distribute your bytecode to any machine, the JVM of that machine would translate the bytecode into the respective machine code. In this way portability lets the programmers focus on development and productivity rather than writing different code for different platforms.

**5. Robust**

The Java Programming language is robust, which means it is capable of handling **unexpected termination** of a program. There are 2 reasons behind this, first, it has a most important and helpful feature called **Exception** **Handling**. If an exception occurs in java code then no harm will happen whereas, in other low-level languages, the program will crash.

Another reason why Java is strong lies in its memory management features. Unlike other low-level languages, Java provides a **runtime Garbage collector offered** by JVM, which collects all the unused variables. The garbage collector is a special program under JVM that runs from time to time and detects any **unused variables and objects and removes them from the memory to free up space.** But in the case of other prior languages, there is no such program to handle memory management, programmers are solely responsible for allocating and deallocating memory spaces, otherwise, the program may crash due to insufficient memory

**6. Secure**

Encryption and Decryption feature to secure your data from *eavesdropping*and *tampering*over the internet. An *Impersonation*is an act of pretending to be another person on the internet.

### 7. Interpreted

In programming languages, you have learned that they use either the compiler or an interpreter, but Java programming language uses both a compiler and an interpreter. Java programs are compiled to generate bytecode files then JVM interprets the bytecode file during execution. Along with this JVM also uses a JIT compiler (it increases the speed of execution).

### 8. Multi-Threaded

Thread is a lightweight and independent subprocess of a running program (i.e, process) that shares resources.

Thread is a lightweight and independent subprocess of a running program (i.e, process) that shares resources. And when multiple threads run simultaneously is called multithreading. In many applications, you have seen multiple tasks running simultaneously, for example, Google Docs where while typing text, the spell check and autocorrect tasks are running.

The server also uses multithreading to provide its services to multiple client requests. In Java, you can create threads in two ways, either by implementing the Runnable interface or by extending the Thread class.