**DEFINITION**

**Java**

Java is a widely used programming language expressly designed for use in the distributed environment of the internet. It is the most popular programming language for Android smartphone applications and is among the most favored for [edge device](https://searchnetworking.techtarget.com/definition/edge-device) and [internet of things](https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT) development.

Java was designed to have the look and feel of the [C++](https://searchsqlserver.techtarget.com/definition/C) language, but it is simpler to use than C++ and enforces an [object-oriented programming](https://searchmicroservices.techtarget.com/definition/object-oriented-programming-OOP) model. Java can be used to create complete applications that may run on a single computer or be distributed among [servers](https://whatis.techtarget.com/definition/server) and [clients](https://searchenterprisedesktop.techtarget.com/definition/client) in a network. It can also be used to build a small application module or [applet](https://searchmicroservices.techtarget.com/definition/applet) for use as part of a webpage.

**Elements and principles of Java**

It is difficult to provide a single reason as to why the Java programming language has become so ubiquitous. However, the language's major characteristics have all played a part in its success, including the following components:

* **Programs created in Java offer portability in a network.** The source code is compiled into what Java calls [bytecode](https://whatis.techtarget.com/definition/bytecode), which can be run anywhere in a network on a server or client that has a [Java virtual machine](https://www.theserverside.com/definition/Java-virtual-machine-JVM) (JVM). The JVM interprets the bytecode into code that will run on computer hardware. In contrast, most programming languages, such as [COBOL](https://searchitoperations.techtarget.com/definition/COBOL-Common-Business-Oriented-Language), C++, [Visual Basic](https://searchwindevelopment.techtarget.com/definition/Visual-Basic) or [Smalltalk](https://whatis.techtarget.com/definition/Smalltalk), compile code into a [binary file](https://whatis.techtarget.com/definition/binary-file). Binary files are platform-specific, so a program written for an Intel-based Windows machine cannot on run a Mac, a Linux-based machine or an IBM mainframe. The JVM includes an optional [just-in-time (JIT) compiler](https://www.theserverside.com/definition/just-in-time-compiler-JIT) that dynamically compiles bytecode into executable code as an alternative to interpreting one bytecode instruction at a time. In many cases, the dynamic JIT compilation is faster than the virtual machine interpretation.
* **The code is**[**robust**](https://whatis.techtarget.com/definition/robust)**.** Unlike programs written in C++ and some other languages, Java [objects](https://searchmicroservices.techtarget.com/definition/object) contain no references to data external to themselves or other known objects. This ensures that an instruction cannot contain the address of data storage in another application or in the [operating system](https://whatis.techtarget.com/definition/operating-system-OS) itself, either of which would cause the program and perhaps the operating system itself to terminate or [crash](https://whatis.techtarget.com/definition/crash). The JVM makes a number of checks on each object to ensure [integrity](https://searchdatacenter.techtarget.com/definition/integrity).
* **Java is object-oriented.** An object can take advantage of being part of a [class](https://whatis.techtarget.com/definition/class) of objects and inherit code that is common to the class. Objects are thought of as "nouns" that a user might relate to rather than the traditional procedural "verbs." A [method](https://whatis.techtarget.com/definition/method) can be thought of as one of the object's capabilities or behaviors. Being object-oriented is relatively common in today's programming landscape, but back in 1996, only a handful of languages were implementing object-oriented concepts and [design patterns](https://searchsoftwarequality.techtarget.com/definition/pattern) effectively. The ability to develop with a language created from the ground up with object-orientation as its explicit purpose made Java an exciting platform upon which to program.
* **Applet offers flexibility.** In addition to being executed on the client rather than the server, a Java applet has other characteristics designed to make it run fast.
* **Developers can learn Java quickly.** With [syntax](https://whatis.techtarget.com/definition/syntax) similar to C++, Java is relatively easy to learn, especially for those with a background in C.

A common misconception is that there is an association between Java and [JavaScript](https://www.theserverside.com/definition/JavaScript). The two languages share similarities in syntax, but, otherwise, are two very different constructs.

**Java platforms**

There are three key platforms upon which programmers develop Java applications:

1. **Java SE.** Simple, stand-alone applications are developed using Java Standard Edition. Formerly known as J2SE, Java SE provides all of the [APIs](https://searchmicroservices.techtarget.com/definition/application-program-interface-API) needed to develop traditional desktop applications.
2. **Java EE.** The Java Enterprise Edition, formerly known as [J2EE](https://www.theserverside.com/definition/J2EE-Java-2-Platform-Enterprise-Edition), provides the ability to create server-side components that can respond to a web-based request-response cycle. This arrangement allows the creation of Java programs that can interact with internet-based clients, including web browsers, [CORBA](https://searchsqlserver.techtarget.com/definition/CORBA)-based clients and even [REST](https://searchmicroservices.techtarget.com/definition/REST-representational-state-transfer)- and [SOAP](https://searchmicroservices.techtarget.com/definition/SOAP-Simple-Object-Access-Protocol)-based web services.
3. **Java ME.** Java also provides a lightweight platform for mobile development known as Java Micro Edition, formerly known as [J2ME](https://www.theserverside.com/definition/J2ME-Java-2-Platform-Micro-Edition). Java ME has proved a very popular platform for [embedded device](https://whatis.techtarget.com/definition/embedded-device) development, but it struggled to gain traction in the smartphone development arena. In terms of smartphone development, Android has become the mobile development platform of choice.

**Examples of Java in use**

Using the various components provided by Java EE, it is easy for developers to write programs that employ popular software design patterns and universally agreed upon best practices.

For example, Struts, Spring and JavaServer Faces frameworks all use a Java [servlet](https://searchmicroservices.techtarget.com/definition/servlet) to implement the front controller design pattern for centralizing requests.

Meanwhile, a big part of the Java ecosystem is the large variety of [open source](https://whatis.techtarget.com/definition/open-source) projects, software platforms and APIs that the community has built using the language. For example, the[Apache Foundation](https://whatis.techtarget.com/definition/Apache-Software-Foundation) hosts a variety of projects written using Java, including:

* Simple logging frameworks for Java (SLF4J).
* [Big data](https://searchdatamanagement.techtarget.com/definition/big-data) processing frameworks, such as [Yarn](https://searchdatamanagement.techtarget.com/definition/Apache-Hadoop-YARN-Yet-Another-Resource-Negotiator) and [Hadoop](https://searchdatamanagement.techtarget.com/definition/Hadoop).
* Integration platforms like [Apache Camel](https://www.theserverside.com/definition/Apache-Camel), Apache Axis and CXF for RESTful web service development.
* [Microservices](https://searchmicroservices.techtarget.com/definition/microservices) development platforms.

More enterprises will attempt to transition Java EE environments into the [cloud](https://searchnetworking.techtarget.com/definition/cloud). As Java developers are creating Java cloud services, the ability to scale up those services quickly is a key concern, as is the ability to collaborate in the cloud.

**Criticisms of Java**

Java programs can run just about anywhere, although nuances sometimes poked holes in that theory. In the early years of the Java programming language, [WORA](https://whatis.techtarget.com/definition/write-once-run-anywhere-WORA) was a common Java rallying cry, standing for "Write Once, Run Anywhere," although some developers, with their tongues in cheeks, whispered "WODE: Write Once, Debug Everywhere."

Java syntax is often decried for being too verbose. Superfluous setters and getters and [strong typing](https://whatis.techtarget.com/definition/strongly-typed) tend to make Java programs look bloated. In response, several peripheral languages have emerged to address these issues, including [Groovy](https://www.theserverside.com/definition/Groovy), which was quickly embraced by the Java community.

Another common criticism is that Java programs fail to scale linearly in the world of high-performance computing. Due to the way Java references objects internally, complex and concurrent list-based operations can bring the JVM to a crawl. The [Scala](https://searchbusinessanalytics.techtarget.com/definition/Scala-Scalable-Language) language addresses many of the shortcomings of the Java language that reduce its ability to scale.