**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.