**What is J2SE?**

J2SE is a collection of basic Java classes and APIs. Its latest version Java 6 (also known as Java Standard Edition 6.0 or Java SE 6 or Java 1.6), codenamed Mustang, was released in December, 2006. Current revision is the Update 26, which was released on June, 2011. It has 3700+ classes and interfaces. It focuses on new specifications and APIs including XML, Web Services, JDBC version 4.0, programming based on Annotations, API’s for Java compiler and Application client GUI. This was on top of already existing features like Annotations, Generics and Autoboxing. Annotations are a mechanism for tagging classes with metadata so that they can be used by metadata-aware programs. Generics is a mechanism of specifying types for objects belonging to collections such as Arraylists, so that type safety is guaranteed at compile time. Autoboxing allows the automatic conversions between primitive types (e.g. int) and wrapper types (e.g. Integer). Also, the support for older versions of Windows (Win9x series) was removed starting from the Update 7.

**What is J2EE?**

J2EE provides a server programming platform in Java. J2EE adds functionality (libraries) for the deployment of distributed and multi-tier java applications running on application servers. Current version of J2EE is Java EE 6. JDBC (Java Database Connectivity), RMI (Remote Method Invocation), JMS (Java Message Service), web services and XML are some of the specifications offered by Java EE. Furthermore, specifications unique to Java EE such as Enterprise JavaBeans (EJB), Connecters, Servlets, portlets, Java Server Pages (JSP) are also offered. The aim of this is to allow programmers to develop application with high scalability and portability. Java EE developers can focus on business logic (instead of infrastructure/integration) because the application servers will take care of transactions, security and concurrency.

**What is the difference between J2SE and J2EE?**

J2SE is a collection of base classes and APIs that provides basic functionality (Java language, virtual machine and base libraries) for developing standard Java applications, while J2EE offers a collection of technologies and APIs for developing multi-tier enterprise applications. In other words, J2SE is used for developing applications that execute as standalone desktop programs or applets, but J2EE is typically used for writing applications that execute inside a J2EE container. J2EE has all the functionality of J2SE. But, it has additional functionality such as EJB, JSP, Servelts and XML technology. It also includes tests for checking the compliance of applications with existing applications that support J2EE.

<http://www.differencebetween.com/difference-between-j2se-and-vs-j2ee/>

**Java SE** stands for Java standard edition and is normally for developing desktop applications, forms the core/base API.

**Java EE** stands for Java enterprise edition for applications which run on servers, for example web sites.

**Java ME** stands for Java micro edition for applications which run on resource constrained devices (small scale devices) like cell phones, for example games.

[**http://stackoverflow.com/questions/1065240/whats-the-main-difference-between-java-se-and-java-ee**](http://stackoverflow.com/questions/1065240/whats-the-main-difference-between-java-se-and-java-ee)

[**https://www.youtube.com/watch?v=Ji9Za\_HnfVs**](https://www.youtube.com/watch?v=Ji9Za_HnfVs)

[**https://www.facebook.com/groups/131207903938864/**](https://www.facebook.com/groups/131207903938864/)

**API:** [**http://oopbook.com/guides/api-documentation/**](http://oopbook.com/guides/api-documentation/)

* **There are three notions of the JVM: specification, implementation, and instance. The specification is a document that formally describes what is required of a JVM implementation. Having a single specification ensures all implementations are**[**interoperable**](https://en.wikipedia.org/wiki/Interoperability)**. A JVM implementation is a computer program that meets the requirements of the JVM specification. An instance of a JVM is an implementation running in a**[**process**](https://en.wikipedia.org/wiki/Process_(computing))**that executes a computer program compiled into**[**Java bytecode**](https://en.wikipedia.org/wiki/Java_bytecode)**.**
* [**https://en.wikipedia.org/wiki/Java\_virtual\_machine**](https://en.wikipedia.org/wiki/Java_virtual_machine)