History of JAVA

JDK Alpha and Beta (1995)

The Java Alpha and Beta was the first releases but they have highly unstable APIs and ABIs. The supplied Java web browser was named WebRunner.

JDK 1.0 (January 23, 1996)

It was the first stable released version of Java. Its codename was Oak. The first stable version of JDK was JDK 1.0.2 and it was called Java 1.

Up to JDK 1.0.1, private and protected keywords could be used together to create yet another form of protection which used to act as a restriction to methods or variables mainly to subclasses of a given class. In JDK 1.0.2, this capability has been removed.

JDK 1.1 (February 19, 1997)

Some additions were included to this version. i.e.

* The concept of Inner Class
* JavaBeans
* JDBC
* RMI
* AWT event model was totally reshaped.
* Reflection (which supported Introspection only, modification was not possible at runtime).
* JIT(Just In Time) compiler on Microsoft Windows platforms, produced for JavaSoft by Symantec
* Internationalization and Unicode support originating from Taligent.

J2SE 1.2 (December 8, 1998)

Its codename was Playground. First time, it was called J2SE (Java 2 Platform, Standard Edition) .It replaced JDK to recognize the base platform from J2EE (Java 2 Platform, Enterprise Edition) and J2ME(Java 2 Platform, Micro Edition) .It was a very important java release as it tripled the size of the Java platform to 1520 classes in 59 packages.

Some additions were included to this version. i.e.

* Java plug-in
* Java IDL, an IDL implementation for CORBA interoperability
* Collections framework
* the Swing graphical API was integrated into the core classes
* Sun's JVM was equipped with a JIT compiler for the first time

J2SE 1.3 (May 8, 2000)

Its codename was Kestrel. Some additions were included to this version. i.e.

* HotSpot JVM included.
* RMI was modified to support optional compatibility with CORBA.
* JNDI (Java Naming and Directory Interface).
* Java Platform Debugger Architecture (JPDA) included.
* JavaSound.
* Synthetic proxy classes.

J2SE 1.4 (February 6, 2002)

Its codename was Merlin. It was the first Java platform which was released under the Java Community Process.

Some additions were included to this version. i.e.

* Improved libraries.
* Perl regular expressions included.
* Provided exception chaining (It allows an exception to encapsulate original lower-level exception).
* IPv6 support (Internet Protocol version 6).
* Logging API (Specified in JSR 47.)
* Image I/O API for reading and writing images in formats like JPEG and PNG.
* XML parser and XSLT processor integrated.
* Security and cryptography extensions (JCE, JSSE, JAAS) integrated.

Support and security updates for Java 1.4 ended in October 2008.

J2SE 5.0 (September 30, 2004)

Its codename was Tiger. It was originally numbered 1.5, which is still used as the internal version number. So, it was changed to 5.0 to "better reflect the level of maturity, stability, scalability and security of the J2SE". This process also was released under the Java Community Process.

Support and security updates for Java 5.0 ended on November 3, 2009 but updates were available to paid Oracle customers until May 2015.

J2SE 5.0added some significant new language features:

* It provided compile-time (static) type safety for collections and eliminates the need for most typecasts.
* Used Metadata or annotations.
* Autoboxing/unboxing.
* Enumerations.
* Enhanced for each loop.
* Improved semantics of execution for multi-threaded Java programs.
* Static imports.

There were also some improvements in standard libraries:

* Automatic stub generation for RMI objects.
* Swing: It provided a skinny look and feel.
* The concurrency utilities in package java.util.concurrent.
* Scanner class for parsing data from various input streams and buffers.

Java 5 was the last release of Java which officially supportedthe Microsoft Windows 9x line (Windows 95, Windows 98, Windows ME).

Windows Vista was the last version of Windows that J2SE 5 supported before going to end in October 2009.

Java 5.0 is the default version of Java installed on Apple Mac OS X 10.5 (Leopard). Java 6 can be installed

Java SE 6 (December 11, 2006)

Its codename was Mustang. After the release of this version, Java replaced the name J2SE to Java SE and dropped the .0 from the version number.

Some additions were included to this version. i.e.

* Dropped the support for older Win9x versions.
* Scripting Language Support.
* Generic API for tight integration with scripting languages.
* Improved Web Service support.
* JDBC 4.0 support.
* Use a Java Compiler API to invoke a Java Compiler programmatically.

After the release of Java 6, Sun released many updates to fix bugs.

Java SE 7 (July 28, 2011)

Its codename was Dolphin. It was launched on 7, July 2011 but was made available for developers on July 28, 2011.

Some additions were included to this version. i.e.

* JVM support for dynamic languages.
* Compressed 64-bits pointer.
* Strings added in switch.
* Automatic resource management in try-statement.
* Underscores allowed in numeric literals.
* Binary integer literals.
* Improved type interface for creating generic instance. (also called diamond operator <>)
* Improved catching and throwing. (catch multiple exceptions and rethrow with improved type checking)
* Provided Java Deployment rulesets.

It was the default version to download on java.com from April 2012 up to the release of Java 8.

## **Java 8 Features**

Release Date : March 18,2014

Code name culture is dropped.

Included features were:

* [Lambda expression](https://howtodoinjava.com/java8/complete-lambda-expressions-tutorial-in-java/) support in APIs
* [Stream API](https://howtodoinjava.com/java8/java-streams-by-examples/)
* [Functional interface](https://howtodoinjava.com/java8/functional-interface-tutorial/) and [default methods](https://howtodoinjava.com/java8/default-methods-in-java-8/)
* [Optionals](https://howtodoinjava.com/java8/java-8-optionals-complete-reference/)
* Nashorn – JavaScript runtime which allows developers to embed
* JavaScript code within applications
* Annotation on Java Types
* [Unsigned Integer](https://howtodoinjava.com/java8/java-8-exact-airthmetic-operations-supported-in-math-class/)
* [Arithmeti](https://howtodoinjava.com/java8/java-8-exact-airthmetic-operations-supported-in-math-class/)c
* Repeating annotations
* [New Date and Time API](https://howtodoinjava.com/java8/date-and-time-api-changes-in-java-8-lambda/)
* Statically-linked JNI libraries
* Launch JavaFX applications from jar files
* Remove the permanent
* generation from GC

Java 9 Features

Java 9 was made available on September, 2017.

The biggest change is the modularization i.e. Java modules.

Some important features/[changes in Java 9](https://howtodoinjava.com/java9/java9-new-features-enhancements/) are:

* [Java platform module system](https://howtodoinjava.com/java9/java-9-modules-tutorial/)
* [Interface Private Methods](https://howtodoinjava.com/java9/java9-private-interface-methods/)
* HTTP 2 Client
* JShell – REPL Tool
* Platform and JVM Logging
* Process API Updates
* Collection API Updates
* [Improvements in Stream API](https://howtodoinjava.com/java9/stream-api-improvements/)
* Multi-Release JAR Files
* @Deprecated Tag Changes

## Java 10 Features

## After Java 9 release, Java 10 came very quickly. Unlike it’s previous release, Java 10 does not have that many exciting features, still it has [few important updates](https://howtodoinjava.com/java10/java10-features/) which will change the way you code, and other future Java versions.

* [JEP 286: Local Variable Type Inference](https://howtodoinjava.com/java10/var-local-variable-type-inference/)
* JEP 322: Time-Based Release Versioning
* JEP 304: Garbage-CollectorInterface
* JEP 307: Parallel Full GC for G1
* JEP 316: Heap Allocation on Alternative Memory Devices
* JEP 296: Consolidate the JDK Forest into a Single Repository
* JEP 310: Application Class-Data Sharing
* JEP 314: Additional Unicode Language-Tag Extension
* JEP 319: Root Certificates

Java 11 Features

[Java 11](https://howtodoinjava.com/java11/features-enhancements/) (released on September 2018) includes many important and useful updates. Let’s see the new features and improvements, it brings for developers and architects.

* HTTP Client API
* Launch Single-File Programs Without Compilation
* String API Changes
* Collection.toArray(IntFunction)
* Files.readString() and Files.writeString()
* Optional.isEmpty()

**Java 12 Features**

[Java 12](https://howtodoinjava.com/java12/new-features-enhancements/) (released on March 19, 2019) is latest version available for JDK. Let’s see the new features and improvements, it brings for developers and architects.

* · Collectors.teeing() in Stream API
* · String API Changes
* · Files.mismatch(Path, Path)
* · Compact Number Formatting
* · Support for Unicode 11
* · Switch Expressions (Preview)

Comment:

2. Applications of Java

# Mobile Applications

Java is considered as the official programming language for mobile app development. It is compatible with software such as Android Studio and Kotlin. Now you must be wondering why only Java? The reason is that it can run on Java Virtual Machine(JVM), whereas Android uses DVK(Dalvik Virtual Machine) to execute class files. These files are further bundled as an Android application package(APK). With Java and its OOPs principles, it provides better security and ease of simplicity with Android.

# Desktop GUI Applications

All desktop applications can easily be developed in Java. Java also provides GUI development capability through various means mainly Abstract Windowing Toolkit (AWT), Swing, and JavaFX. While AWT holds a number of pre-assembled components like menu, list, button. Swing is a GUI widget toolkit, it provides certain advanced elements like trees, scroll panes, tables, tabbed panels, and lists.

# Web-based Applications

Java is also used to develop web applications. It provides vast support for web applications through Servlets, Struts, or JSPs. With the help of these technologies, you can develop any kind of web application that you require. The easy coding and high security offered by this programming language allow the development of a large number of applications for health, social security, education, and insurance.

# Enterprise Applications

Java is the first choice of many software developers for writing applications and Java Enterprise Edition (Java EE) is a very popular platform that provides API and runtime environment for scripting. It also includes network applications and web-services. JavaEE is also considered as the backbone for a variety of banking applications which have Java running on the UI to back server end.

# Scientific Applications

Sofware developers see Java is the weapon of choice when it comes to coding the scientific calculations and mathematical operations. These programs are designed to be highly secure and lighting fast. they support a higher degree of portability and offer low maintenance. Some of the most powerful applications like the MATLAB use Java for interacting user interface as well as part of the core system.

# Gaming Applications

Java has the support of the open-source most powerful 3D-Engine, the jMonkeyEngine that has the unparalleled capability when it comes to the designing of 3D games. However, it does cause an occasional latency issue for games as garbage collection cycles can cause noticeable pauses. This issue will be solved in the newer versions of JVMs.

# 

# 

# Big Data technologies

Java is the reason why the leading Big Data technologies like Hadoop have become a reality and also the most powerful programming languages like Scala are existing. It is crystal clear that Java is the backbone when it comes to developing Big Data using Java.

# Business Applications:

Java EE platform is designed to help developers create large-scale, multi-tiered, scalable, reliable, and secure network applications. These applications are designed to solve the problems encountered by large enterprises. The features that make enterprise applications powerful, like security and reliability, often make these applications complex. The Java EE platform reduces the complexity of enterprise application development by providing a development model, API, and runtime environment that allow developers to concentrate on functionality.

# Distributed Applications:

Distributed applications have several common requirements that arise specifically because of their distributed nature and of the dynamic nature of the system and platforms they operate on. Java offers options to realize these applications. The Jini (Java Intelligent Networking Infrastructure) represents an infrastructure to provide, register, and find distributed services based on its specification. One integral part of Jini is JavaSpaces, a mechanism that supports distribution, persistence, and migration of objects in a network.

# Cloud-Based Applications:

Cloud computing means on-demand delivery of IT resources via the internet with pay-as-you-go pricing. It provides a solution for IT infrastructure at a low cost. Java provides you with features that can help you build applications meaning that it can be used in the SaaS, IaaS and PaaS development. It can serve the companies to build their applications remotely or help companies share data with others, whatever the requirement.

Comment:

3. What is Object Oriented Programming?

Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

Object-oriented programming is a programming paradigm based on the concept of "objects", which can contain data, in the form of fields, and code.

Object-oriented programming has several advantages over procedural programming:

* OOP is faster and easier to execute
* OOP provides a clear structure for the programs
* Objects have states and behaviour, An object is an instance of a class
* Class − A class can be defined as a template/blueprint that describes the behavior/state that the object of its type support.

4) Features of Java

List of most important features of Java language are

1.Simple

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

### 2.Object-oriented

* 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 incorporates both data and behavior.

3. Platform Independent

* Java is platform independent because it is different from other languages like [C](https://www.javatpoint.com/c-programming-language-tutorial), [C++](https://www.javatpoint.com/cpp-tutorial), etc. which are compiled into platform specific machines while Java is a write once, run anywhere language.

### 4. Secured

* Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because: No explicit pointer
* Java Programs run inside a virtual machine sandbox

### 5. Dynamic

* Java is a dynamic language. It supports dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++.

### 6. Multi-threaded

* A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads.

### 7. Distributed

* Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications

8. High-performance

* Java is faster than other traditional interpreted programming languages because Java bytecode is "close" to native code. It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

Comment: