## Introduction

Software is a development process which converts the imaginaries into reality by writing comes set of programs. In IT we develop two types of applications. They are standalone applications and distributed applications.

A standalone application is one which runs in the context of local disk. All standalone applications are not sharable. System software and application software comes under standalone applications.

* System software is used for developing functionality of hardware devices. Examples are C and ALP (Assembly Language Programming).
* Application software is used for developing organizations-oriented applications. This is also known as backend software’s. Examples are dbase, dbase III plus, FoxPro, oracle versions released till now.
* Internet software is used for developing distributed applications.

A distributed application is one which can be accessed across the globe. Distributed application is also one which runs in the contest of the World Wide Web. To develop distributed applications, we must use client-server architecture.

In client-server architecture we must have at least two programs they are client program and server program. A client program is one which always makes a request to get the service from the server. A server program is one which will do three operations receiving the request from client, processing the client request and sending the response to the client.

**History of JAVA:**

The Java programming language was developed by a team of engineers, the Green Team comprising of James Gosling, Patrick Naughton, Mike Sheridan at Sun Microsystems Inc. in 1991 as its initial implementation.

The language was originally designed for televisions and small embedded systems like set-top boxes, but its features were too advanced for the digital cable industry back then. Hence, Java’s first licensed version was publicly available by Sun Microsystems in 1996 as JDK 1.0(Java Development Kit). Due to its non-compliance with its earlier specification, the Java 1.0 compiler was rewritten by Arthur Van Hoff.

James Gosling is known as the Father of Java Programming Language. He recognized the name **Java** for the language as there were many suggestions from **Silk** to **Oak,** but Oak was not considered later as it was a registered trademark of another Tech company. He wanted a dynamic and unique name, it is said that James Gosling while having a cup of coffee nearby his office thought of the name Java, a coffee grown in Indonesia.

Hence, it was chosen and the language was renamed **Java** in 1995. In January 2010, Oracle acquired Java and Sun Microsystems and now continues to release the licensed JDK every year. Java as a programming language and its history is very interesting to know.

### **Evolution of Java:**

There are many java versions that have been released. Each version adds a new feature to this powerful language and provides developers useful functionalities. Java SE 8 is recognized as a crucial update to this language as it added many new features and specifications and became the benchmark of an Object-Oriented Programming Language.

Currently, on the date of publishing of this post, Java 17 is the latest Java SE Version available on Oracle Java SE Platform. Java SE 18 is still a work in progress.

* JDK Alpha and Beta (1995)
* JDK 1.0 (23rd Jan, 1996)
* JDK 1.1 (19th Feb, 1997)
* J2SE 1.2 (8th Dec, 1998)
* J2SE 1.3 (8th May, 2000)
* J2SE 1.4 (6th Feb, 2002)
* J2SE 5.0 (30th Sep 2004): This version of Java introduced major changes in the Java programming world. Some of them are:
  + [Java Concurrency API](https://www.java2blog.com/2016/12/java-threadpoolexecutor-example.html)
  + Var argument methods
  + Generics
  + Annotations
  + [For each loop](https://www.java2blog.com/2016/09/java-8-foreach-examples.html)
  + Enumerations
  + Static import
  + Autoboxing and unboxing etc.
* Java SE 6 (11th Dec 2006)
* Java SE 7 (28th July 2011): Java 7 has the following new major changes:
  + Multi-catch block
  + Support of String in Switch case
  + Try with resources etc.
* Java SE 8 (18th March 2014): Java 8 has introduced major programming paradigms. Some of them are:
  + Lambda Expressions.
  + Functional Interfaces
  + Default methods in the interface.
  + Streams
  + Enhanced Security etc.
* Java SE 9 (September 21,2017): This version was developed to include better support for multi-gigabyte heaps, better native code integration, a different default garbage collector. Other notable features were: \
  + Provided Money and Currency API.
  + Tight integration with JavaFX.
  + Java implementation of reactive streams.
  + More Concurrency Updates.
  + Provided Java Linker
* Java SE 10 (March 10,2018): This version was released to remove primitive data types and move towards 64-bit addressable arrays to support large data sets for data analytics. Some notable features were:
  + Experimental Java-Based JIT Compiler
  + Garbage-Collector Interface
  + Consolidate the JDK Forest into a Single Repository
* Java SE 11 (September 2018)
* Java SE 12 (March 2019)
* Java SE 13 (September 2019)
* Java SE 14 (March 2020)
* Java SE 15 (September 2020)
* Java SE 16 (March 2021)
* Java SE 17 (September 2021)

## Terminologies of Java

For an introduction to Java Programming, it is recommended to have some idea of the basic Java terms to help your learning. Let’s have a quick look at them.

### **1. Java Development Kit (JDK)**

Generally referred to as JDK, this kit provides the functionality required to develop, compile and execute your java application or program. In short, we can say if we want to create, compile or run our java application, then JDK must be installed in our system.

### **2. Java Runtime Environment (JRE)**

The Java Runtime Environment provides the ability to execute our applications at runtime. It comes included with the JDK. JRE installation ensures the smooth running of your java application.

However, it is not responsible for compiling a program. This is a unique aspect of JRE, a user does need to have a complete JDK set up to run/use any Java application, only having the JRE will be sufficient as it provides support through browsers, applets, and other plugins to run a java application.

### 3. **Java Virtual Machine (JVM)**

Commonly referred to as JVM, Java Virtual Machine to say is an abstract machine that enables a computer to run java programs as well as programs in other programming languages that are already compiled to java **Bytecode**. There are two phases to this

* + The Java Compiler compiles the java source code and generates an intermediatory code. This intermediate code is known as the Byte Code.
  + In the executing phase, this Byte Code is interpreted by the Java Interpreter and can be executed on any machine running a JVM. So, JVM reifies execution and the JRE provides the environment for it.

To summarize the role of each, refer to the diagram below:



### **4. Byte Code**

As discussed above, the Java Compiler compiles the Java Source Code into an intermediate code. This is generally saved as a **.class File**. This intermediate code is known as Byte Code and is an important aspect of Java as this makes the Language Platform-Independent. This is a machine-independent code and can be read or executed by any machine running a JVM.

### **5. Garbage Collection**

Garbage Collection eases the work of a programmer. Its main job is to deallocate or delete unused references or objects. Thus the programmer need not worry about deleting occupied memory resources or memory management. If the reference is not used over a long period of time, the garbage collector will automatically delete it.

JAVA released to the market in three categories J2SE (JAVA 2 Standard Edition), J2EE (JAVA 2 Enterprise Edition) and J2ME (JAVA 2 Micro/Mobile Edition).

* J2SE is basically used for developing client-side applications/programs.
* J2EE is used for developing server-side applications/programs.
* J2ME is used for developing server-side applications/programs.

If you exchange the data between client and server programs (J2SE and J2EE), by default JAVA is having on internal support with a protocol called http. J2ME is used for developing mobile applications and lower/system level applications. To develop J2ME applications we must use a protocol called WAP (Wireless Applications Protocol).

**Features:**

* 1. Simple
  2. Platform independent
  3. Architectural neutral
  4. Portable
  5. Multi-threading
  6. Distributed
  7. Networked
  8. Robust
  9. Dynamic
  10. Secured
  11. High performance
  12. Interpreted
  13. Object Oriented Programming Language

**Simple**: JAVA is simple because JAVA is free from pointers hence, we can achieve less development time and less execution time. Rich set of API (application protocol interface) is available to develop any complex application. The software JAVA contains a program called garbage collector which is always used to collect unreferenced (unused) memory location for improving performance of a JAVA program. [Garbage collector is the system JAVA program which runs in the background along with regular JAVA program to collect unreferenced memory locations by running at periodical interval of times for improving performance of JAVA applications.

**Platform Independent:** A program or technology is said to be platform independent if and only if which can run on all available operating systems.

The language like JAVA will have a common data types and the common memory spaces on all operating systems and the JAVA software contains the special programs which converts the format of one operating system to another format of other operating system. Hence JAVA language is treated as platform independent language.

**Architectural Neutral:** A language or technology is said to be architectural neutral which can run on any available processors in the real world. The languages like C, Cpp are treated as architectural dependent. The language like JAVA can run on any of the processor irrespective of their architecture and vendor.

**Portable:** A portable language is one which can run on all operating systems and on all processors irrespective their architectures and providers. The languages like C, Cpp are treated as non- portable languages whereas the language JAVA is called portable language.

**Multi-Threading:** A flow of control is known as thread. A multi-threaded program is one in which there exists multiple flow of controls i.e., threads.

A program is said to be multi-threaded program if and only of there exists n number of sub- programs. For each and every sub-program there exists a separate flow of control. All such flow of controls is executing concurrently. Such flow of controls is known as threads. Such type of applications is known as multi-threading applications.

**Distributed:** A service is a said to be a distributed service which runs in multiple servers and that service can be accessed by n number of clients across the globe. In order to develop distributed applications, we must require architecture called trusted network architecture. To develop these applications, we require a technology called J2EE. Distributed applications are preferred by large scale organizations.

**Networked:** In real world we have two types of networks. They are un-trusted networks and trusted networks.

Un-trusted networks: A network is said to be un-trusted network in which there exists n number of inter connected non-autonomous architecture. Un-trusted network is also known as LAN. Using this network architecture, we develop centralized applications.

Trusted network: A network is said to be trusted network in which there exists n number of inter connected autonomous architecture. Trusted network is also known as WAN. Using this network, we can develop distributed applications. A distributed application is one which runs on multiple servers and it can be access in unlimited graces. In order to develop distributed applications, we may use a technology called J2EE.

**Java is object-oriented programming language**:

We have two types of programming models (paradigms) are available. They are procedure-oriented programming language and object-oriented programming language.

If we represent the data using procedural oriented programming languages then there is no security for the data which we represent.

**For example**, when we represent the data of a student in C language using structures concept, the student data can be accessed by all the functions which we write as a part of C program. If one of the functions manipulates or damages the data then we are losing correction-less (integrity) of the data. Examples of procedure-oriented programming languages are FORTRON, COBOL, PASCAL, BASIC, C, etc.

When we represent the data in object-oriented programming language, we get the security. Examples of object-oriented programming languages are LISP, ADA, ALGOL, SMALLTALK, OBJECT COBOL, OBJECT PASCAL, C++, JAVA, DOT NET, etc. In order to say any language is an object-oriented programming language it has to satisfy 8 principles of OOPs.

**OOPs Principles:**

1. Class.
2. Object.
3. Data Abstraction.
4. Data Encapsulation.
5. Inheritance.
6. Polymorphism.
7. Dynamic Binding.
8. Message Passing.

## Class

A **class** is a blueprint/template that defines the state and Behaviour of objects. It is the logical entity that wraps all our code into a single unit and structures the code. In Java, we cannot compile our code without a **class**. Hence, we must write the code only within the class body surrounded by Curly braces.

The unique characteristic of a **class** is that a user can define their data members (Variables with Data Types) and methods, and we can access them by creating instances of the class.

**Object**

An entity that has a state and Behaviour is known as an **Object**. In java, an Object**is an instance of**a class. Objects are created during runtime based on the blueprint/template that a class provides.

It is crucial to create objects; without them, we cannot access our program’s class members and methods. An object typically has three characteristics:

* **State:** It represents the value of the Object.
* **Behaviour:** It means the behaviour (functionality) of an object.
* **Identity:** Java assigns a unique ID for each Object. The value of the ID is not visible to the external user.

**Example:** For the Employee class, each Employee object will have a state with name, age, and department, and behaviour such as working on the assignment.

**Data Abstraction:**

“Data abstraction is a mechanism of retrieving the essential details without dealing with background details”.

In real world we have three levels of abstractions. They are physical level abstraction,

conceptual/logical level abstraction and view level abstraction.

Physical level abstraction is one; it always deals with physical organization architecture of the application.

Conceptual/logical level abstraction is one it always deals with what kind of data we are entering without dealing with physical architecture of the application. For example, entering the data into the database, writing the coding and applying testing principle comes under conceptual level abstraction.

View level abstraction deals with what kind of data we are retrieving without dealing with both conceptual level abstraction and physical level abstraction. For example, retrieving the data from the data base in various combinations. All internet users come under view level abstraction.

**Data Encapsulation:**

“Data encapsulation is the process of wrapping up on data and associated methods in a single unit”.

* Data encapsulation is basically used for achieving data/information hiding i.e., security.
* When we want to send the data from client to the server we must always send in the form of JAVA object only. Since, by default the JAVA object is in encrypted form (we should not send the data from client to the server in the form of fundamental data).

**Inheritance:**

* Inheritance is the process of taking the features (data members + methods) from one class to another class.
* The class which is giving the features is known as base/parent class.
* The class which is taking the features is known as derived/child/sub class.
* Instance is known as sub classing or derivation or extendable classes or reusability.

**Polymorphism:**

Polymorphism is a process of representing “one form in many forms”.

* In object-oriented programming’s, we have two types of polymorphism. They are compiling time polymorphism and run time polymorphism.
* JAVA does not support compile time polymorphism but JAVA supports only run time polymorphism.

**Dynamic Binding:**

“Dynamic binding is a mechanism of binding an appropriate version of a derived class which is inherited from base class with base class object”.

* Every real time application will be developed with concept of polymorphism and executed

with the concept of dynamic binding.

* Dynamic binding is basically used to reduce the amount of memory space for improving the performance of JAVA applications.

1) C1 O1 = new C1();

2) O1.sum ();

NOTE:

* Function overriding = function heading is same but function definition is different.
* A function is said to be overloaded function if and only if function name is same but its signature (signature represents number of parameters, type of parameters and order of parameters) is different.

**Message Passing:**

Definitions:

1. Exchanging the data between multiple objects is known as message passing.
2. Message passing is the process of exchanging the data between two remote/local objects for a period of time across the network (trusted) for generating multiple requests for obtaining multiple responses for performing a meaningful operation.
3. Message passing JAVA is achieved through the concept of methods.