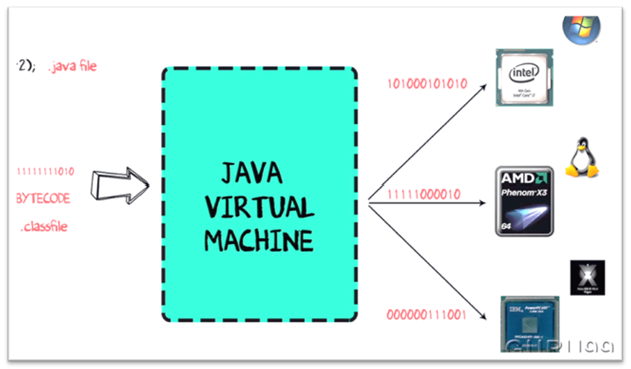
***Java - intro to JVM***

For compiling C language, we need to use different compilers for different OS







In order to write and execute a software program you need the following

**1) Editor**– To type your program into, a notepad could be used for this

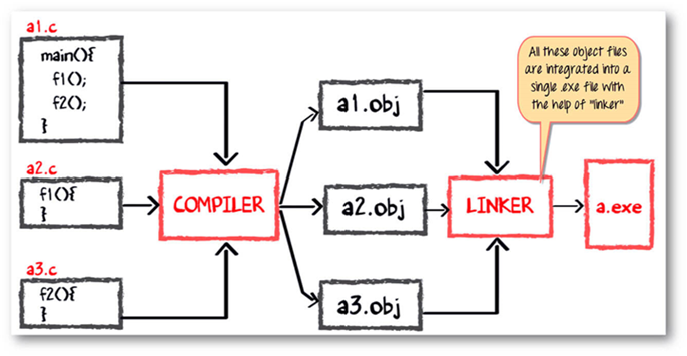
**2) Compiler**– To convert your high language program into native machine code

**3) Linker**– To combine different program files reference in your main program together.

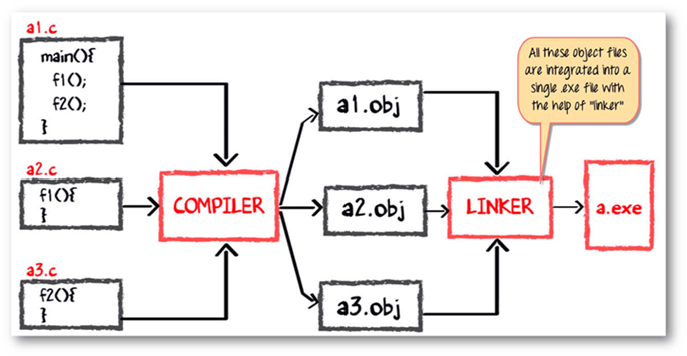
**4) Loader**– To load the files from your secondary storage device like Hard Disk, Flash Drive , CD into RAM for execution. The loading is automatically done when your execute your code.

**5) Execution** – Actual execution of the code which is handled by your OS & processor

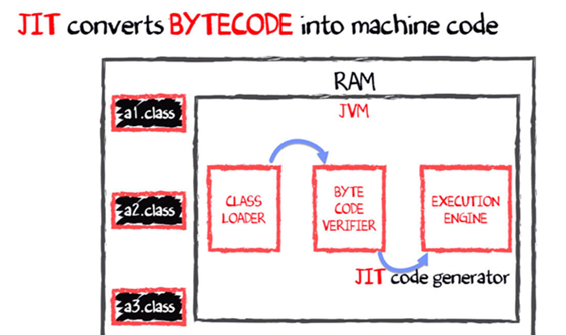
How c program execution works:



How java program execution works:



The compiler will compile the three files and produces a corresponding .class file which consists of BYTE code. **Unlike C, no linking is done**.

The Java VM or Java Virtual Machine resides on the RAM. During execution, using the class loader the class files are brought on the RAM. The BYTE code is verified for any security breaches.

The .class

Will be bought on to jvm using class loader

class loader

using class loader

**JIT - java-in-time compiler**

**JVM - java virtual machine**



JVM: it is responsible for allocating memory space

* Programming languages are classifies as
* Higher Level Language Ex. C++ , Java
* Middle Level Languages Ex. C
* Low Level Language Ex Assembly
* finally the lowest level as the Machine Language.

Why JAVA is both interpreted and compiled language?

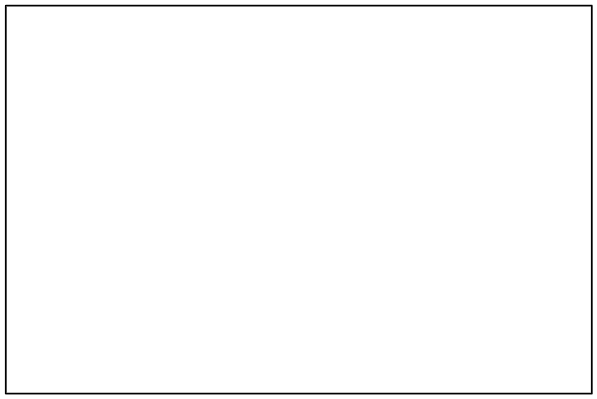
A **compiler** is a program which converts a program from one level of language to another. Example conversion of C++ program into machine code.

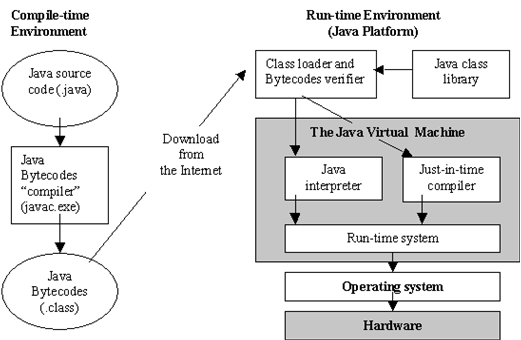
The java compiler is a convert's high level java code into bytecode (which is also a type of machine code).

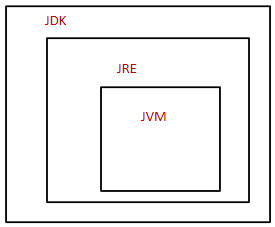
A **interpreter**is a program which converts a program at one level to another programming language at the same level. Example conversion of Java program into C++

In Java , the Just In Time Code generator converts the bytecode into the native machine code which are at the same programming levels.

Java platform architecture:



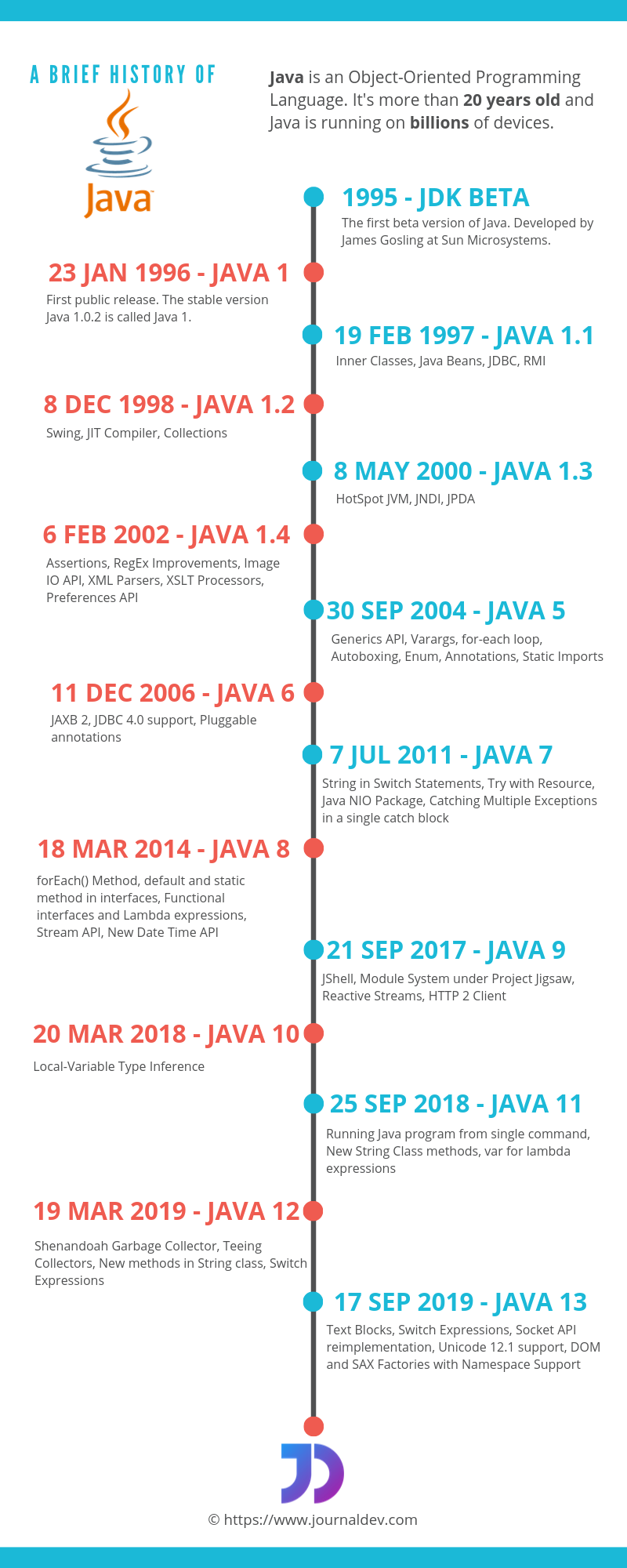
 JDK



JDK parts has JRE and JVM

***History and features of java***

* Java was originally designed for interactive television, but it was too advanced technology for the digital cable television industry at the time.
* The history of Java starts with the Green Team. Java team members (also known as **Green Team**)
* The principles for creating Java programming were "Simple, Robust, Portable, Platform-independent, Secured, High Performance, Multithreaded, Architecture Neutral, Object-Oriented, Interpreted, and Dynamic"
* It was developed by James gosling
* **James gosling** , **Mike Sheridan**, and **Patrick Naughton** initiated the Java language project in June 1991. The small team of sun engineers called **Green Team**
* Java was started as a project called “**Oak**” by James Gosling in June 1991. The goal was to implement a virtual machine that is much simpler than C/C++.
* In 1995 it was named as java because oak was already a trademark for oak technologies



Features of java:



1.Object Oriented program

Java is an object oriented 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.

Basic concepts of OOPs are:

1. [Object](https://www.javatpoint.com/object-and-class-in-java)
2. Class
3. [Inheritance](https://www.javatpoint.com/inheritance-in-java)
4. [Polymorphism](https://www.javatpoint.com/runtime-polymorphism-in-java)
5. [Abstraction](https://www.javatpoint.com/abstract-class-in-java)
6. [Encapsulation](https://www.javatpoint.com/encapsulation)

2.Simple:

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

* Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
* There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

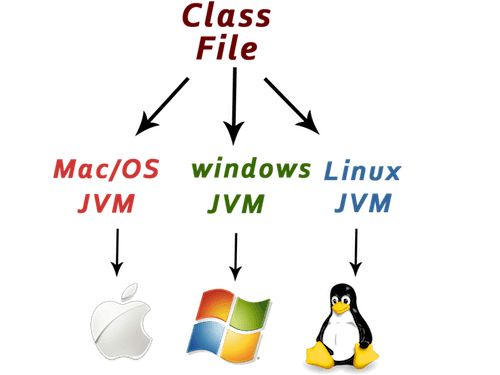
3.Secure:

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**
* **Classloader:** Classloader in Java is a part of the Java Runtime Environment(JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.
* **Bytecode Verifier:** It checks the code fragments for illegal code that can violate access right to objects.
* **Security Manager:** It determines what resources a class can access such as reading and writing to the local disk.

Java language provides these securities by default. Some security can also be provided by an application developer explicitly through SSL, JAAS, Cryptography, etc

4.Platform Independent: (for more reference read first page)



Java code can be run on multiple platforms, for example, Windows, Linux, Sun Solaris, Mac/OS, etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms, i.e., Write Once and Run Anywhere(WORA).

5.Robust:

Robust simply means strong.

* It uses strong memory management.
* There is a lack of pointers that avoids security problems.
* There is automatic garbage collection in java which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
* There are exception handling and the type checking mechanism in Java. All these points make Java robust.

6.Architecture-neutral:

Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.

7.Portable:

Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

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.

9.Distributed:

Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

10.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. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area. Threads are important for multi-media, Web applications, etc.

11.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++.

Java supports dynamic compilation and automatic memory management (garbage collection).

12.Interpreter:

an Interpreter in Java is a computer program that helps to convert a high-level program statement into a machine code comprising source code, pre-compiled code, and scripts. An Interpreter converts the code into machine code when the program is run.

Interpreter is always present in JAVA it is called is JIT(Just-In-Time)Compiler used to debug the code along with the Debugger that allows it is run various codes for the coding part and develop the project.

Working of compiler and interpreter:

