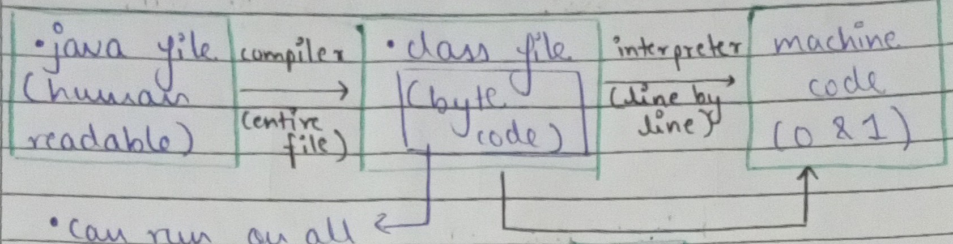
Date __/__/____ Page No. ____

Introduction to Java



- can run on all O.S
- code doesn't run directly, we need JVM for that.

JVM converts byte code to machine code

Note points:

- Java is platform independent
- we can provide this byte code to any system
- but JVM is platform dependent, which means for every O.S the executable file that we get, it has step by step set of instruction dependent on platform.

* JDK vs JRE vs JVM vs JIT

JDK (Java Development Kit)
↳ provides env. to develop & run Java program

JRE (Java Runtime Environment)
↳ provides environment to the program

- → deployment technologies
- → UI toolkit
- → integration libraries
- → base libraries
- etc.

JVM (Java Virtual Machine)

JIT (Just in time)

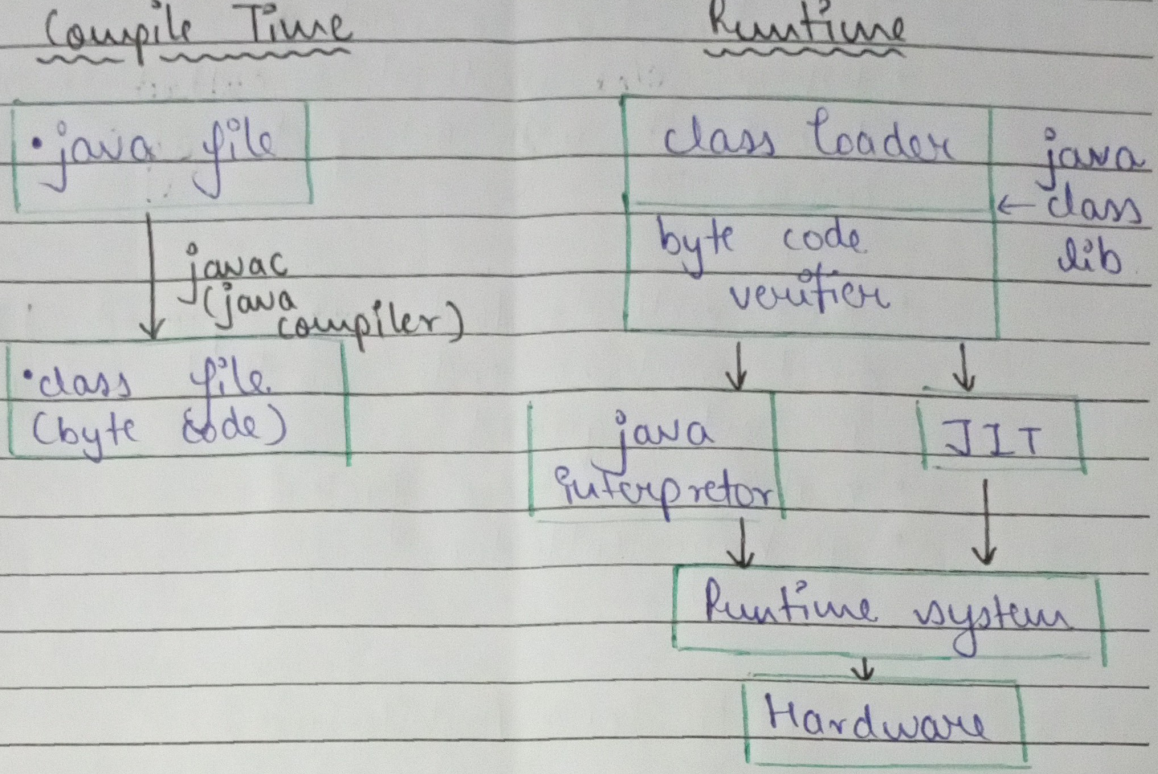
- → Java interpreter
- → garbage collector
- etc.

only run

- → development tools
- → Javac ↳ java compiler
- → archiver ↳ jar
- → docs generator ↳ Javadoc
- → interpretor / loader
- etc.

Date __/__/____ Page No. ____

* Java development and runtime env.:



* (How JVM works) class loader:

~~(1.) Linking:~~

(1.) Loading:
- read .class file and generate binary data
- an object of this class is created in heap

(2.) Linking:
- JVM verifies the .class file
- allocates memory for class variables and default values
- replace symbolic references from the type with direct reference

(3.) Initialization:
- all static variables are assigned with their values defined in the code and static block
- JVM contains the stack & heap memory locations

ABHINAV

Date ___/___/_____ Page No. _____

* JVM Execution:

(1.) Interpreter
- Line by line execution
- When one method is called many times it will interpret again and again

(2.) JIT
- Those methods that are replaced, JIT provides direct machine code so that interpretation is not required.
- makes execution faster
- garbage collector.

* Working of Java Architecture: