Java Architecture

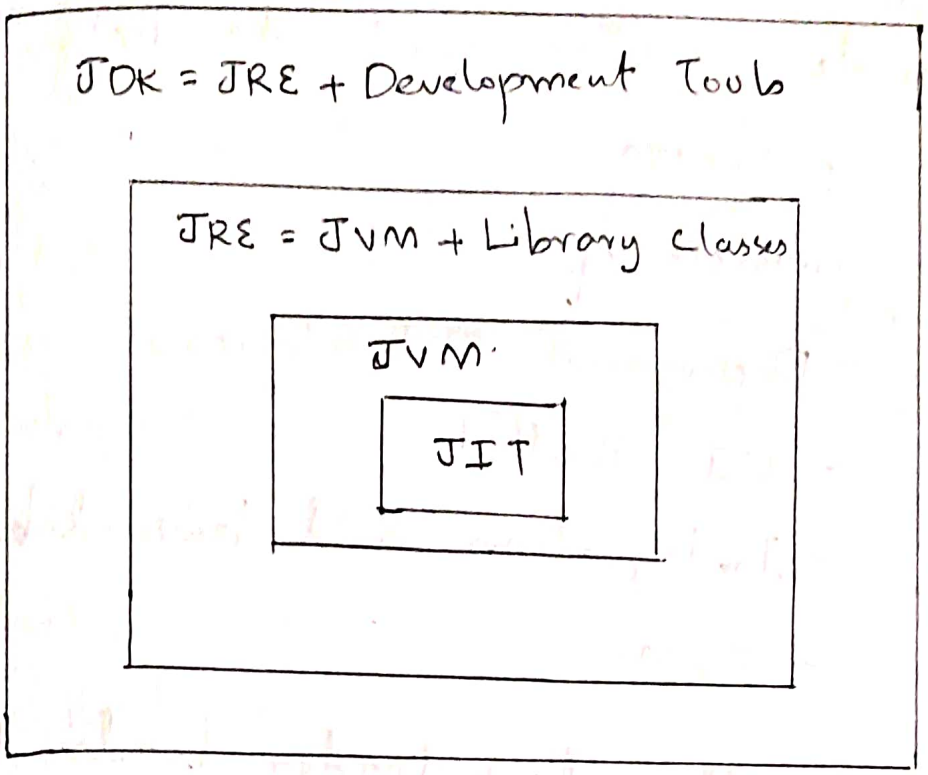
→ How the java code executes?

— First the source code gets compiled by javac and gets converted to bytecode in a file with extension .class. Now this is interpreted line by line and run using JVM. by converting to machine code.

→ Why is Java Platform Independent?

— Bytecode can run in any operating systems. Bytecode that is created from any compiler can be run from any JVM. JVM is platform dependent. So If an OS has a JVM installed, any Bytecode can run on this JVM. For C++, .exe file can only be run on same OS that it is compiled.

JDK vs JRE vs JVM vs JIT



→ what is JDK? what are its components?

- It is a cross-platformed SD environment that offers dev tools to develop Java-based software application.
- Consists of compiler javac, to compile source code to byte code.
- Consists of JRE to execute the program.
- Also has archiver (jar), docs generator (javadoc), interpreter / loader.

→ what is JRE and its components?

- It is an installation package that provides environment to only run the program.
- It consists of
  - Deployment Technologies
  - UI Toolkits
  - Integration and base libraries
  - JVM.
- Here the class loader loads the classes needed to execute programs and verifies the code format by bytecode verifier.

→ what is JVM, its working and Architecture?

- It is like a run-time engine to run Java Applications.
- This is the one who calls the entry point main method.

* class Loader Subsystem

- Loading

- class loader loads the class and generate binary data and stores it in method area of JVM memory along with fully qualified name of class and its immediate parent class, if file has class, interface or enum and variables, method info.

- Linking

  - Verifies bytecode
  - Allocates memory for class variables and provides default values.
  - replaces symbolic reference with direct reference.

- Initialization

  - static variables are assigned to their values.

ClassLoaders, Principle

- Bootstrap class loader
  - Loads trusted class, core java API classes present in JAVA_HOME/jre/lib
- Extension class loader
  - child of bootstrap loader.
  - Loads class in extension directory JAVA_HOME/jre/lib/ext
- System / Application classLoader
  - child of extension class loader.
  - Loads class from application class path.

* Principle used is Delegation-Heirarchy
  - System CL delegates to extension and it delegates to bootstrap
  - If Bootstrap finds, it load or it transfers to extension and if it

doesn't, transfers to system and if it doesn't, it throws a run-time exception ClassNotFoundException.

JIT and Interpreters

- Interpreter converts each bytecode instruction to native instruction (machine code) by looking to predefined JVM instruction to machine instruction mapping.
- For heavily used functions, instead of converting to native again and again, JIT helps in providing the converted version to JVM at run time.