PayPal Pre-Work Notes

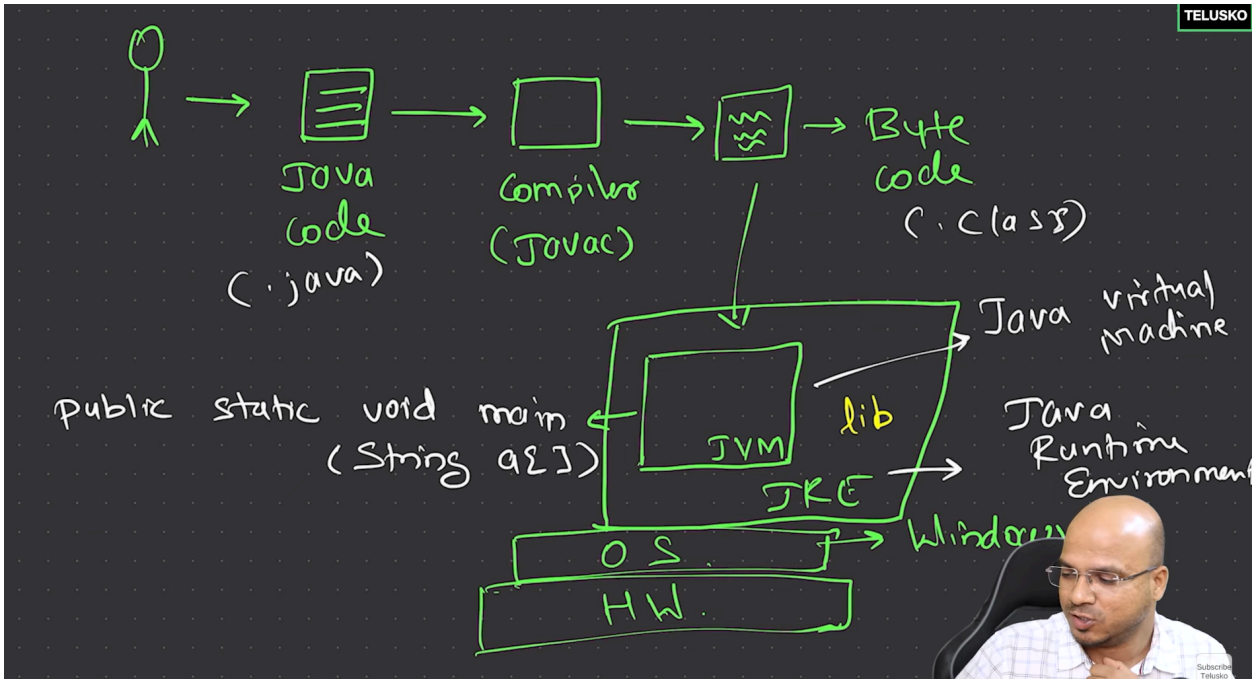
Java

Introduction

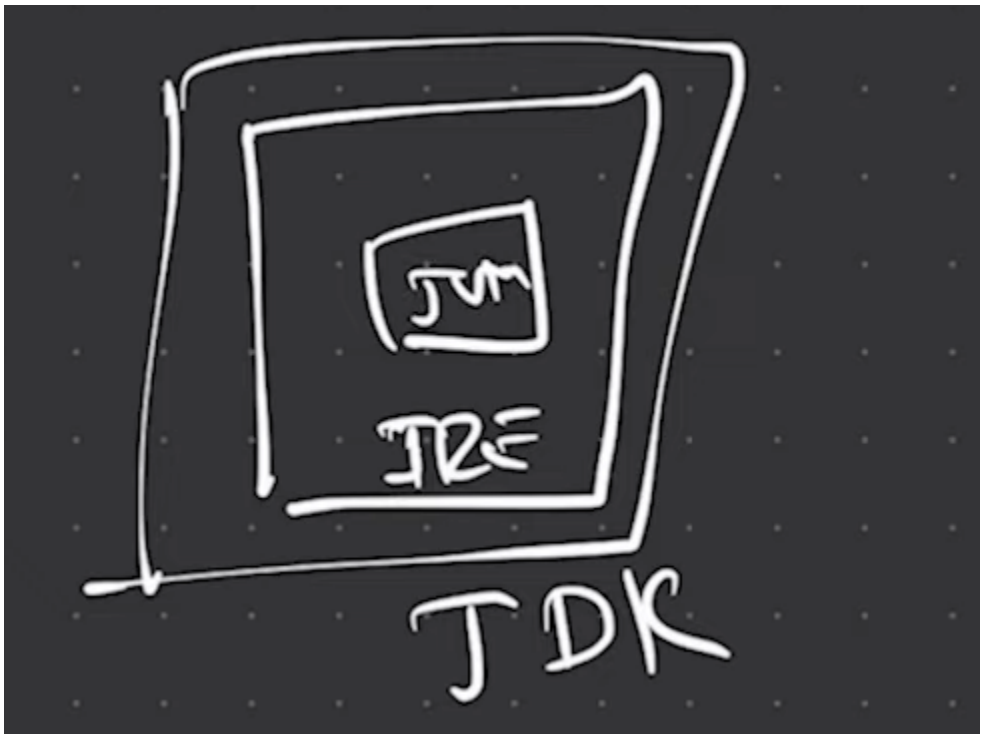
Java is a widely-used, object-oriented programming language renowned for its versatility and portability. Java follows the principle of "**Write Once, Run Anywhere" (WORA)**, meaning compiled Java code can execute on any platform equipped with a Java Virtual Machine (JVM). It is prized for its strong type system, automatic memory management via garbage collection, and extensive standard library, making it ideal for developing everything from desktop software to enterprise-level applications and mobile apps. Java's popularity stems from its reliability, scalability, and rich ecosystem of frameworks and tools that streamline development across diverse computing environments.

Multiple languages like kotlin, groovy are working on JVM, which is the same as working on Java. We need **java** to run the application and we need **javac**(java compiler) to compile the code.

Java Runtime

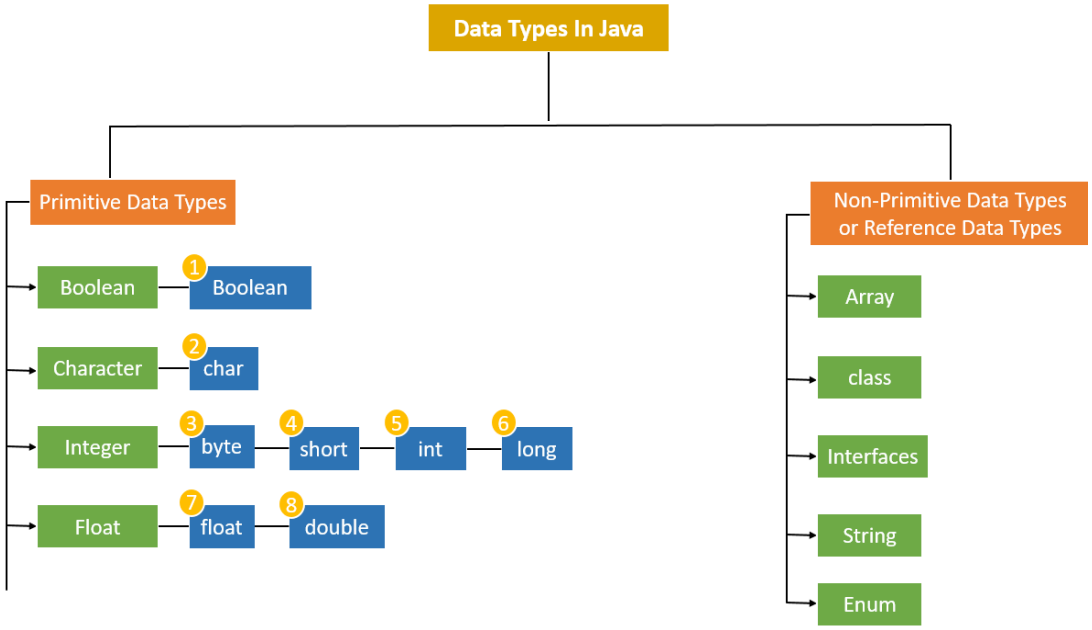


Java code is converted to **byte code(.class file)** via **compiler(javac)**, which runs on jvm(which is machine dependent, but makes java machine independent). Need to tell the main function to jvm, by listing out the main method. Since Java is Object oriented, we need to put all the code inside a class. JVM is a part of JRE(Java runtime environment), which provides support for libraries.



JRE is encompassed by JDK, which is used by developers to develop applications. But for someone who wants to use this application, they don't need to install JDK. They only need the JRE to run the application.

A **strongly typed language** is one in which types are enforced strictly, meaning that once a variable is declared to be of a certain type, it cannot be treated as another type without explicit conversion. **Java** is a strongly typed language.



The value assigned to a variable is called a **literal.** They can be int, float, boolean or null etc.

- **Type Conversion**: Automatic conversion from a smaller to a larger data type (e.g., `int` to `double`).
- **Type Casting**: Manual conversion from a larger to a smaller data type with possible data loss (e.g., `double` to `int`).

Java also utilizes the camel casing to name the variables and the functions.

Resources

Core Java	https://www.youtube.com/playlist?list=PLsyeobzWxl7pe_IiTfNyr55kwJPWbgxB5