دستورکار ۰ و ۱

- Introduction to java
- Introduction to git

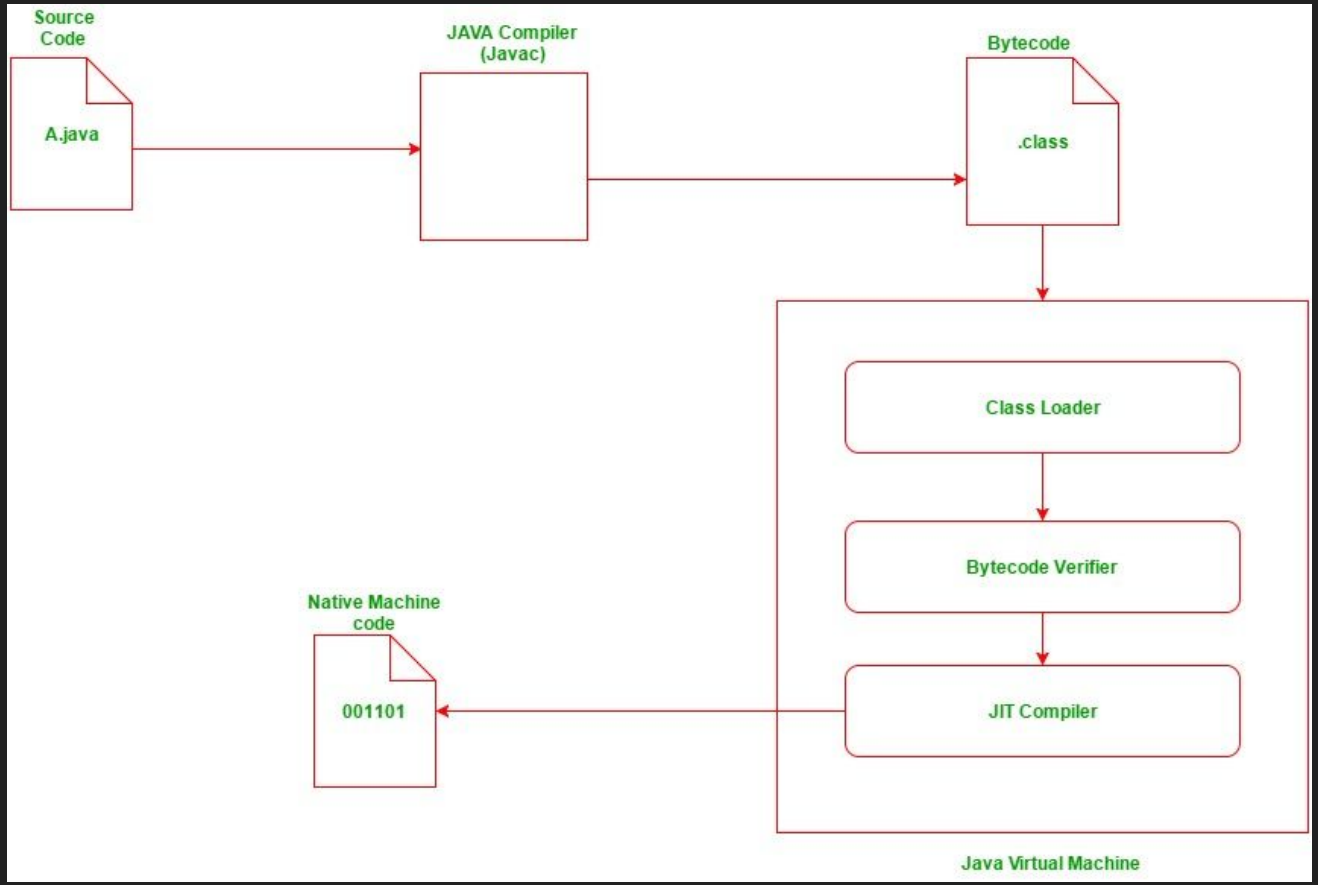
Java

Java

- Java is a popular programming language used for mobile apps, web apps, desktop apps, games and much more
- Java is a popular programming language, created in 1995. It is owned by Oracle, and more than 3 billion devices run Java.
- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.) or should we say it is cross-platform

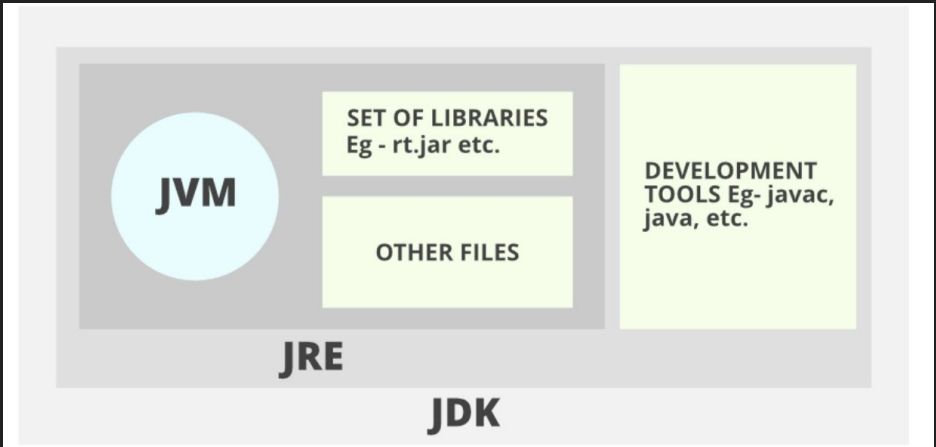
Java

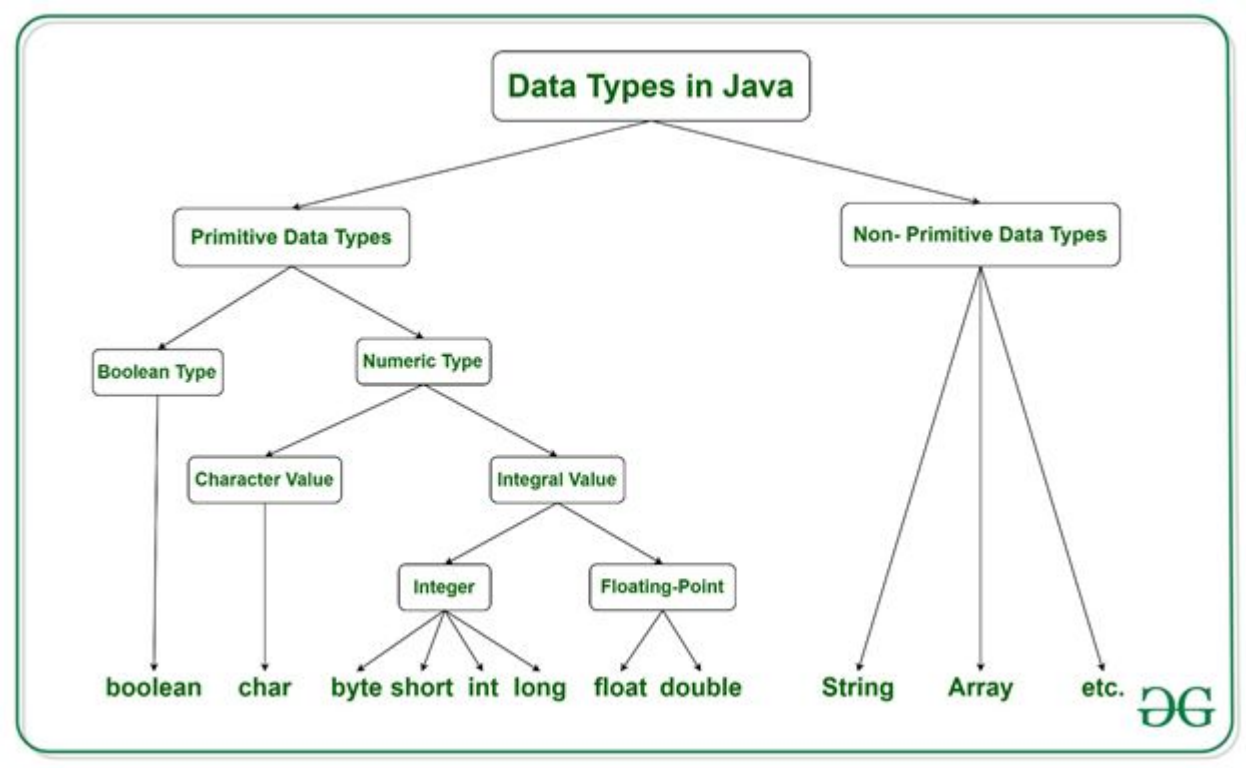
- Java is considered cross-platform because of its "write once, run anywhere" (WORA) principle. This means that Java programs, once written and compiled, can run on any platform or operating system that has a Java Virtual Machine (JVM) installed.
- The cross-platform nature of Java is primarily attributed to two key factors:
  - Java bytecode: an abstraction layer
  - JVM implementation for different platforms: implemented for various operating systems and hardware architectures
- Note: Java is case-sensitive: "MyClass" and "myclass" has different meaning.


Source Code
A.java
JAVA Compiler (Javac)
Bytecode
.class
Class Loader
Bytecode Verifier
JIT Compiler
Native Machine code
001101
Java Virtual Machine

JDK vs JRE vs JVM

- JDK: ابزاری است که امکان توسعه و اجرای یک برنامه جاوا را فراهم می‌کند.
- JRE: ابزاری است که شرایط را تنها برای اجرای یک برنامه جاوا فراهم می‌کند.
- JVM: بخش مهمی در JDK و JRE است و در هر دو وجود دارد.




Data Types in Java
Primitive Data Types
Non- Primitive Data Types
Boolean Type
Numeric Type
Character Value
Integral Value
Integer
Floating-Point
boolean
char
byte
short
int
long
float
double
String
Array
etc.

Java naming convention

- For classes, interfaces and abstract classes use pascal case:
  - For example: MyClass
- For objects and methods use camel case:
  - For example: oneSingleObjectName
- Notes:
  - Use meaningful names, don’t use k, p, l, …

git

Version Control Systems

- Version control systems are a category of software tools that helps in recording changes made to files by keeping a track of modifications done in the code.
  - Maintaining version history
- Rollback to the previous version if the current version does’nt work
- Developed by linus torvalds for managing linux development.
- Widely adopted now by several projects.
- Examples are git, CVS, SVN

  etc.

Basic Git commands

Check if git is installed

```
git --version
```

Configure global identity parameters (username and email address)

```
git config --global [user.email]
```

```
git config --global [user.name]
```

```
git config list (some of these configurations are automatically set up)
```

Basic git commands

Go to the project folder and initialize it to be a git repository

```
git init
```

This folder has been initialized to be a git repository, it will mark that folder as the branch master (we will talk about branches later)

```
git status
```

Shows files that are not added to the git repository, use this command to add them to the staging area

```
git add [file names or . to add all the files and folders in the folder]
```

To commit the changes use this command (git commit will commit the changes to the repository or in other words, it will add a new version of your application on top of all the versions you have commited):

```
git commit -m ‘ some commit message ’
```

Basic git commands

Now use git status to see the current status of the working directory is commited.

To see the log of the commits use the following command

```
git log --oneline
```

To get more details about each of the git commands use the - - help option for

Now lets do some coding :)

Basic git commands

- Pushing to online repository
  - git remote add origin https://github.com/username/repositoryName.git
  - git push origin master