Java Compilation Process (from my understanding)

Executing a Java program requires 2 steps: compilation and execution. During compilation, the main ‘.java’ file. is passed through the Java compiler, which is independent from the operating system. The compiler then converts the java codes into bytecodes, which is an instruction set for JVM in the next step. All public classes contained in the main file will change into a ‘.class’ file. These class files, however, are also independent of the operating system, hence it can be run on any system. During execution, the class files will be passed to JVM (Java Virtual Machine) to be loaded by the class loader, a subsystem of JVM. Class loader will use the loadclass function that fetches the class referenced by its class name in the form of a string from the bytecode. By default, the class loader will use primordial class loader, which is embedded into all the JVMs. After it is loaded, it is then checked by the bytecode verifier, which checks for codes that executes damaging actions such as violating rules of accessing private data. The last step is to convert the loaded bytecode into codes that can be run by the machine of the operating system. The Just-In-Time compiler (JIT compiler) does that job. It will then execute the instructions and run the final code.