1DV532 – Starting Out with Java

Lesson 3

Creating, Compiling, and Executing Java Programs

Dr. Nadeem Abbas
nadeem.abbas@lnu.se

What we do in Java!

1. **Write** a **source code (java program)** and save it as "**.java**" file

2. **Compile** the .java file (using **javac** compiler) to get ***bytecode*** (.class file)

3. **Run** the compiled **bytecode** (.class) on a Java **Virtual Machine (JVM)**

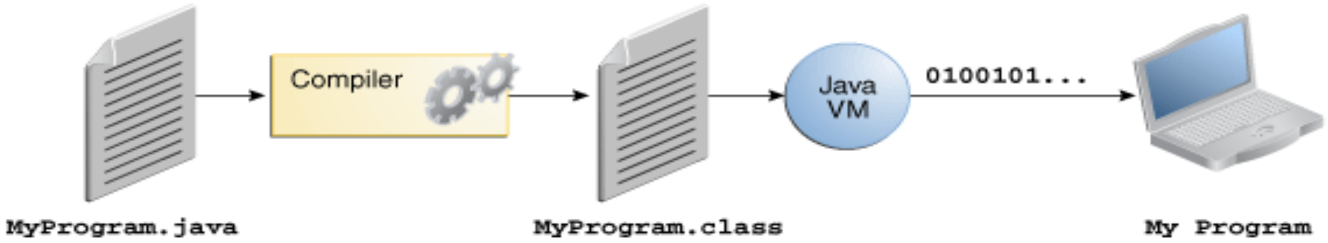


*Image taken from https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html

Java Example Program - Hello.java

```java
/**
 * Hello.java
 * Author: Nadeem Abbas
 */
package lecture1;   // package declaration

/*
 * A program that prints "Hello World"
 * on the screen.
 */
public class Hello {  // Class declaration

   public static void main(String[] args) { // main method
      /* Print to screen */
      System.out.println("Hello World!");
   }
}
```

Java Example Program - Hello.java

- Java programs are always saved in files ending with (the postfix) ***.java***
- Hello.java consists of a single ***class*** called *Hello*.
    - We will learn about ***classes*** in next step
- A **class *X*** must be saved in a file ***X.java***

```
public class A {
        // must be saved as A.java
}
```

- The class *Hello* belongs to the *package* lecture1
    - *Package* will be discussed later.

Java Example Program - Hello.java

- As shown in the example program, each Java program is made up of one or more *classes*
  - A class contains one or more ***methods***
  - *public static void main(String[] args)* is an example of a method in the example program
    - Methods in Java are comparable to functions in other languages, e.g. C.
- A method named **main** is the starting point of Java programs.
  - Each method contains program ***statements (instructions written in java language)***
  - We will learn more about methods in next step

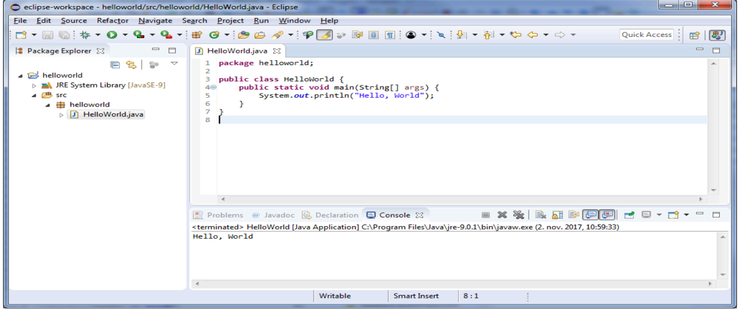
Java Example Program - Hello.java

- Java is case sensitive – *main* is not the same as *Main*.
- *System.out.println("Hello World!");*
  - prints Hello World! as one line on the screen.
- The curly brackets { and } denote start and end of code blocks.
- // class declaration is an end-of-line comment
- /* Print to screen */ is a block **comment**
- **Comments** are ignored by the compiler, i.e., they do not influence execution

Program Terminology

- ***Code*:** A program or a part of a program
- ***Source code* (or *source program*):** Complete program written in a high-level language such as Java
  - Input to the compiler program
- ***Object code*:** The translated low-level program
  - The output from the compiler program, e.g., **Java byte-code**

Creating, Compiling, and Running Programs



```
package helloworld;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World");
    }
}

```

```
<terminated> HelloWorld [Java Application] C:\Program Files\Java\jre-9.0.1\bin\javaw.exe (2. nov. 2017, 10:59:33)
Hello, World
```

Source code (developed by the programmer)

```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

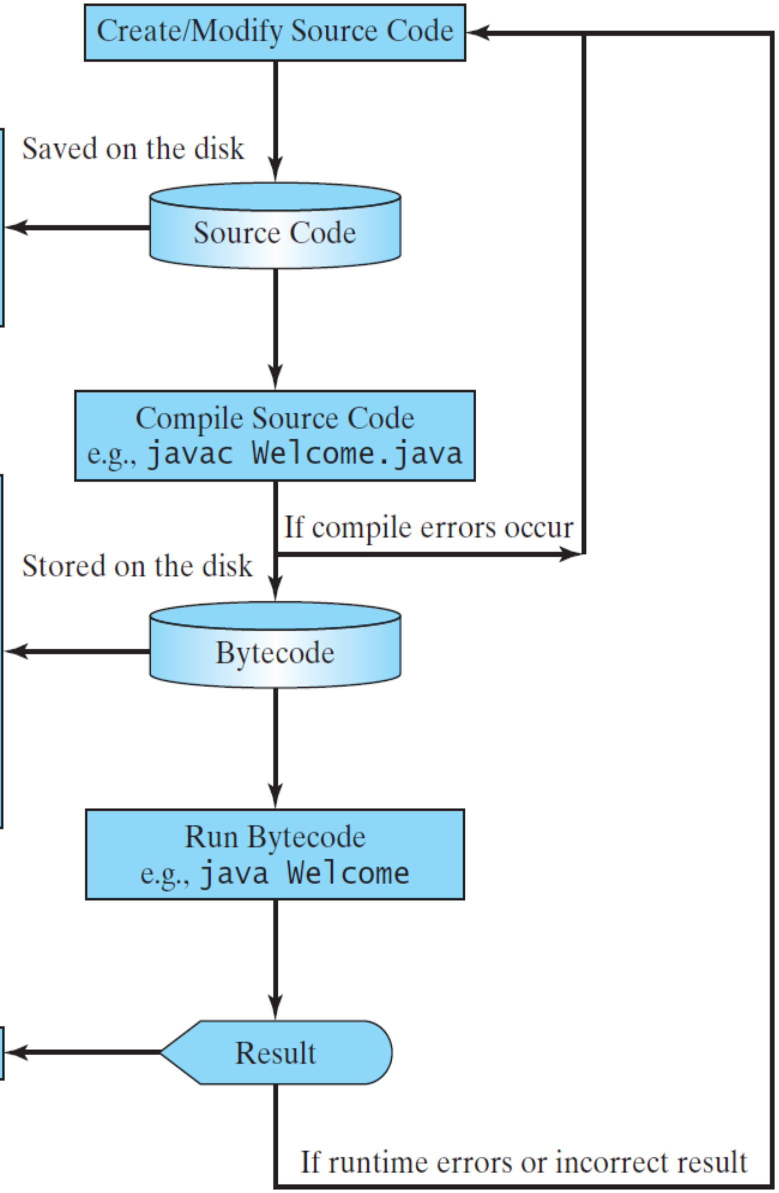
Bytecode (generated by the compiler for JVM to read and interpret)

```
…
Method Welcome()
  0 aload_0
  …

Method void main(java.lang.String[])
  0 getstatic #2 …
  3 ldc #3 <String "Welcome to Java!">
  5 invokevirtual #4 …
  8 return
```

“Welcome to Java ” is displayed on the console

```
Welcome to Java!
```



Compiling and Executing Java Code

- Java is a high-level programming language.
- As described in Lesson 1, programs written in a high-level language are not directly understandable by computers and need to be translated into machine code, a language understandable by computer.
- Translation from high-level to low-level or machine language is done by special software known as ***compiler***.

Compiling and Executing Java Code

- Every computer (processor or processor family) has its own machine language.

- Thus, to execute a program written in high-level languages on different machines (computers), we first need to ***recompile*** the program to translate it into machine specific language.
  - **This hurts portability of programs**

Compiling and Executing Java Code – **Byte Code**

- Java was originally designed for programming home appliances
  - Home appliances use wide variety of computer processors each having its own machine language
- Thus, writing a program that can run (without recompilation) on a number of difference processers was a challenge
  - Java solves this problem by providing a compiler that translates java source code to **byte code**
  - **Byte code** is similar to machine instructions but is architecture/ processor neutral and can run on any platform that has a **Java Virtual Machine (JVM)**

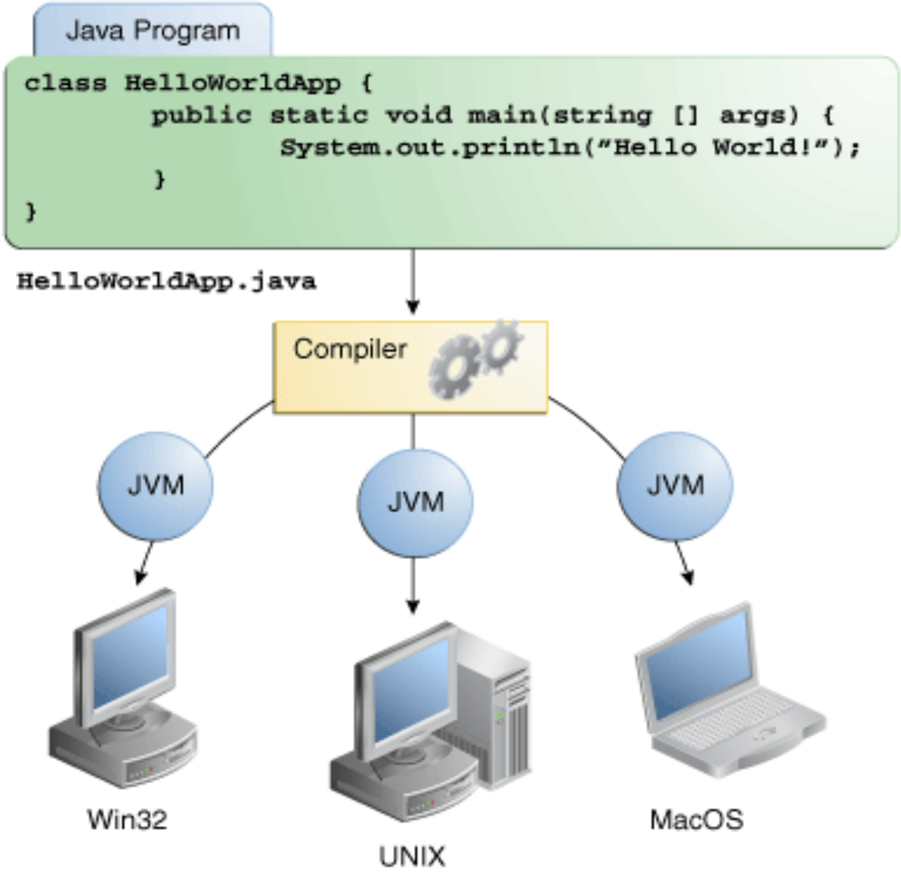
Compiling and Executing Java Code – **JVM**

- **Java Virtual Machine (JVM)** is a fictitious or conceptual machine, implemented in software, on which Java bytecode is executed

- It is an essential part of the Java Platform/JDK that enable java applications to become independent of any specific hardware and operating system

- **Write once – Run anywhere**
  - Use of the byte-code and JVM allows programmers to write and compile java programs once and run anywhere

Write once - Run anywhere



```
class HelloWorldApp {
        public static void main(string [] args) {
                System.out.println("Hello World!");
        }
}
```

*Image taken from https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html

Compiling and Executing Java Code

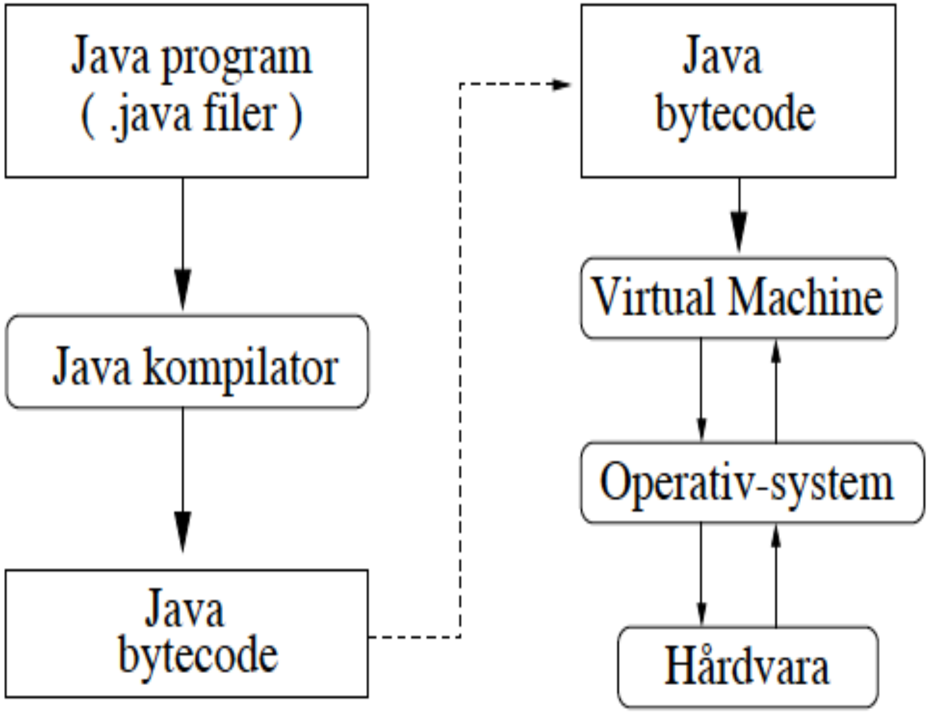
- **Command line compilation**

  ```
  C:\temp> javac Hello.java
  ```

  – Above command produces Hello.class file which contains byte code)

- **Command line execution**

  ```
  C:\temp> java Hello [args]
  ```



Creating Compiling and Executing Java Code using IDEs

- Step by step instructions to create, compile and execute java programs using **NetBeans** and **Eclipse** are given under the Sections 1.11 and 1.12 (pages 45– 50) of the course Book B2: "*Introduction to Java Programming, Brief Version, Global Edition, 11/E*" by Y. Daniel Liang

Suggested Readings

- Absolute Java, Global Edition, 6/E by Walter J. Savitch, Chapter 1
- Introduction to Java Programming, Brief Version, Global Edition, 11/E Liang, Chapter 1
- Java Tuotrials:
  - **The Java Technology Phenomenon** https://docs.oracle.com/javase/tutorial/getStarted/intro/index.html
  - **The "Hello World!" Application** https://docs.oracle.com/javase/tutorial/getStarted/cupojava/index.html
  - **Common Problems (and Their Solutions)** https://docs.oracle.com/javase/tutorial/getStarted/problems/index.html

Lnu.se