CHAPTER 1

Introduction to Java Software Development

Shin-Jie Lee (李信杰)
Associate Professor
Computer and Network Center
Department of Computer Science and Information Engineering
National Cheng Kung University

Origins of the Java Language

- Created by Sun Microsystems team led by James Gosling (1991)
  - Originally designed for programming home appliances
    - Difficult task because appliances are controlled by a wide variety of computer processors
    - Team developed a two-step translation process to simplify the task of compiler writing for each class of appliances

Major release versions of Java

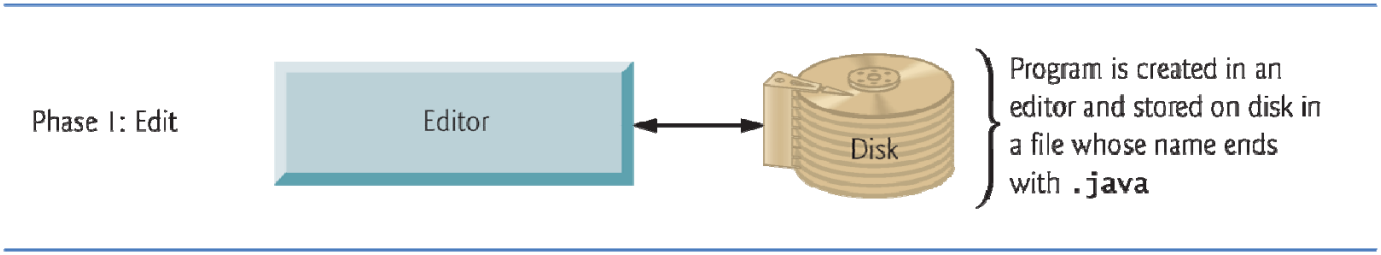
- JDK 1.0 (January 21, 1996)
- JDK 1.1 (February 19, 1997)
- J2SE 1.2 (December 8, 1998)
- J2SE 1.3 (May 8, 2000)
- J2SE 1.4 (February 6, 2002)
- J2SE 5.0 (September 30, 2004)
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)
- Java SE 8 (March 18, 2014)
- Java SE 9 (Sep 21, 2017)

Typical Java Development Environment

- Java programs normally go through five phases
  - edit
  - compile
  - load
  - verify
  - execute

Typical Java Development Environment

- Phase 1 consists of editing a file
  - Type a Java program (source code) using the editor.
  - Make any necessary corrections.
  - Save the program.
  - A file name ending with the .java extension indicates that the file contains Java source code.

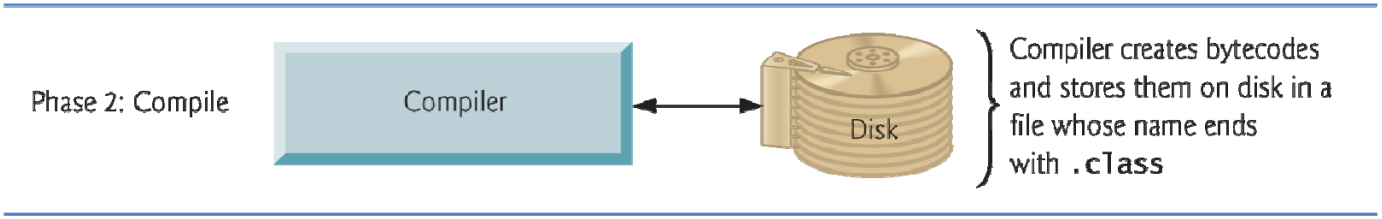


Typical Java Development Environment

- Linux editors: `vi` and `emacs`.
- Windows editors:
  - Notepad
  - EditPlus (`www.editplus.com`)
  - TextPad (`www.textpad.com`)
  - jEdit (`www.jedit.org`).
- Integrated development environments (IDEs)
  - Provide tools that support the software development process, including editors for writing and editing programs and debuggers for locating logic errors
    - E.g. Eclipse (`www.eclipse.org`)

Typical Java Development Environment

- Phase 2: Compiling a Java Program into Bytecodes
  - Use the command javac (the Java compiler) to compile a program. For example, to compile a program called `Welcome.java`, you'd type
    - `javac Welcome.java`
  - If the program compiles, the compiler produces a .class file called `Welcome.class` that contains the compiled version of the program.

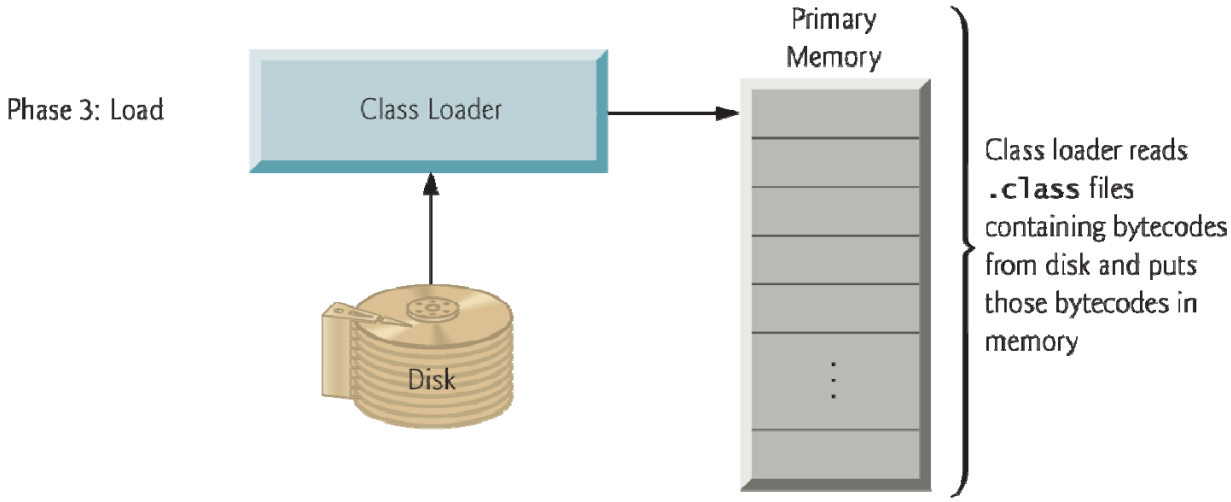


Typical Java Development Environment

- Java compiler translates Java source code into bytecodes that represent the tasks to execute.
  - Bytecodes are platform independent
  - Bytecodes are executed by the Java Virtual Machine (JVM)—a part of the JDK and the foundation of the Java platform.
    - `java Welcome`

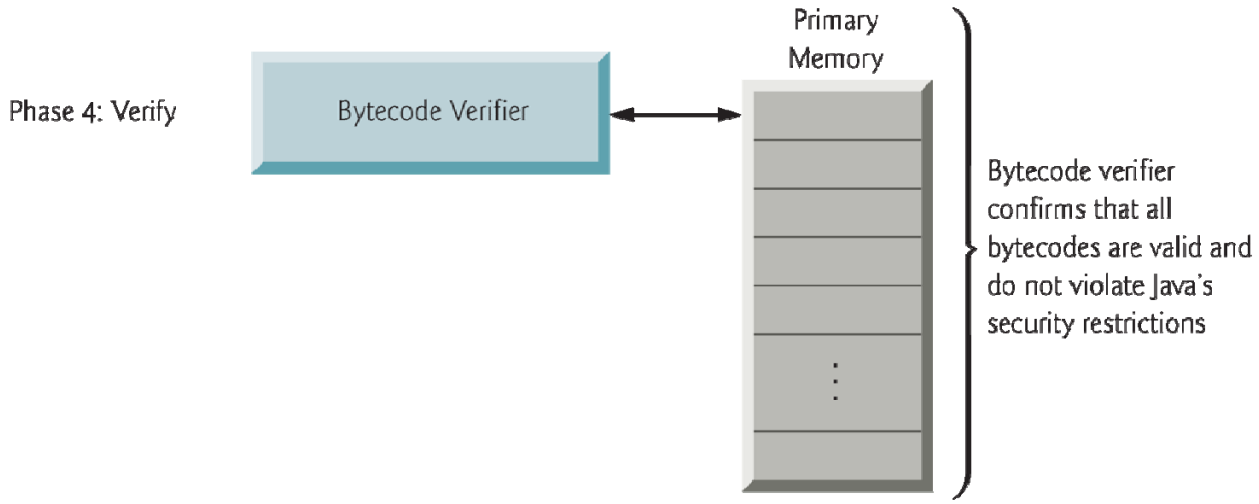
Typical Java Development Environment

- Phase 3: Loading a Program into Memory
  - The JVM places the program in memory to execute it—this is known as loading.
  - Class loader takes the `.class` files containing the program's bytecodes and transfers them to primary memory.
  - Also loads any of the `.class` files provided by Java that your program uses.



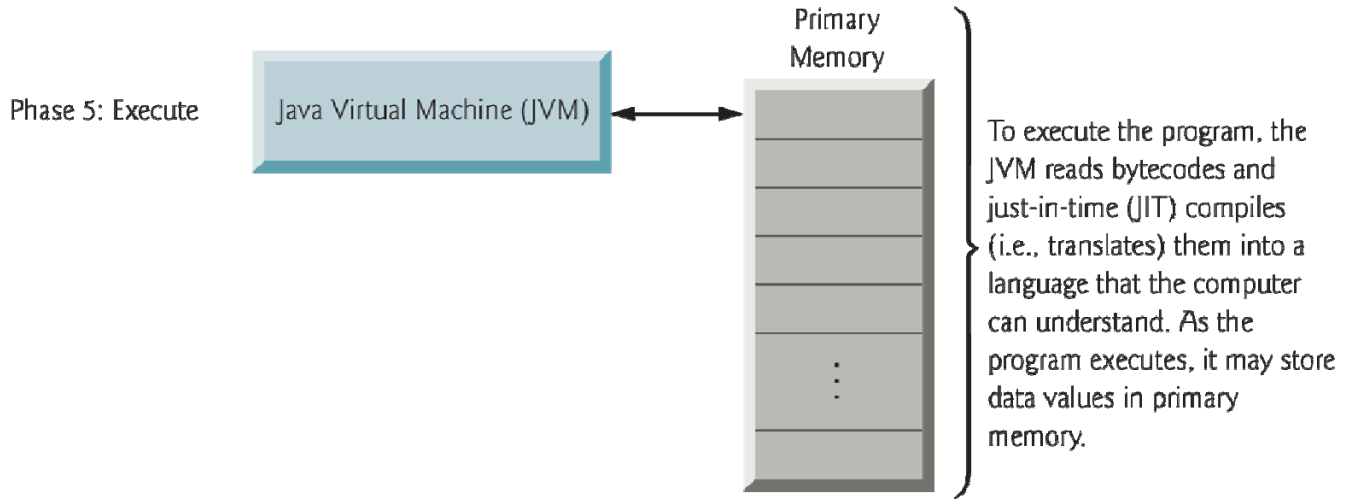
Typical Java Development Environment

- Phase 4: Bytecode Verification
  - As the classes are loaded, the bytecode verifier examines their bytecodes
  - Ensures that they're valid and do not violate Java's security restrictions.



Typical Java Development Environment

- Phase 5: Execution
  - The JVM executes the program's bytecodes.
  - JVMs typically execute bytecodes using a combination of interpretation and so-called just-in-time (JIT) compilation.
  - Analyzes the bytecodes as they're interpreted
  - A just-in-time (JIT) compiler—known as the Java HotSpot compiler—translates the bytecodes into the underlying computer's machine language.



Installing the JDK

- The **javac.exe** program is the compiler, which means it's the program that turns code you can read (the code you write in Java) into code your computer can read (the collection of 0s and 1s that a computer needs when it runs a program).
- The **java.exe** program runs the programs that you write.

Installing the JDK

- Get the latest version of the JDK, follow these steps:
  1. Open **http://www.oracle.com/technetwork/java/javase/downloads/index.html** in a web browser.
  2. Click the Download JDK button.
  3. Follow the instructions provided by the web site.
  4. Run the installer and accept any defaults.

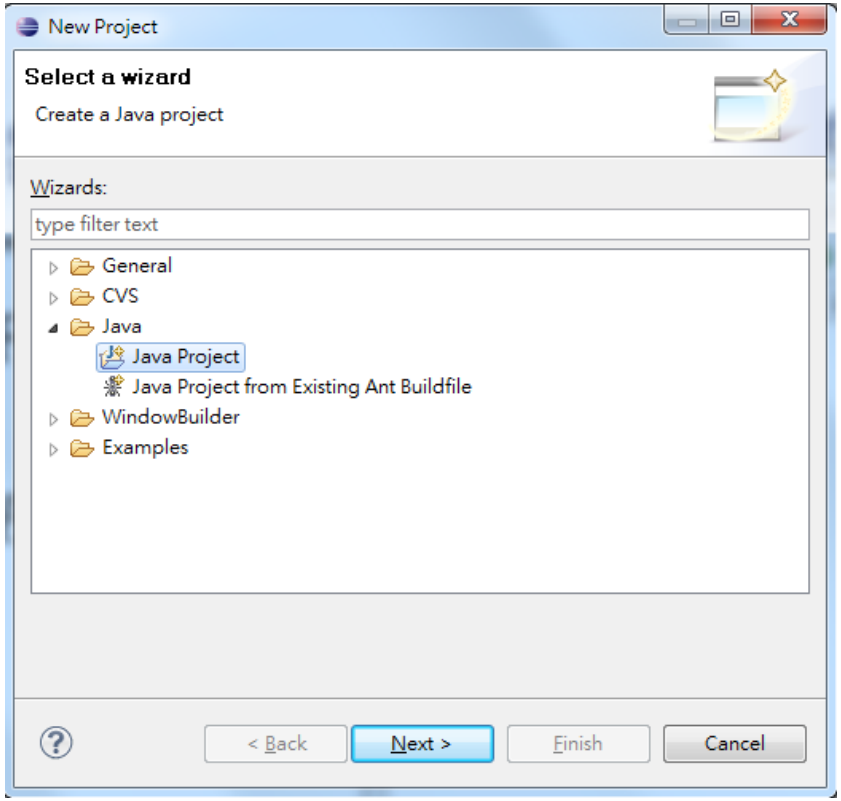
Installing Eclipse

- Before you can install Eclipse, you have to download it. To do so, follow these steps:
  - 1. Open **http://www.eclipse.org/downloads/** in a web browser.
  - 2. Find the **Eclipse IDE for Java Developers** choice.
  - 3. Follow the instructions provided by the web site.
  - 4. Run the installer and accept any defaults.

Creating Your First Project

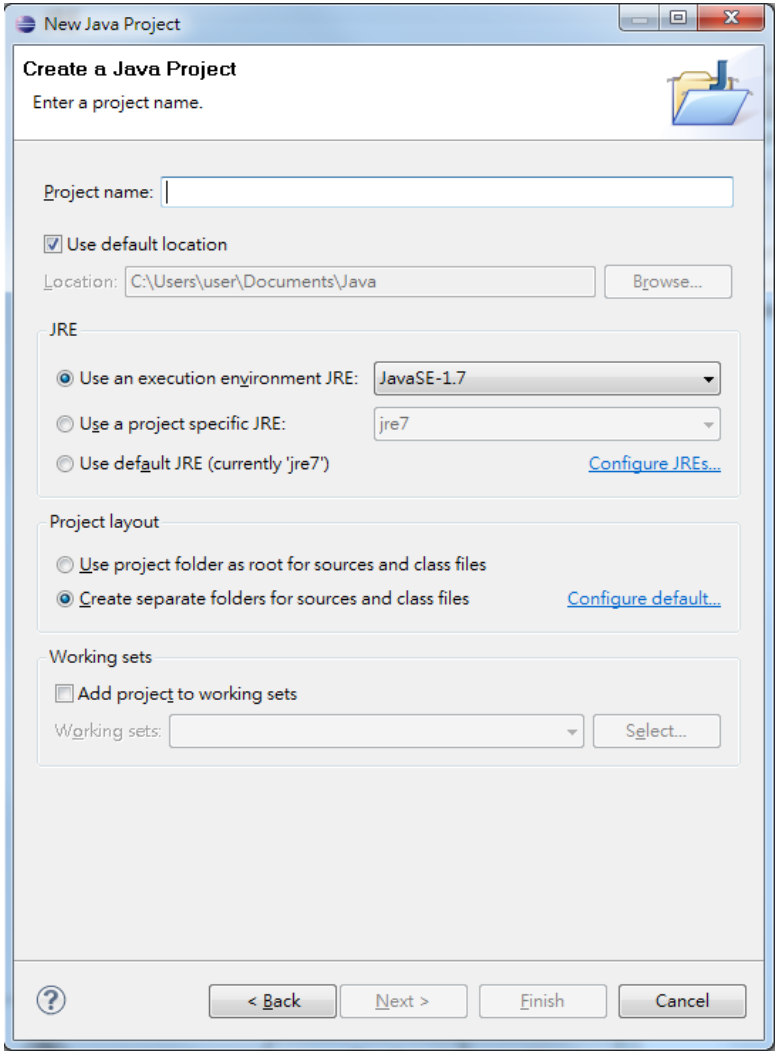
- 1. From the File menu, select New, and then select Project.

Eclipse's New Project window.



Creating Your First Project

- 2. In the New Project window, double-click Java Project.

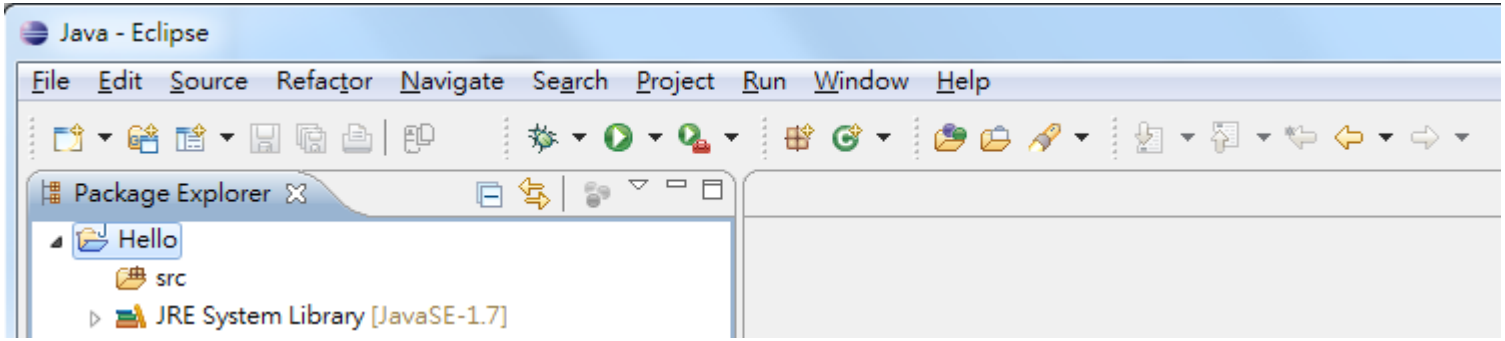


Eclipse's New Java Project window.

Creating Your First Project

- 3. Type **Hello** in the Project name field.
- 4. Click OK.

The main area of the Eclipse IDE.

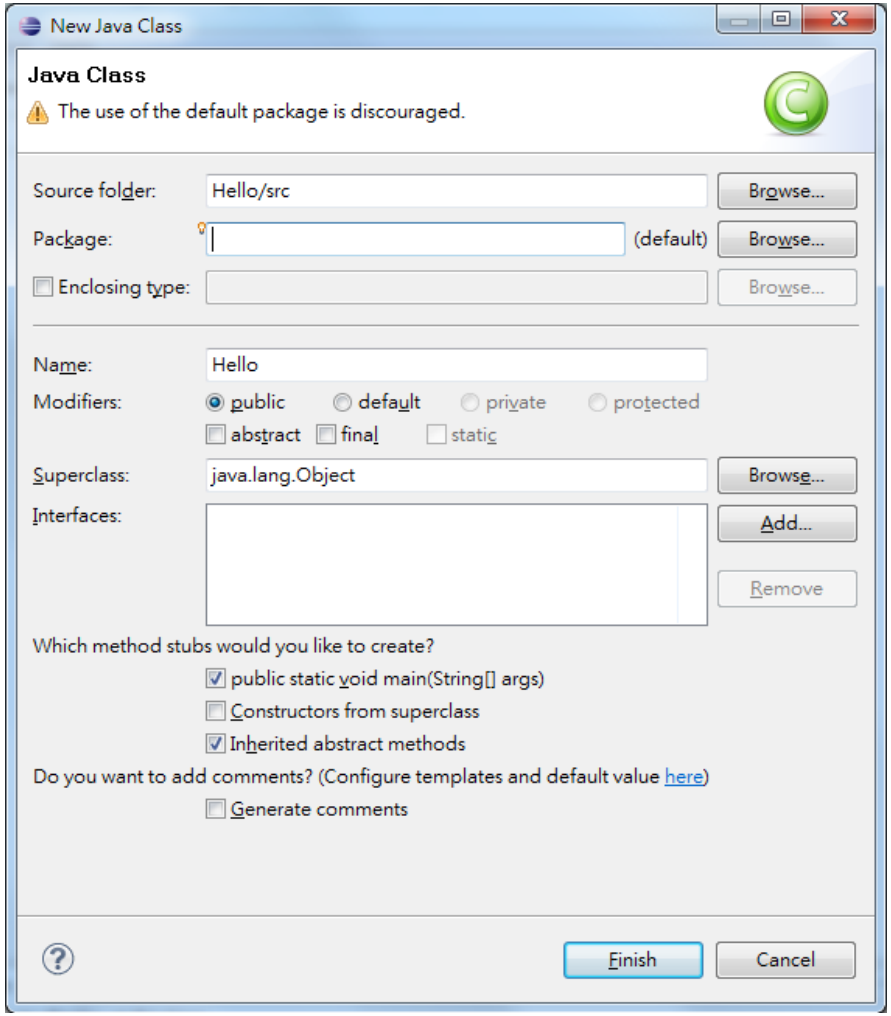


Creating the Program

- To create a class with a main method for your first program, follow these steps:
  - 1. Right-click the **Hello** project in the Eclipse Package Explorer, choose New, and then choose Class.

Creating the Program

Eclipse's New Java Class window.



Creating the Program

- 2. In the Package field, type whatever you like for the package.
- 3. Check the checkbox that gives you a **main** method (**public static void main (String args[])**).

Creating the Program

Example: Preliminary Hello class

```
public class Hello {
	/**
	* @param args
	*/
	public static void main(String[] args) {
		// TODO Auto-generated method stub

	}
}
```

Creating the Program

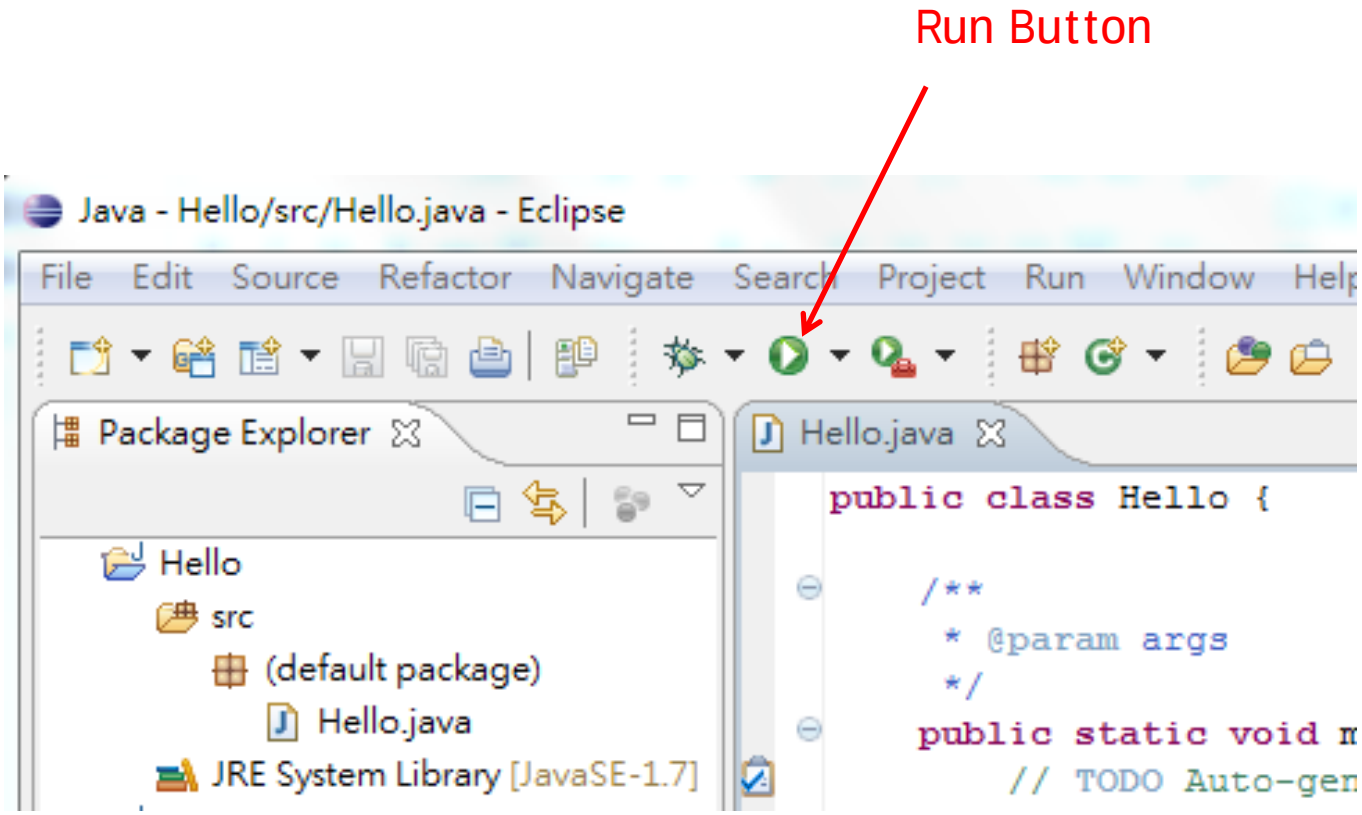
- 4. Within the **main** method, type:
  System.out.println(“Hello, World!”);

Example: Basic Hello program

```
public class Hello {

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

Run the Program



Adding More Functionality

- The **args** array holds all the values that were provided to the Java runtime engine when someone started your program.

Example: Reading arguments

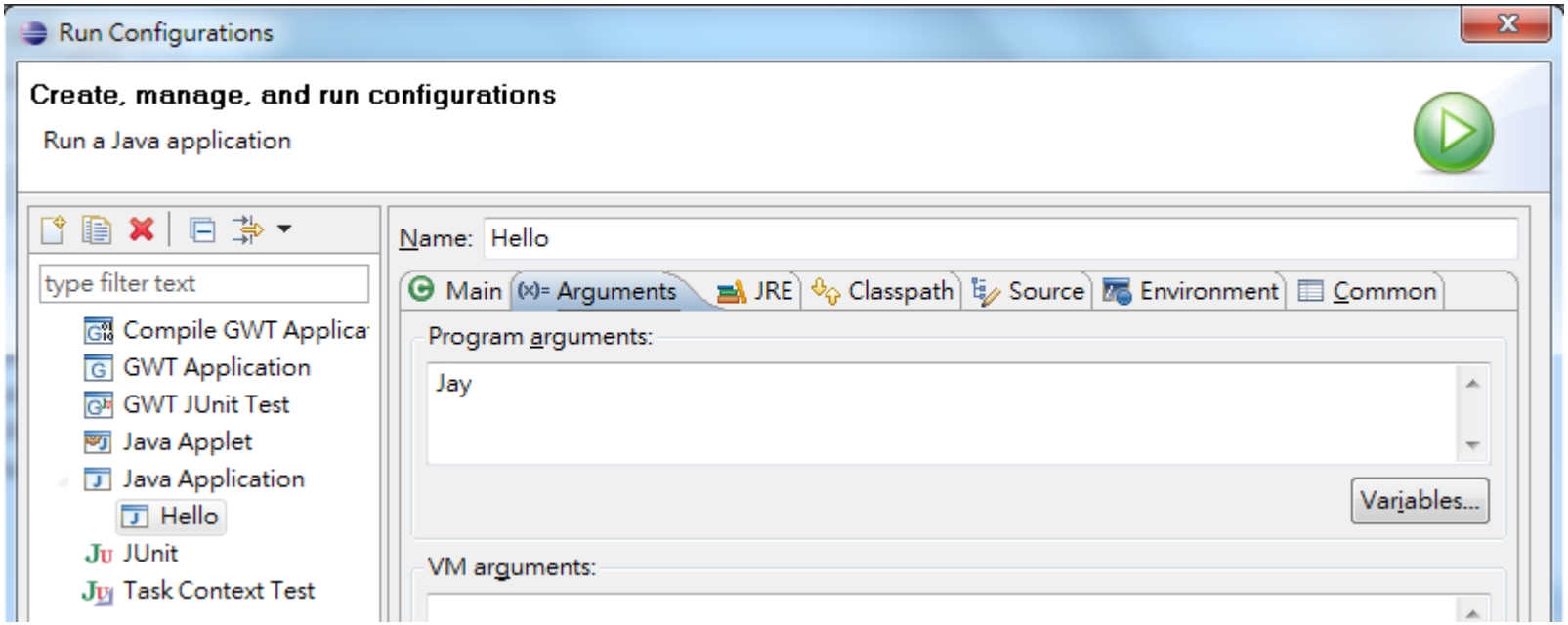
```
public class Hello {
        public static void main(String[] args) {
                System.out.println("Hello, " +
                        args[0] + "!");
        }

}
```

Adding More Functionality

- 1. From the Run menu, choose Run Configurations.

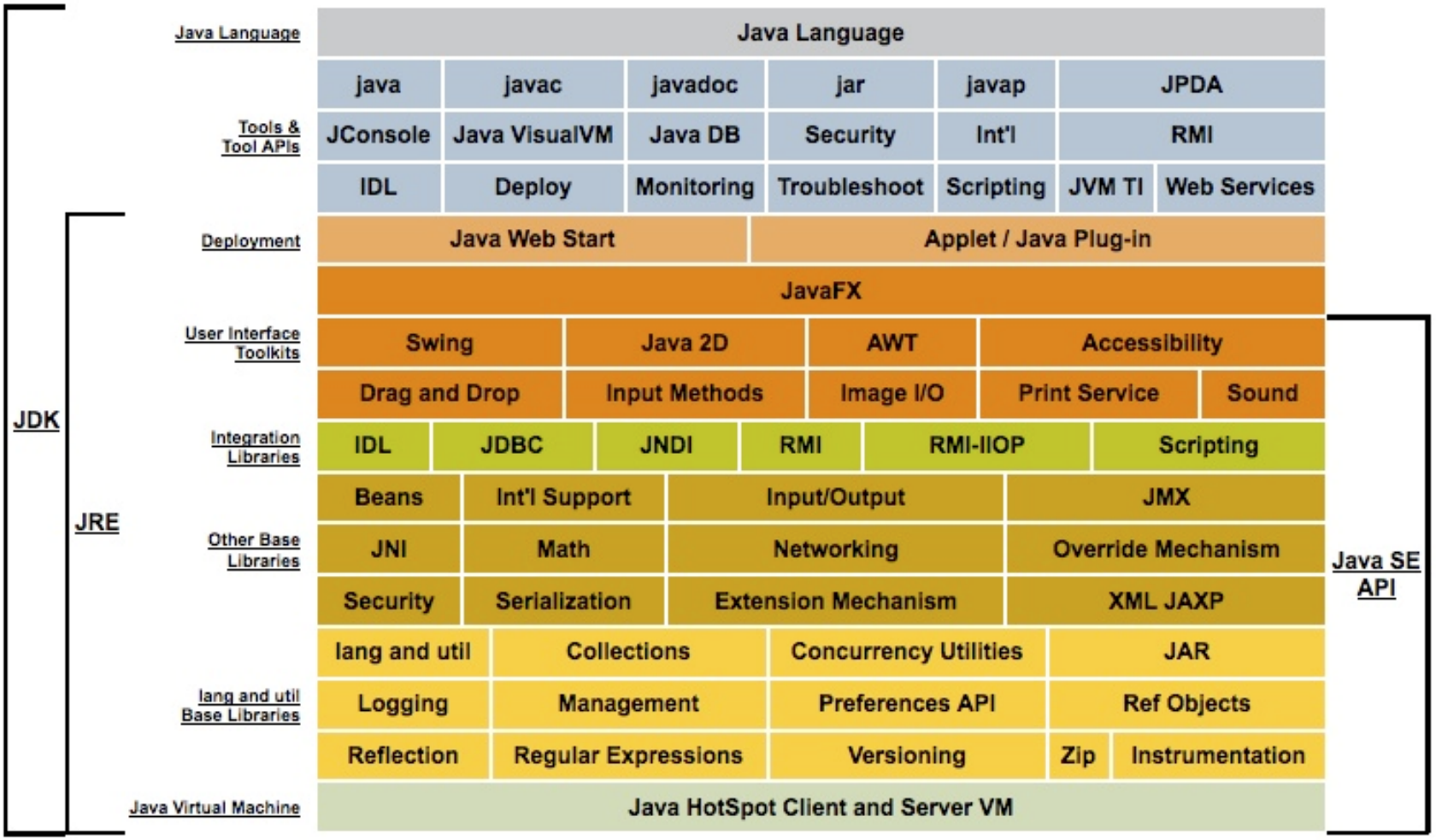
The Run Configurations window.



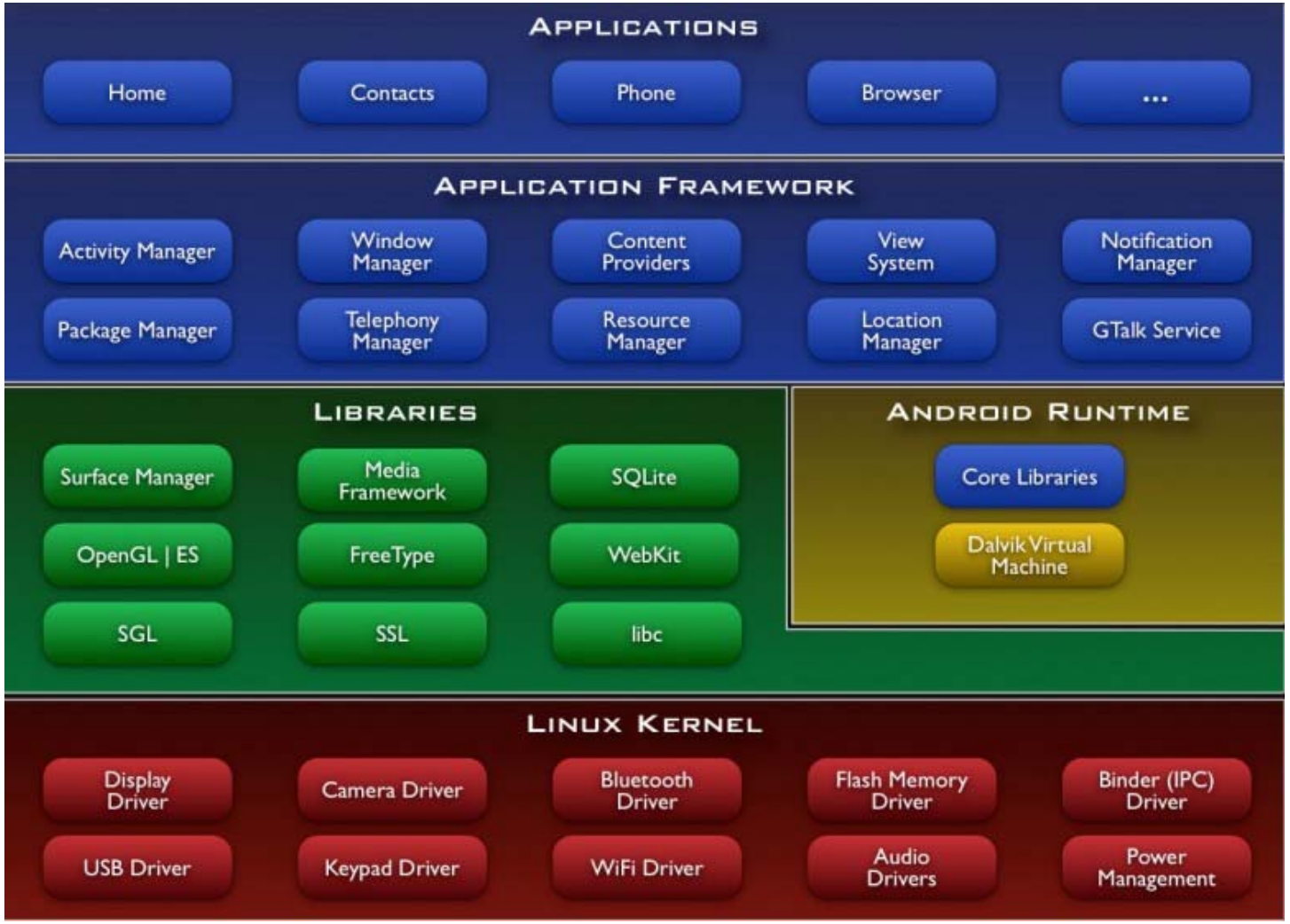
- 2. In the **Arguments** tab, type your name.
- 3. Click the **Run** button.

Java APP = Android APP ?

Java Framework



Android Framework



Reference

- "Absolute Java". Walter Savitch and Kenrick Mock. Addison-Wesley; 5 edition. 2012
- "Java How to Program". Paul Deitel and Harvey Deitel. Prentice Hall; 9 edition. 2011.
- "Java 7 for Absolute Beginners". Jay Bryant. Apress; 1 edition. 2011.