Getting to know your Java Programming Environment

Sang Shin
JPassion.com
“Code with Passion!”

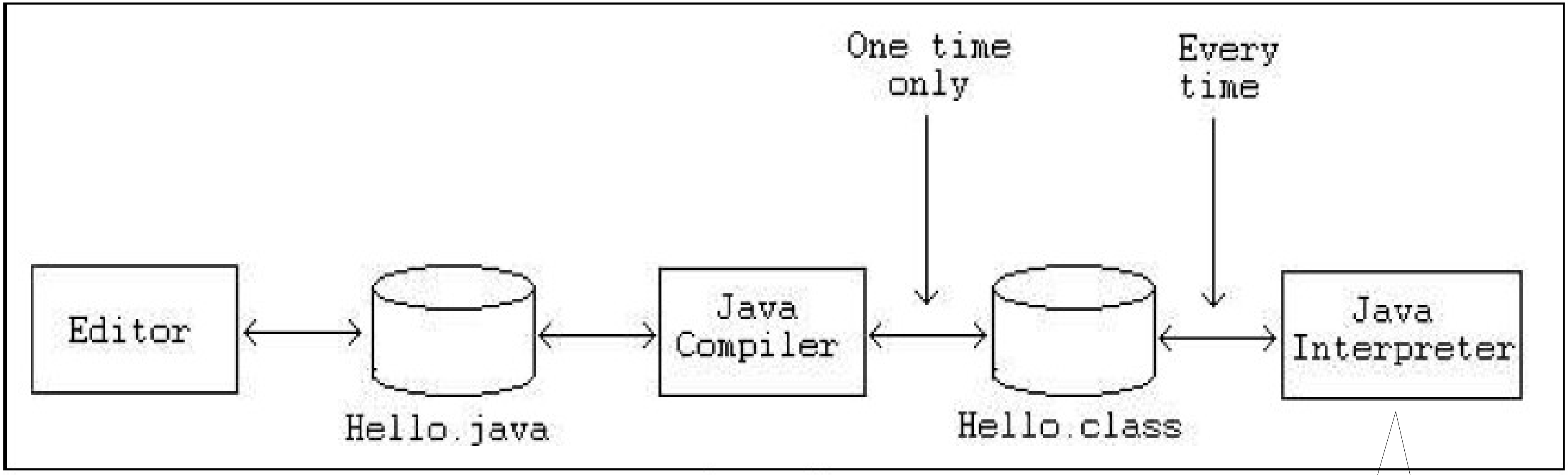
Topics

- Java Program Development
- Build and run a Java program using IDE (NetBeans or Eclipse)
- Build and run a Java program using editor and JDK command line tools
- Differentiate between syntax-errors and runtime errors

Java Program Development

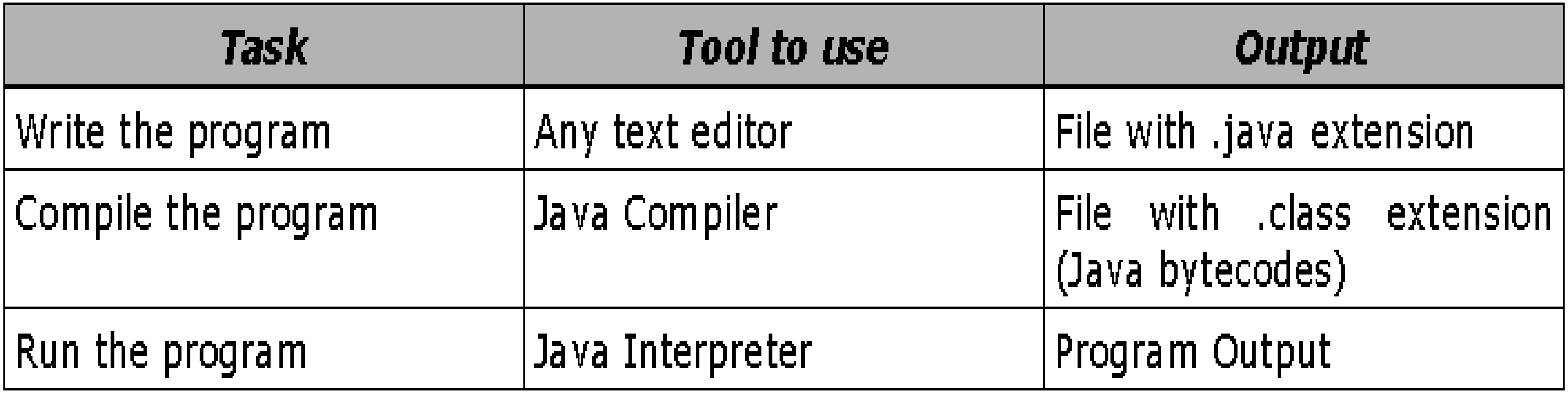
Java Program Development Steps

- The following figure describes the process of compiling and executing a Java program



Java runtime

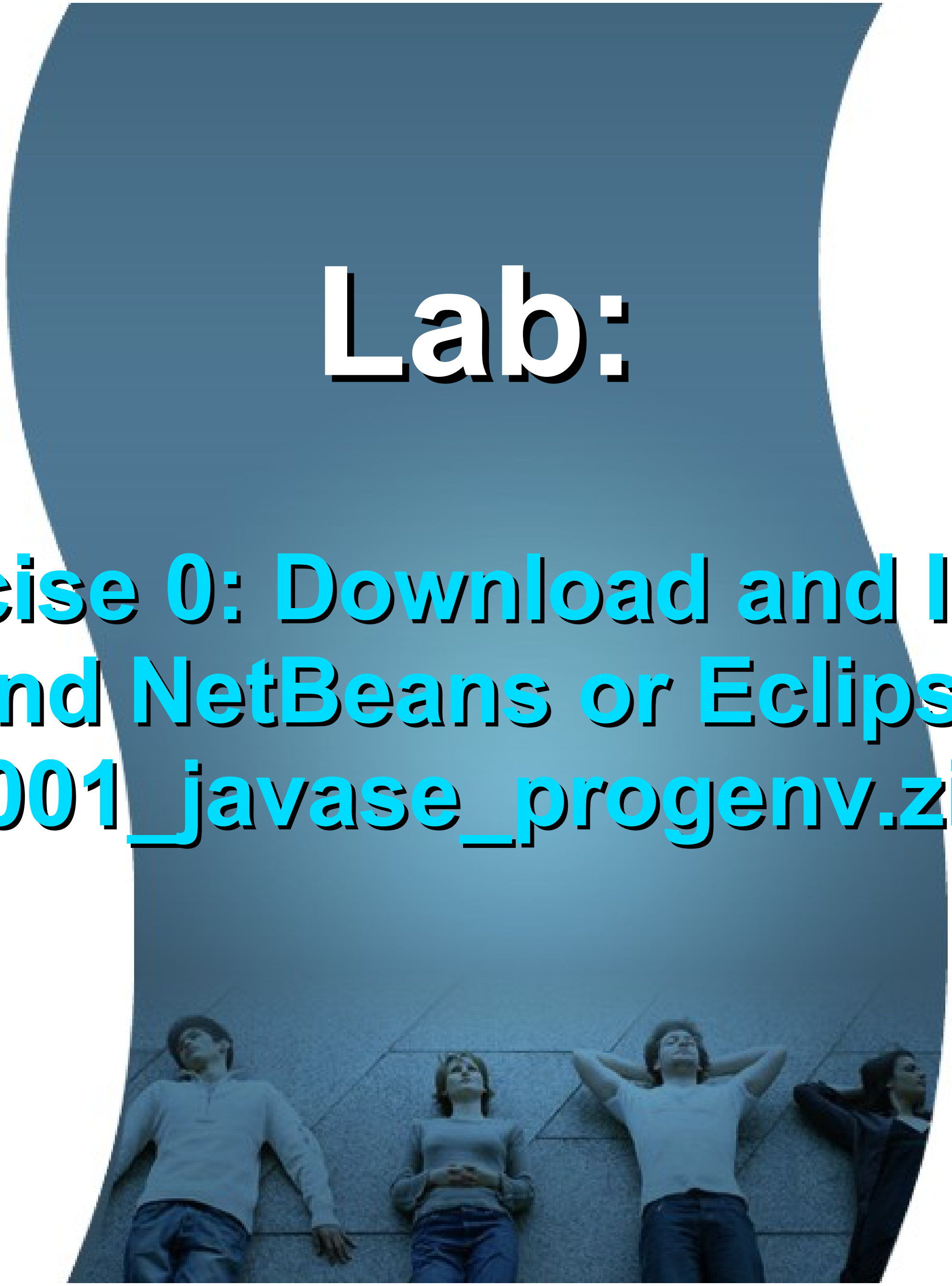
Java Program Development Steps



Task	*Tool to use*	*Output*
Write the program	Any text editor	File with .java extension
Compile the program	Java Compiler	File with .class extension (Java bytecodes)
Run the program	Java Interpreter	Program Output

Java Development Environment

- Option #1 – Use your editor of your choice for coding and then use command line Java tools from JDK such as “javac” (compiler) for compiling and “java” (Java runtime) for execution
- Option #2 – Use IDE (Integrated Development Environment)
  - NetBeans
  - Eclipse (for Eclipse-variant such as Spring Tool Suite, STS in short)
  - Intellij IDEA
- In general, IDE provides much more pleasant and productive development environment, so all Java developers use IDE of his/her choice for Java programming
  - For this course, you can use either NetBeans or Eclipse



Lab:

Exercise 0: Download and Install JDK and NetBeans or Eclipse/STS 1001_javase_progenv.zip

Build and run Java Program using Command-line Tools

Tools to Use

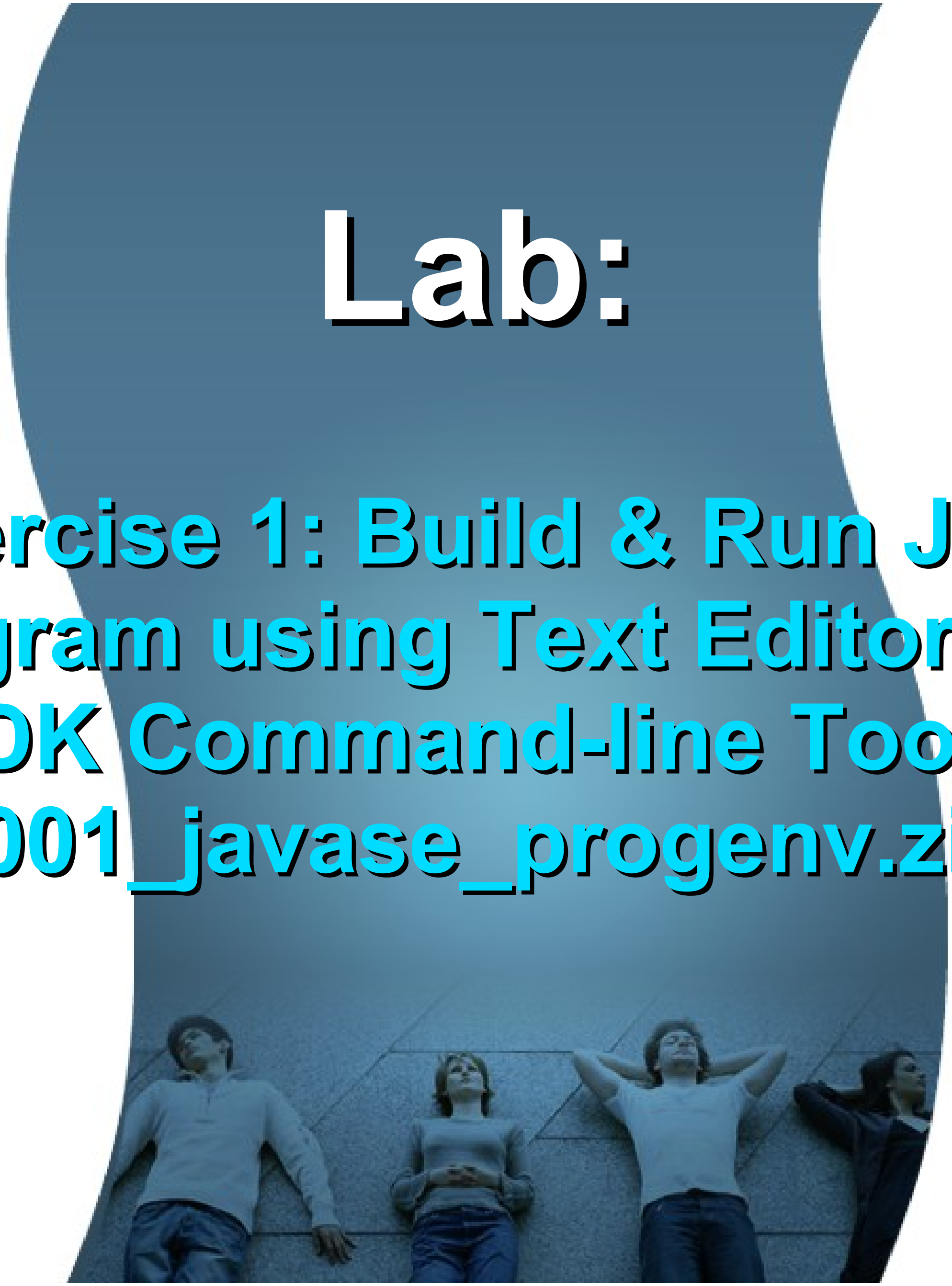
- Text Editor
  - Examples: Sublime Text, Textmate (Mac only), Notepad (Windows only), Vi or VIM, Emacs
- Java command line tools – are available <JDK>/bin directory
  - *javac* – compiler, used for compilation
  - *java* – java runtime (java interpreter), used for execution

My First “Helloworld” Java Program

```
public class Hello {

        /**
         * My first Java program
         */
        public static void main( String[] args ){

                //prints the string “Hello world” on screen
                System.out.println(“Hello world”);

        }
}
```

Use Text Editor & Command-line Tools

- Step 1: Using text editor, write Java program
  - Name it “Hello.java”
- Step 2: Compile the Java program
  - *javac Hello.java*
- Step 3: Run the Java program
  - *java Hello*



Lab:

Exercise 1: Build & Run Java Program using Text Editor and JDK Command-line Tools 1001_javase_progenv.zip

Build and run Java Program using IDE

What is IDE?

- Integrated Development Environment (IDE) provides complete application development environment
  - Code editor (with many conveniences such as syntax highlighting)
  - GUI builder (for building GUI application)
  - Built-in compiler
  - Built-in debugger
  - Built-in tester
  - Built-in profiler (for performance measurement)
  - Many more built-in features
- Popular IDE's
  - Eclipse or STS (Free), NetBeans (Free), IntelliJ IDEA (Free/Commercial)

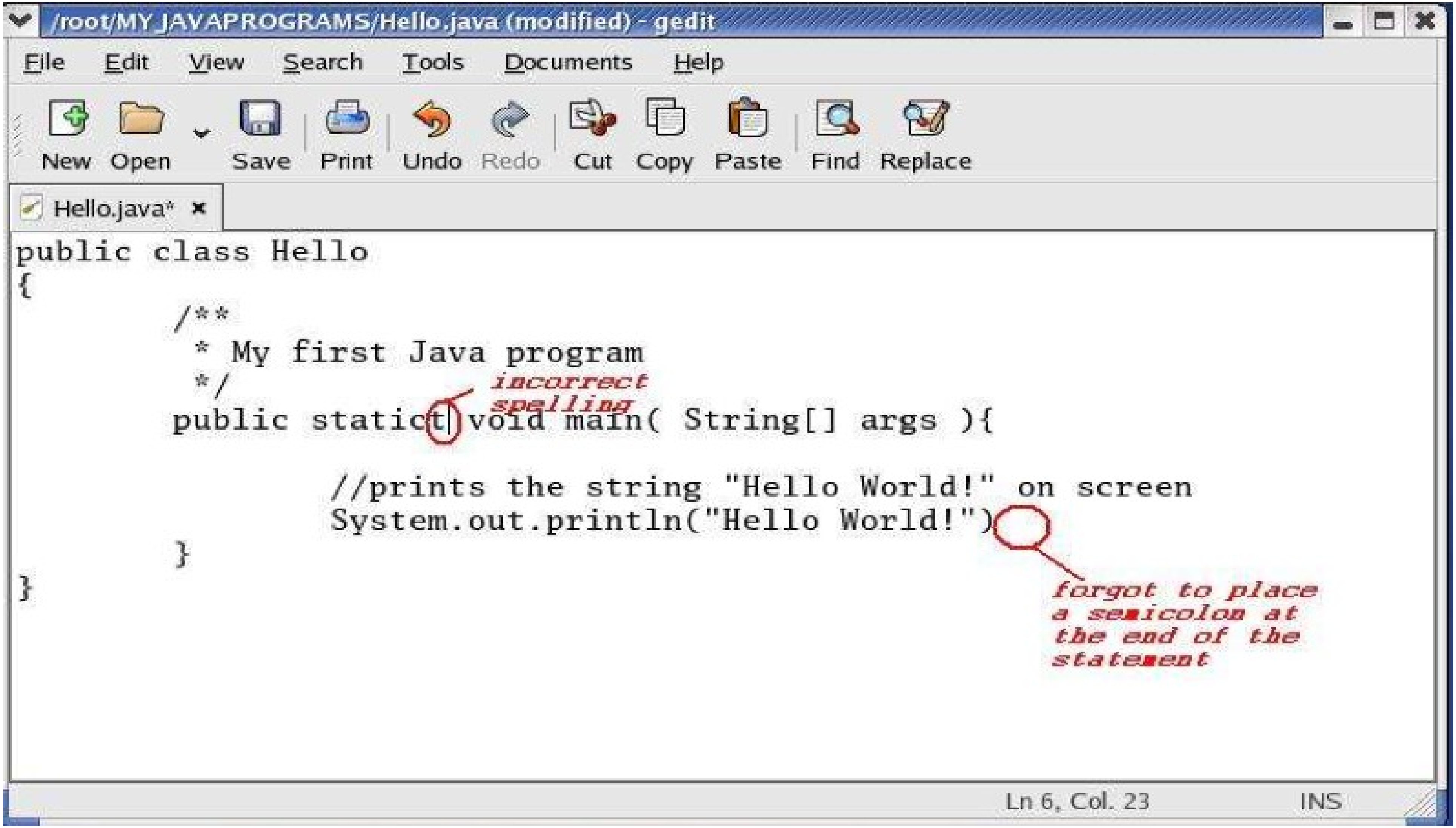
Errors: Syntax Errors & Runtime Errors

Errors : Syntax (Compile) Errors

- Syntax Errors
  - Errors are usually from typing errors
  - Detected at compile time, hence the reason they are called compile errors

- Common Syntax Errors:
  - Misspelled a keyword in Java
  - Forgot to write a semi-colon at the end of a statement

Example: Syntax Error

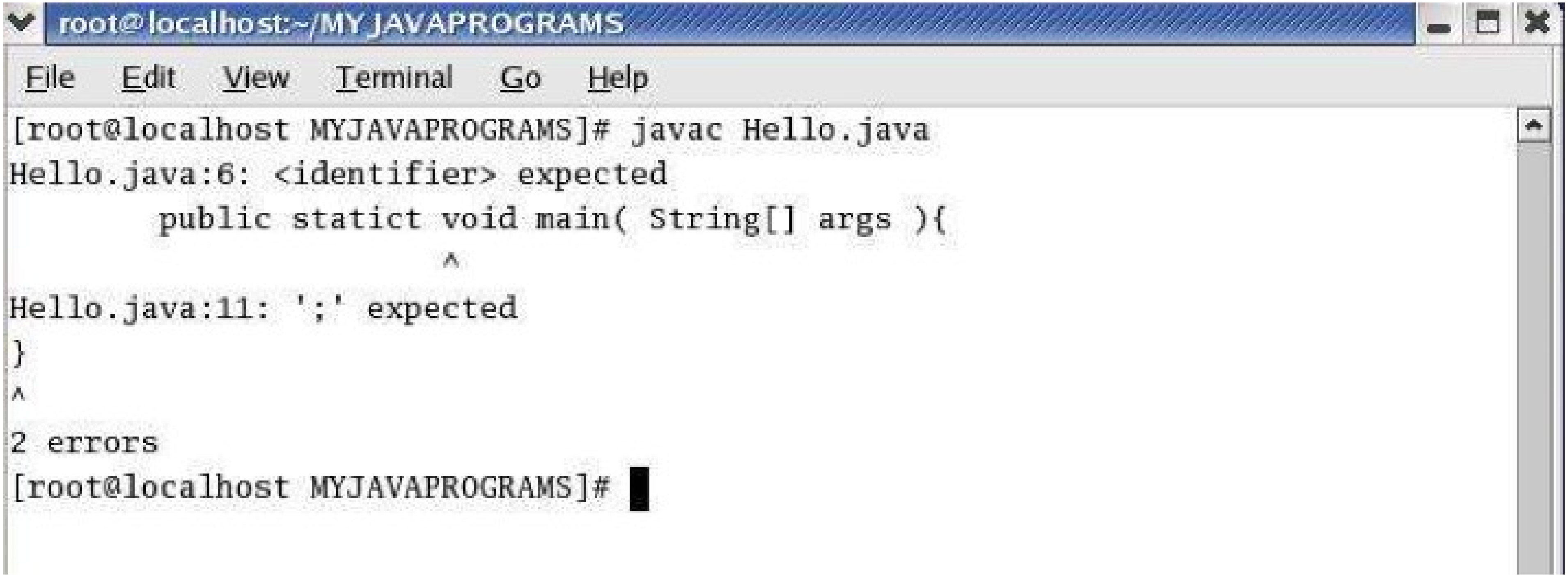


```
public class Hello
{
        /**
         * My first Java program
         */
        public statict void main( String[] args ){

                //prints the string "Hello World!" on screen
                System.out.println("Hello World!")
        }
}
```

incorrect spelling

forgot to place a semicolon at the end of the statement

Syntax Error detected by Compiler


root@localhost:~/MYJAVAPROGRAMS
File Edit View Terminal Go Help

```
[root@localhost MYJAVAPROGRAMS]# javac Hello.java
Hello.java:6: <identifier> expected
        public static void main( String[] args ){
                               ^
Hello.java:11: ';' expected
}
^
2 errors
[root@localhost MYJAVAPROGRAMS]#
```

Errors: Runtime Errors

- Run-time Errors
  - Errors that will not display until you run your program
  - Examples:
    - You want your program to print 100 strings of “Hello world”, but it only printed 99.
    - Your program expects a number input from the user, but the user inputted a string, and so your program causes an exception

Lab:

Exercise 2: Build & Run Java Program using IDE 1001_javase_progenv.zip

Code with Passion!
JPassion.com