Lesson 04: Our First Java Program (W01D4)

Balboa High School

Michael Ferraro

<mferraro@balstaff.org>

Do Now

Start a terminal shell. From there, issue these commands and then come up with an explanation for what's happening. You may be asked to share with the class.

```
env

echo "Hello"

echo "Hello, $USER"

APCS="cool"

echo "$APCS"

APCS="I am $APCS"

echo "$APCS"
```

Aim

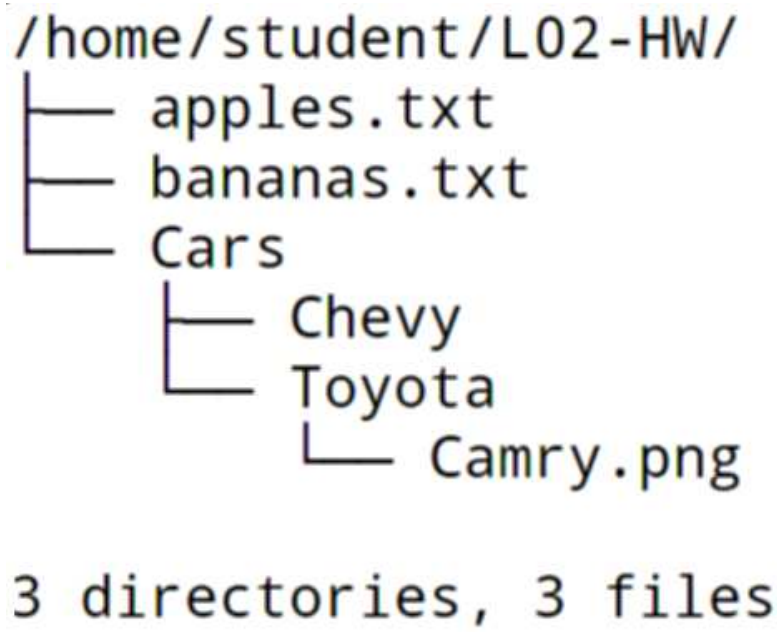
Students will learn about environment variables, go over the home Java development setup, and work on their first Java program, `HelloWorld`.

Recap of Environment Variables

- In general, *variables* are containers for information that needs to be remembered, later recalled, possibly changed, recalled again, etc.
- *Environment vars* maintain information about the running state of the operating system. (See the output of the `env` command.)
- What happens when you type the name of a program (or command) that isn't in the current directory?

A Clever Solution to `L02` HW

Let's see a clever solution to the last HW[a]...



[a]Note for teacher: `watch`, `mv`

Our First Program

Here's an outline of the process for writing a simple program:

- Use a text editor (e.g., `gedit` or Notepad) to write *source code*.

Our First Program

Here's an outline of the process for writing a simple program:

- Use a text editor (e.g., `gedit` or Notepad) to write *source code*.
- From the command prompt, compile the program into Java bytecode using `javac`.

Our First Program

Here's an outline of the process for writing a simple program:

- Use a text editor (e.g., `gedit` or Notepad) to write *source code*.
- From the command prompt, compile the program into Java bytecode using `javac`.
- Run the Java class file using `java`.

Our First Program

Here's an outline of the process for writing a simple program:

- Use a text **edit**or (e.g., `gedit` or Notepad) to write *source code*.
- From the command prompt, **compile** the program into Java bytecode using `javac`.
- **Run** the Java class file using `java`.

Step 1: Source Code

1. Start a terminal shell.
2. Create a directory to contain our work:
   `/home/student/Desktop/FirstProgram`
3. `cd` to the `FirstProgram` directory.
4. `gedit HelloWorld.java &`

Step 1: Source Code

1. Start a terminal shell.
2. Create a directory to contain our work:
   `/home/student/Desktop/FirstProgram`
3. `cd` to the `FirstProgram` directory.
4. `gedit HelloWorld.java &`

Notes about Java source file names:

- All Java sources end in `.java`
- The first letter of each word in the filename is capitalized — `CamelCase`.

Step 1: Source Code

```
public class HelloWorld {

}
```

All Java programs need a `class` statement that contains all the rest of the code. The class name **must** match the filename (minus the `.java` extension).

Step 1: Source Code

```
public class HelloWorld {




}
```

Make room for what's next!

Step 1: Source Code

```
public class HelloWorld {

    PRESS [TAB] TO INDENT...



}
```

Step 1: Source Code

```
public class HelloWorld {

    public static void main(String[] args) {
    }


}
```

`main()` is a Java method. When the `HelloWorld` program is run, whatever code appears within this method will be the first to run.

Step 1: Source Code

```
public class HelloWorld {

    public static void main(String[] args) {

    }

}
```

Make room. . .

Step 1: Source Code

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println();
    }

}
```

`System.out.println()` is a Java statement. Think of it as a command that we might run at the command prompt.

Step 1: Source Code

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println();
    }

}
```

`System.out.println()` is a Java statement. Think of it as a command that we might run at the command prompt.

Just as `ls` can take *arguments* — as in `ls /etc/*.conf` — so too can `System.out.println()` in Java. Instead of taking filenames, it takes a `String`, or text, to be printed to the screen.

Step 1: Source Code

```
public class HelloWorld {

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

}
```

In Java, `String`s must be wrapped in double quotes (“ ”).

Step 1: Source Code

```
public class HelloWorld {

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

}
```

Save the file!

Step 2: Compile the Source

Back to the terminal shell:

- `ls -l` ← make sure you see `HelloWorld.java`

Step 2: Compile the Source

Back to the terminal shell:

- `ls -l` ← make sure you see `HelloWorld.java`
- Compile your Java source code into a new, Java bytecode file:

```
javac HelloWorld.java
```

Step 2: Compile the Source

Back to the terminal shell:

- `ls -l` ← make sure you see `HelloWorld.java`
- Compile your Java source code into a new, Java bytecode file:

  ```
  javac HelloWorld.java
  ```

- See what's changed: `ls -l`

Step 2: Compile the Source

Back to the terminal shell:

- `ls -l` ← make sure you see `HelloWorld.java`
- Compile your Java source code into a new, Java bytecode file:

  ```
  javac HelloWorld.java
  ```

- See what's changed: `ls -l`
- `HelloWorld.class` is a Java class file that can be *run*!

Step 3: Run the Class

```
java HelloWorld
```

→ `java` runs the Java Virtual Machine (a.k.a. JVM). It knows how to find a Java class in the current directory (or in a `classpath`) and run Java bytecode, or a set of instructions that it understands how to interpret.

`HelloWorld` in Windows

Many of you will repeat this process in Windows at home. Let's practice here.

- Reboot into Windows
- Using Notepad, we'll compose `HelloWorld.java`
- Use a command prompt to compile and run `HelloWorld`

HW

Your HW is to get your home computer ready to edit, compile and run Java programs. Once that's done, **repeat today's activity:** `HelloWorld`.

- Mac OS/X: You likely already have the JDK installed (i.e., `java` and `javac` already work from your terminal). In Spotlight, type `terminal` to find your Terminal application. Then follow the instructions in the earlier slides, using `TextEdit` as your text editor.[a]

- Windows:
  - Follow the instructions in this PDF file for getting the Oracle JDK installed and configured on your computer.
  - Go on to the next slides...

[a] Make sure you save `HelloWorld.java` as *plain text!*

`HelloWorld` on Windows

1. Start a Command Prompt (Windows key + R → `cmd`)
2. `cd Desktop`
3. `mkdir FirstProgram`
4. `cd FirstProgram`
5. `notepad HelloWorld.java`
   - retype the source code on the final "Step 1: Source Code" slide
   - save the file (ctrl + s)
6. `javac HelloWorld.java`
7. `java HelloWorld`