CS1 — Practical Session 1: Java Installation & Compiling and Running Simple Java Programs

Kurt Driessens & Kamil Bujnarowski

September 5th, 2018

1 Introduction

In this first practical session you will learn how to setup a Java Virtual Machine and the Java Compiler and how to compile your first program(s). You will mainly interact with the command-line interface (Terminal) of your OS and a text-editor to create files. This guide uses `fixed-width font` for command-line in- and output. The symbol `$` indicates user input and you have to press return (↵) after typing the command:

```
$ command
Example output.
```

2 Installing the Java Software Development Kit (JDK)

Installation may differ depending on your operating system. See below which one applies to you.

2.1 Windows 7/8.1/10/...

If you are working on a university computer, Java may already be installed. Proceed to the "Test the installation" section below to check if this is the case - if not, return here for installation instructions.

Download and install JDK You can download the installation package from

`http://www.oracle.com/technetwork/java/javase/downloads/index.html`

Click the Java Platform (JDK) 10 Platform icon. On the next screen, select the correct operating system and hardware platform. Don't forget to accept the Licence Agreement. Wait for the installation file to download and run it. Follow the instructions on screen to install JDK 10.

Update environment variables Now, you need to update the environment variables, so Windows understands the commands to compile and run Java programs. First, open the 'System' screen in the Control Panel. Click 'Advanced system settings' in the menu on the left and then click the 'Environment Variables' button.

Select the 'PATH' variable in the 'User variables' list (or in the 'System variables' list if you have administrator rights) and click 'Edit'. Add the text `;<INSTALLATION_PATH>\bin` to the variable value, where `<INSTALLATION_PATH>` is the folder where you installed the JDK, for example `C:\Program Files\Java\jdk1.8.0_60`.

If the 'PATH' variable doesn't exist yet, create a new variable with the name 'PATH' and the value `<INSTALLATION_PATH>\bin`.

Restart your computer for the changes to become effective.

Test the installation Open a command prompt window (Start → `cmd` in Start Search box). Enter the following command

```
$ java -version
```

You should get an output similar to this (the version numbers may differ):

```
java version "1.8.0_60"
Java(TM) SE Runtime Environment (build 1.8.0_60-b27)
Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)
```

Now try and test the java compiler command. The compiler should respond with the version number (again, the version number may differ):

```
$ javac -version
javac 1.8.0_60
```

If any of the given commands don't give the desired output, repeat step 2 to check if the environment variables are set correctly.

Create a project folder Create a folder called `CS1` in your (My) Documents folder. If you work on a university computer, you should create this folder on the `I:` drive. In the command prompt (or terminal) window, browse to this folder using `cd <FOLDER_PATH>` (for example `cd C:\Users\MyName\Documents\CS1`) to change folders and, for example, `i:` to change drives.

Then continue with Section 3

2.2 Ubuntu

Open a terminal (Applications → Accessories → Terminal).

Check which version of the Java virtual machine is installed on your system:

```
$ java -version
```

You should get an output similar to this (the version numbers may differ):

```
java version "1.7.0_65"
OpenJDK Runtime Environment (IcedTea 2.5.1) (7u65-2.5.1-4ubuntu1~0.14.04.2)
OpenJDK 64-Bit Server VM (build 24.65-b04, mixed mode)
```

Next, check if the Java compiler is installed:

```
$ javac -version
```

If the compiler is not installed you will get a message like:

```
The program 'javac' can be found in the following packages:
 * openjdk-7-jdk
 * ...
```

Proceed with installing the compiler:

```
$ sudo apt-get install openjdk-7-jdk
```

Retry the previous command and this time the compiler should respond with a version number:

```
javac 1.7.0_65
```

Create a new directory `CS1` in your home folder (`/home/<USERNAME>/CS1`):

```
$ cd
$ mkdir CS1
```

and enter this directory by executing

```
$ cd CS1
```

Then continue with Section 3

2.3 Mac OS X

Open a terminal (`/Applications/Utilities/Terminal.app`).

Check which version of the Java virtual machine is installed on your system:

```
$ java -version
```

You should get an output similar to this (the version numbers may differ):

```
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)
```

Remark: If you run (Snow-)Leopard or Lion the version and build numbers might deviate.

Next, check which version of the Java compiler is installed:

```
$ javac -version
javac 1.7.0_45
```

Create a new directory `CS1` in your home folder (`/users/<USERNAME>/CS1`):

```
$ cd
$ mkdir CS1
```

and enter this directory by executing

```
$ cd CS1
```

Then continue with Section 3

3 Writing, compiling and running your first Java program

Leave the terminal window open and launch a text editor. For easy reference between lectures and practical sessions, we recommend you use jEdit. This editor was written in Java and even comes with a Java based installation program. Since you've made sure Java is correctly installed on your machine, you should be fine using that installer. It can be downloaded from `http://www.jedit.org/index.php?page=download` or from the CS1 practical sessions page on EleUM.

When the installation is complete, startup jEdit and type (do not copy-paste!) your first program:

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

Save the file as `HelloWorld.java` in the CS1 folder.

In the terminal window, make sure you are in the CS1 folder, and type:

```
$ javac HelloWorld.java
```

Your program is now compiled and a new file `HelloWorld.class` has been created. You can verify this using `dir` on Windows and `ls` on Linux or Mac OS based machines. For example,

```
$ ls
HelloWorld.class  HelloWorld.java
```

Now, run your first program:

```
$ java HelloWorld
Hello World!
```

Exercise 1

If you got this far, congratulations, now adapt the HelloWorld program to perform some more advanced ASCII art. Save each of the programs into a differently named file, e.g. Lego.java, Fish.java, etc. and make sure you remember to adapt the name in the `public class <name>` statement to match the filename to avoid compiler errors.

Get your programs to display:

```
a)         ___________
          / (_)  (_) /|
         / (_)  (_) / /
        /__________/ /
        |__________|/
```

```
b)            ,--,_
       __    _\.---'-.
       \ '.-"     // o\
       /_.'-._    \\  /
              `"--(/"`
c) Your name using the following alphabet:
   ___   ___   ___  ___  ___  ___   ___  _ _  _     _
  / - \ | - > /  _>|   \| __>| __> /  _>| | || | _ | |
  | | | | _ \|  (_ | | || __>| __>|  < \|   || |< \| |
  |_|_| |___/ \___>|___/|___>|_|   \___/|_|_||_| \___/

   _  _  _    _____  _  _  ___  ___  ___   ___   ___
  | |/ /| |  /     \| \| |/   \| -_>/   \ | - > / __>
  |   < | |_ | | | ||  \ || | || |  | | | |   \ \__ \
  |_|\_\|___||_|_|_||_\__|\___/|_|  \__\_>|_|\_\<___/

        _____  _ _  _ _  _  _  _  _ __  __ ____
       |_   _|| | || | || || |< \/ >\ \/ /<_  /
         | |  | | || | ||    | >  <  \  /  / /_
         |_|  \___/ \_/ |_/\_|<_/\_> /_/  /____>
```

Exercise 2

Now do the impossible: Make your Java program output the product of two positive numbers, but make it so that the value printed by Java is a negative number.

Assignment

For this week's assignment, go find some more challenging ASCII art on the Internet, and make your program output that.

Then, **upload** the .java file EleUM under today's assignment to make sure you get credit for your work.