The University of Melbourne
School of Computing and Information Systems

COMP90041
Programming and Software Development
Second Semester, 2018

Lab 1 — Programming Practice (Week 2)

Getting Started

The object of this week is for you to gain some familiarity with the mechanics of creating, editing, compiling and running Java programs using the Sun Microsystems' Java compiler and virtual machine that will be the official Java platform for all the labs and projects throughout the semester.

It would be a good idea to work on these exercises in a pair with another student, so you can swap ideas. You are welcome to work in pairs or teams on all the *unassessed* workshop exercises, like those this week, but of course you must work alone on anything that will be submitted for assessment.

Which machines do I use for Java?

While you are free to choose your favorite Hardware/Software platform for developing your Java programs, they are expected to compile and run on either of the following machines:

`dimefox.eng.unimelb.edu.au` `nutmeg.eng.unimelb.edu.au`

To access these machines from a Windows computer, such as the ones in the labs, we recommend using either `MobaXterm` (`http://mobaxterm.mobatek.net/`) or the `PuTTY` program (`http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html`, download `putty-0.62-installer.exe` and execute it to install) Both of these are already installed on the lab computers. `MobaXterm` is more powerful, but a little more complex.

From a Mac OS X computer, use the Terminal program from the Applications/Utilities list. At the command line, type `ssh` followed by a space, your CIS department login ID, an at-sign (), and the selected host name. Enter your password when prompted.

Setting up your environment

The first time you log in, you should do a few things. First, create a directory (folder) for your work by running the command:

```
mkdir ~/COMP90041
```

and protect it so no one else can read it:

```
chmod go-rwx ~/COMP90041
```

Then change to that directory by the command

```
cd ~/COMP90041
```

On the departmental student machines, various Java compilers and Java Runtime Environments (JRE) are installed. Thus, you need to make sure that you are using the correct versions of these programs. To do this, type the following:

```
java -version
```

It should report back at least Java 1.6.0

You will also want to copy files from your own computer to the servers and vice-versa. If you use `MobaXterm` as your terminal program, it also provides this functionality, so you won't need another program. If you use `PuTTY`, though, you'll need a separate program Two popular programs for this are FileZilla (`http://filezilla-project.org/`, Windows, Mac, Linux) and WinSCP (`http://winscp.net/download/`, Windows only). Both present you files and directories on your local machine on the left, and on the servers on the right, and allow you to drag files between them. This will allow you to develop files on your own computer, and upload them to the server for final testing and submission.

Using javac and java

Now you can try to compile and run a simple Java program. Download the Java source file `FirstProgram.java` from the subject's LMS page, and then copy it to the server. Try to compile it using the following command:

```
javac FirstProgram.java
```

If all is well, `javac` will execute and finish with no messages. Then you can run the program by typing:

```
java FirstProgram
```

You should then see a hello and welcome message from the program appear on the console. If this happened then congratulations, you have just compiled and executed your first Java program!

Java IDEs

There are many Java IDEs (Integrated Programming Environments) available. We recommend Eclipse or NetBeans, but if you prefer another IDE, that's fine, too. Eclipse and NetBeans are installed in the labs; try both to see which you prefer. You can download them (for Windows, Mac, or Linux) for your own computer, and read through extensive tutorials at `http://www.eclipse.org/` or `http://netbeans.org/`. They both support numerous programming languages and have numerous plugins, but we only need Java, and do not need any plugins, for this subject.

Whichever IDE you use, start up a new project, and put into it the same simple Java example as you used above. Run the program (via the "Run" tool), and observe the output. Really, the IDEs are just invoking the Java compiler and Java interpreter behind the scenes.

Workshop Exercises

These are just for practice, and will not be assessed.

1. Write a Java program that displays your name, address and telephone number, each on a separate line.

Homework

These will also not be assessed.

2. Write a Java program that declares and initialises a `String` variable to "Hello, World!". Print out the variable on a line by itself. Then, using the `String` methods discussed in Chapter 1, print out the same string in ALL UPPERCASE on a line by itself.

   ```
   Hello, World!
   HELLO, WORLD!
   ```

3. Modify the above program by just changing the initial value of the `String` variable to "So Long!" and run it again. Now it should print out:

   ```
   So Long!
   SO LONG!
   ```

4. If you've finished everything else, go to the Useful links page and click on one of the Java Core API links. This will be a useful reference when you start writing non trivial Java programs.