Pair Programming and Java

# Introduction

Software development is a complex process and one methodology that has been developed to help manage this complexity is known as Extreme Programming (XP). This lab introduces a practice central to XP called Pair Programming. (Beck) You’ve been working with a partner in the previous two lab assignments, but for this lab we will formalize this process as we begin programming in Java.

In pair programming, two programmers share one computer. One student is the “driver,” who controls the keyboard and mouse. The other is the “navigator,” who observes, asks questions, suggests solutions, and thinks about slightly longer-term strategies. The two programmers switch roles about every 15 - 20 minutes. Working in pairs should make you much better at programming than working alone. The resulting work of pair programming nearly always outshines that of the solitary programmer, with pairs producing better code in less time. (NCWIT)

To “learn the do’s and don’ts” of pair programming and to see pairs in action, view this entertaining video about pair programming from North Carolina State University: [*An Introduction to Pair Programming for Students*](http://www.realsearchgroup.org/pairlearning/educators.php#ppvideo) (NCSU)

The Java standard development kit (SDK) and runtime environment (JRE) are software applications that are pre-installed with the Raspberry Pi Raspian OS. The Java compiler is a program that converts a human-readable source code file into a standardized form of machine language called **bytecode**. As part of the process, the compiler checks the source code file to verify that it contains syntactically correct Java source code. If there are syntactic errors, the bytecode file is not created and error messages are displayed in the terminal.

The Java Virtual Machine (JVM) is a software interpreter that executes the bytecode on the computer. Many integrated development environments automate these tasks, but in this lab you will be performing them individually. (Lewis)

# Learning Goals and Objectives

* Know the guidelines for Pair Programming.
* Compile and execute Java programs from the command line on the Raspberry Pi

At the conclusion of this lab students are able to:

* Use Pair Programming to create Java programs to display shapes using ASCII characters

# Getting Started

Gather the materials needed:

* Raspberry Pi 2 Starter kit (including the Pi and power supply).
* Micro SD card preloaded with the operating system
* HDMI adapter cable
* Computer lab keyboard, mouse, and monitor
* USB flash drive for transferring files off the Raspberry Pi

# Tasks

1. Working with your partner, connect the keyboard, mouse, and monitor to the Raspberry Pi, then connect the power adapter and plug it into a power outlet. Compare first names with your partner and the person whose first name comes first in dictionary order will be the Driver and the other is the Navigator to start the lab.
2. The Driver has control of the keyboard to log into the home directory with: Login: pi and password: raspberry
3. Navigate to your **csc201** directory; create an empty file called **Hello.java** (refer to your list of Linux commands, if needed). This will be the Java source code file for your program, so capitalization is important –the first letter must be capitalized and all the rest should be lower case.
4. Use **Nano** open the **Hello.java** file, type nano Hello.java. (Alternatively, you can create the file and open it for editing using this single command.) (The Beginner’s Guide to Nano, the Linux Command-Line Text Editor)
5. The Driver types the following lines of code while the Navigator reviews it:

public class Hello

{

public static void main(String[] args)

{

System.out.println("Hello, World");

}

}

1. **WriteOut** to save and then **Exit** Nano. Use the Linux list command (**ls**) to view the contents of the **csc201** directory and note the single **Hello** file with a .**java** extension.
2. To compile the source code file, use the command **javac** followed by a space and then the name of your source code file including the extension (javac Hello.java). If there are no errors, view the contents of the **csc201** directory to see the additional file named **Hello.class**. If error messages are displayed, reopen the **Hello.java** file and work with your partner to discover how your file differs from the text shown above. Correct any errors and recompile.
3. To run the program with the command **javac** followed by a space and the name of your program without the extension (java Hello). The program should display the following output in the terminal: Hello, World
4. Time to switch places with your partner again for the next program.
5. Repeat the process outlined for the Hello program to write a program that displays both of your names. Substitute your first and last names for <Driver> and <Navigator> in the following source code file:

public class Names

{

public static void main(String[] args)

{

System.out.println("This program was written by <Driver> and <Navigator>");

}

}

1. To remove your files from the Raspberry Pi, insert a USB flash drive into one of the open USB drives. To access your flash drive, navigate to the media directory on the Pi:

**pi@raspberrypi** **/ $** cd media/

**pi@raspberrypi** **/media $** ls

**NO\_NAME**

In this example, the name of the flash drive is “NO\_NAME”. Change to this directory and create a subdirectory for your source code files.

**pi@raspberrypi** **/media $** cd NO\_NAME/

**pi@raspberrypi** **/media/NO\_NAME $** mkdir csc201

To be able to write to this directory, change the permissions with the **chmod** command using **a+wx**,

**pi@raspberrypi** **/media $** chmod a+wx csc201

**pi@raspberrypi** **/media/NO\_NAME $** cd csc201/

Now move the Hello.java file and the Names.java to the flash drive.

**pi@raspberrypi** **/media/NO\_NAME $** mv ~/csc201/Hello.java /media/NO\_NAME/csc201/Hello.java

**pi@raspberrypi** **/media/NO\_NAME $** mv ~/csc201/Names.java /media/NO\_NASME/csc201/Names.java

Navigate to the csc201 directory on the Pi and remove the bytecode files.

**pi@raspberrypi** **/media/NO\_NAME/csc201 $** cd ~/csc201/

**pi@raspberrypi** **~/csc201 $** ls

Hello.class Names.class

**pi@raspberrypi** **~/csc201 $** rm Hello.class

**pi@raspberrypi** **~/csc201 $** rm Names.class

To safely remove the USB flash drive, perform these steps:

**pi@raspberrypi** **~/csc201 $** df

Filesystem 1K-blocks Used Available Use% Mounted on

rootfs 3683656 2532652 951092 73% /

/dev/root 3683656 2532652 951092 73% /

devtmpfs 218636 0 218636 0% /dev

tmpfs 44584 248 44336 1% /run

tmpfs 5120 0 5120 0% /run/lock

tmpfs 89160 332 88828 1% /run/shm

/dev/mmcblk0p1 57288 19480 37808 35% /boot

/dev/sda1 3904640 186336 3718304 5% /media/NO\_NAME

**pi@raspberrypi** **~/csc201 $** sudo apt-get install eject

**pi@raspberrypi** **~/csc201 $** udisks --unmount /dev/sda1

The yellow highlighted line shows the drive information to use for udisks command (it might vary somewhat on your Pi).

# Deliverables

Properly shut down the Raspberry Pi, sudo shutdown –h now. Disconnect all peripheral devices and reconnect them to the computer lab machine. Verify that the mouse, keyboard, and monitor all work properly. Return the Raspberry Pi to your instructor in class before you leave.

Complete the assessment for this lab in Blackboard.

# References

Beck, Kent. Extreme Programming Explained. Upper Saddle River: Pearson Education, Inc., 2004.

FrontPage - Raspian. 18 June 2015 <raspian.org>.

Lewis, Nell Dale and John. Computer Science Illuminated, Sixth Edition. Sixth Edition. Burlington: Jones & Bartlett Learning, 2016.

NCSU. Pair Learning: The Use of Pair Programming in Education. 29 June 2015 <http://www.realsearchgroup.org/pairlearning/>.

NCWIT. Pair Programming-in-a-box. 29 June 2015 <https://www.ncwit.org/resources/pair-programming-box-power-collaborative-learning>.

RASPBERRY PI FOUNDATION. Raspberry Pi Learning Resources. 17 June 2015 <https://www.raspberrypi.org/learning/getting-started-with-raspberry-pi-lesson/lessons/>.

RPi Hardware. 30 March 2015. Bill Traynor. 17 June 2015 <http://elinux.org/RPi\_Hardware>.

The Beginner’s Guide to Nano, the Linux Command-Line Text Editor. 30 June 2015 <http://www.howtogeek.com/howto/42980/the-beginners-guide-to-nano-the-linux-command-line-text-editor/>.

Wallace, Matt Richardson & Shawn. Getting Started with Raspberry Pi. Sebastopol: Maker Media, 2013.