ECSE211 Getting Started Guide

Fall 2019

Last updated September 10, 2019

Note: This PDF document is meant to be printed. Consult the most up-to-date web version here:

`mcgill-ecse211-f19.github.io/getting_started_guide/GettingStarted-F19-updated`

1 Background

The primary objective of this course is to learn about the **design process**. In fulfillment of that objective, lectures will teach the basic theory while labs will allow you to get the necessary practical experience, which will come in the form of designing an autonomous robot using the LEGO Mindstorms EV3 kit embedded system.

The heart of Mindstorms is the LEGO "Brick," which serves as a general purpose embedded controller with sensor inputs and motor outputs, operated by a simple user interface (UI) consisting of an LCD display and a set of push buttons. Think of the microcontroller in the "Brick" as a very basic computer with a small processor and a bit of memory, just enough to power a small robot.

A microcontroller becomes an embedded system when you program it with software; as such, a large part of building an embedded system involves designing, writing and testing software. Every embedded system will have a development environment that gives you the basic tools you need to actually write the software. In ECSE202, you wrote simple Java programs using the Eclipse Integrated Development Environment (IDE). For this course, we have kept things the same. As you've no doubt noticed, Java runs on a lot of different devices, from Windows to macOS and your Android phone. This is because all Java code runs on a JVM (Java Virtual Machine), which translates Java code into a form that the specific device actually understands. This ability is often referred to as "write once, run everywhere." The JRE (Java Runtime Environment) includes the JVM and code specific to a device.

In this course, you will run Java on the EV3 robot as leJOS provides the JRE for the EV3. As part of the JRE, leJOS provides various classes that allow you to use the robot's motors, sensors and other components. These classes take care of many of the complex details that usually come up in robotics, allowing you to focus on building a robot and not, for instance, getting a motor to turn in the right direction. A plugin for Eclipse lets you write code on your computer and upload it to the leJOS environment easily.

Unfortunately, the EV3 runs the Mindstorms operating system, which is intended for people without programming experience (e.g. late elementary to high school age) and doesn't support Java, so we need to reconfigure it by installing the leJOS software. As the Brick runs the Mindstorms environment on top of Linux, all we need to do to install leJOS is to put it on an SD card. We will discuss how to do this later.

2 The kit

The kit we provide contains the "Brick" mentioned earlier, motors, sensors and various LEGO parts.

2.1 The battery

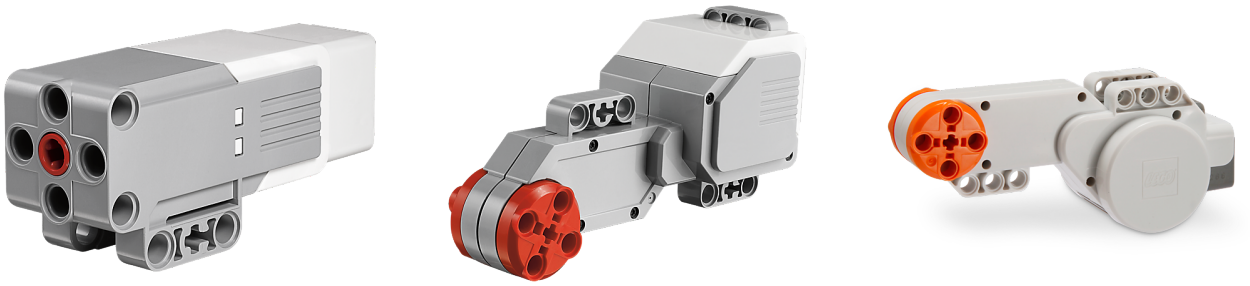
You will receive both AA batteries and a rechargeable battery pack, shown in Fig. 1b. The brick can be powered by either, but we recommend using the rechargeable battery in most cases. The battery can be charged with the included wall adapter, either by itself or while inserted into the brick. When both green and red LEDs are on, the battery is charging. When only the green LED is on, the battery is fully charged. Note that if the battery is in the brick while charging it may not reach 100% until the EV3 is powered off. **We recommend keeping the battery plugged in as often as possible.** In the leJOS software, you will see the battery voltage displayed in the top left corner as shown in Fig. 1a. The highest it can go is 8.0V; we recommend you start charging it once it gets below 7.0V. Try to avoid letting the battery die completely as this can damage it and reduce its charge capacity.

(a) (b)

Figure 1: (a) The main menu with a fully charged battery and (b) the battery itself.

2.2 The motors

There are three types of motors in your kit: medium (Fig. 2a), large EV3 (Fig. 2b) and large NXT (Fig. 2c). The large NXT motors are older than the EV3 ones, but besides a difference in attachment points they are identical. The main difference between the large and medium sizes is power and precision: the large ones are powerful, but not the most accurate. They are well suited for jobs like driving the entire robot around. The medium one by contrast is weaker but much more precise; you would use it for smaller jobs like moving a sensor back and forth. All motors connect to the ports A - D on the top of the brick.



(a) The medium motor. (b) The large EV3 motor. (c) The large NXT motor.

Figure 2: Three types of motors.

2.3 The sensors

While you receive several sensors, there are two that are used the most in the course: the ultrasonic and color sensors. The ultrasonic sensor (Fig. 3a) is used to measure distance by sending ultrasonic waves and measuring how long it takes the wave to echo back. Its main use is in detecting obstacles the robot needs to avoid. Unfortunately, ultrasonic waves are not the best way to measure distance and the sensor can easily fail to detect objects or detect objects that aren't there, which you will need to try and compensate for in software. The color sensor (Fig. 3b), often referred to as a light sensor, can determine the color of an object as well as the intensity of light reflected by said object. This is useful for detecting the black lines on the wooden panels your robots will drive on. In order to work properly, the sensor needs to be placed at a very specific distance from the object, which you will to determine. Other than that, it is much more reliable than the ultrasonic sensor. All sensors connect to the ports labelled 1 - 4 on the bottom of the brick.

(a) The ultrasonic sensor. (b) The light sensor.

Figure 3: The two most frequently used sensors.

2.4 The Brick

There's a microcontroller in the brick, basic computer with a small processor and a bit of memory. A microcontroller becomes an embedded system when you program it with software

The "Brick" you receive, shown in Fig. 4, is an embedded system as described in the background section. The important features are as follows:

- **LCD**: Used to display information

Figure 4: The LEGO "brick".

- **Speaker**: Can make beeps, useful in some labs to let you know what your code is doing. Also capable of playing the *Imperial March* from Star Wars.
- **SD Card slot**: For the SD card on which leJOS is installed.
- **USB type-A**: Used for the USB Wi-Fi adapter.
- **USB mini-B**: Used for connecting to a computer.
- **Ports A-D**: Compatible with the RJ-11 like wires you are provided for connecting motors.
- **Ports 1-4**: Same as the motor ports, except used for the various sensors

The brick can be turned on by pressing the center button for a second or two. In leJOS, it can be turned off by pressing the back (a.k.a. escape) button (top left) and confirming the shutdown command.

2.5 Defective components

On rare occasions, one of your parts may be defective. If you believe this to be the case, show the problem to a TA in the lab to verify before exchanging the part for a new one at the counter where you originally picked up your kit.

3 Preparing your computer

In order to work with the EV3, you need to set up your development environment as mentioned in Section 1.

1. Download the file `leJOS_0.9.1_Package.zip` from MyCourses and extract the contents. The directory structure should look like this:
   - Computer
   - Documentation
   - SD_Card
   - `ecse211-eclipse-preferences.epf`
2. Copy the `leJOS_EV3_0.9.1-beta` folder found within the Computer folder to a location on your machine. Keep track of this location.
3. Install the 32-bit (x86) Java 7 SDK on your machine. It can be downloaded from here: `www.oracle.com/java/technologies/javase-java-archive-javase7-downloads.html`. LeJOS does not support more recent versions.
4. Install and setup Eclipse:
   (a) Follow these instructions here: `www.eclipse.org/downloads/packages/installer`. The installer will ask you what version to install. "Eclipse IDE for Java Developers" is sufficient for this course. Note that you will need to install Java 8 or above (if you haven't already) to run Eclipse.
   (b) When starting Eclipse for the first time, select (or create) a workspace folder that is dedicated for this course, so your other projects can have their own settings.
   (c) Configure Eclipse settings according to our requirements. Open Eclipse Preferences and select Import (the button in the lower-left corner of the window with this arrow icon ↘), then choose `ecse211-eclipse-preferences.epf`, which can be found in the zip file you downloaded from MyCourses. Select all the boxes.
5. Install the Eclipse plugin. Open Eclipse, and navigate to `Help → Install New Software...`. Paste this link in the "Work with..." textbox: `http://lejos.sourceforge.net/tools/eclipse/plugin/ev3`. Then select "Add..." and enter `LeJOS EV3 Eclipse plugin` as a repository name. Check the "leJOS EV3 Support" box and follow the prompts.

   Select `Install anyway` when the warning about unsigned content appears. After installation, you need to tell the plugin where you installed leJOS as well as how to connect to your brick. The plugin settings page is called `leJOS EV3` and is part of the Eclipse "Preferences" window. Fig. 5 shows an example of how to set it up on a Linux system. Set it up as follows:
   - Make sure `EV3_HOME` is set properly to wherever you installed leJOS. This is the folder inside `Computer`.

- The box "Run Tools in separate JVM" should be checked
- Make sure "Use ssh and scp" is **not** checked
- You will set the IP address after setting up LeJOS on the EV3, so leave it blank for now.

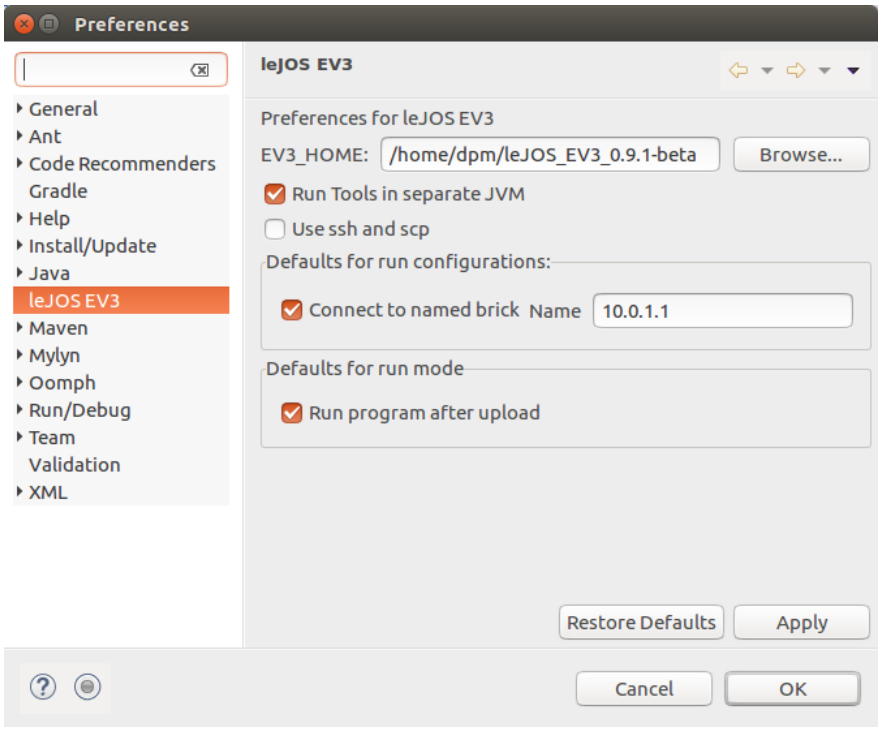


Figure 5: The EV3 settings.

4 Installing leJOS to the SD card

The next step is to install leJOS onto the SD card[1].

If you have problems, post in the discussion board or ask a classmate for assistance.

1. Insert the SD card into a card reader connected to (or part of) your computer.
2. Format the entire SD card as follows:
   - **Partition table:** MS-DOS
   - **Partition layout:** a single FAT32 partition occupying the entire SD card, i.e. 16GB or 14 GiB

[1]The SD card you receive may already have leJOS on it, but it is often in some broken state after being used for an entire semester so it needs to be replaced.

Your OS should have a tool that can do this. Alternatively, there are many third party tools that can do it for you. Feel free to look at the many good guides for SD card formatting online if you have trouble. Be careful to select the correct device, as otherwise you could end up *wiping out your operating system*.

3. Copy all files from the folder `SD_CARD` to the SD card.

4. Safely eject the card using the appropriate method for your operating system.

5. Insert the card into the SD card slot on the brick, making sure the brick is powered off.

6. Turn the brick on (center button) and wait for leJOS to go through the installation procedure, which will take about 10 minutes.

7. When it's done, you should see the screen as shown in Fig. 6.

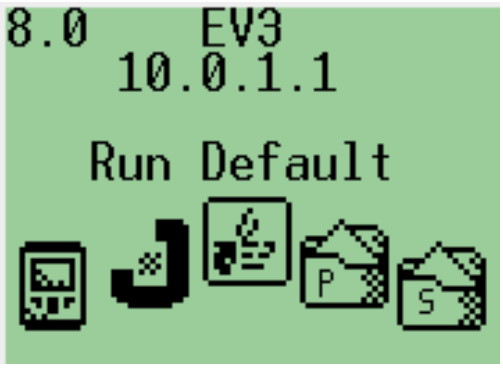


Figure 6: The EV3 screen after leJOS installs.

5 Running code on the brick

Now that leJOS is installed, the next step is to get some code running on the brick. To do this, we need to establish a connection between your computer and the brick. There are three different ways to do this: USB, Bluetooth or WiFi. As all of them use the Internet Protocol, any software that works with one will work with the others as long as the settings are correct. Since you will need Wi-Fi for debugging starting in Lab 2, we will explain how to set it up here.

1. Insert the Wi-Fi adapter then turn the brick on and connect to the DPM Wi-Fi network using the Wi-Fi menu. The password is **dddpppmmm**. Connect your computer to the same network. Note that you will lose internet access. An alternative is to use a personal hotspot.

2. In Eclipse Preferences → leJOS EV3, check and fill the "Connect to named brick" field with the IP address shown on the third line of the EV3 LCD. It should be in the `192.168.2.x` range.

3. Create a new Eclipse project as follows:

   (a) Open Eclipse and navigate to `File → New → Project...`

   (b) Select `leJOS EV3 Project`

   (c) Name it `HelloWorld` and select `JavaSE-1.7` for `Use an execution environment JRE`[2]. Click `Finish`. An example of the page is shown in Fig. 7.

   (d) Create a new class named `HelloWorld` in the package `ca.mcgill.ecse211.helloworld` i.e. following the submission guidelines for this course.

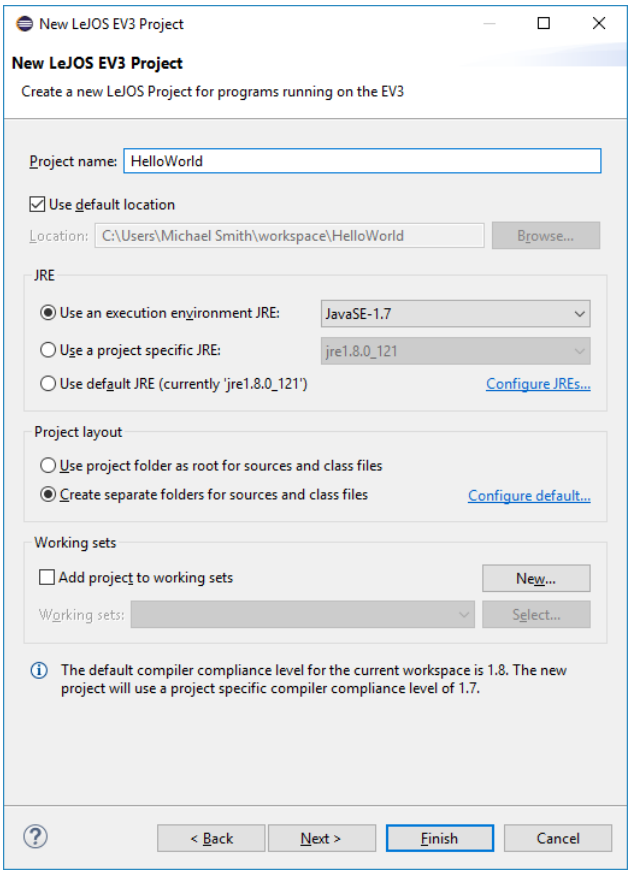


Figure 7: Creating a new EV3 project.

   (e) Add some code so that your program prints to the LCD and waits for a button to be pressed to exit[3]. Listing 1 provides an example.

[2] If you do not specify `JavaSE-1.7`, your code will crash immediately with a `ClassException` error when you run it on the brick.

[3] If you don't wait for a button press the program will exit immediately and you will end up looking at the leJOS menu rather than the text specified in your code.

```java
package ca.mcgill.ecse211.helloworld;

import lejos.hardware.Button;

/**
 * HelloWorld example: prints to screen and waits
 * for button press.
 */
public class HelloWorld {

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

}
```

Listing 1: HelloWorld code.

(f) Run the project, selecting `leJOS EV3 Program` when prompted with the `Run as` dialog. This tells Eclipse that it needs to run your code on the EV3 and not your laptop.

(g) Assuming the Eclipse plugin is set up as in step 5 of Section 3, the code should be compiled and uploaded to the brick.

(h) Verify that the LCD shows "Hello World!" and exit the program by pressing any button on the EV3.

Note that the desktop computers in the lab only support USB. To set that up, plug in the brick to the computer with the provided USB cable and check the network settings. There should be a new interface with an IP address in the `10.0.1.x` range in addition to the normal Ethernet connection. If you either don't see a new interface or it does not have an IP address in the correct range, the computer cannot see the EV3. This is likely a driver, USB port or firewall issue.

6 Resetting the EV3

Over the course of the semester, you will likely at some point need to reset the EV3 brick. The cause might be anything from an infinite loop in your code that doesn't let you quit to a bug in leJOS. Whatever the cause, we present several ways of resetting the robot if you can't get it to respond. We recommend trying them in the order presented, so the last step should only be done if all else fails.

1. **Soft Reset:** Hold the middle and down buttons simultaneously. Your program should exit and return to the main menu.

2. **Stop Program:** Using the EV3 control panel (see Section 7), connect to your robot and click on the button named Stop Program.

3. **Hard Reset:** Hold the middle and back buttons simultaneously until the robot starts rebooting.

4. **Remove battery:** The last way of resetting the brick is to simply remove and reinsert the battery.

The last two steps don't allow leJOS to shutdown nicely, and can cause the SD card to get corrupted. If you experience weird behaviour after trying one of the last two steps, you may need to reformat your SD card and re-install leJOS.

7 Useful information

While the above discusses all the essentials, there are several points that aren't essential for the first lab but can come in handy later:

1. When the EV3 is connected to your laptop, you can control it and get data from it. Open Eclipse and navigate to `leJOS EV3 → Start EV3Control`. Click `Search`, wait for it to find your robot, and click `Connect`[4]. There are many useful options available to you, ranging from testing motors and sensors to running code or killing a running program. Of particular importance is the `Console` tab, which shows what's displayed on the LCD and the output of your program. This means that you can use `System.out.println()` in your code and have the output on your laptop rather than the small LCD, useful when you need to examine a lot of data (eg, in CSV format). Fig. 8 shows what the console looks like when running the `HelloWorld` program. Feel free to explore the other tabs as well.

2. It is possible to run your code on the EV3 with the Java debugger as long as the EV3 is connected. The process is exactly the same as debugging on your computer, except for the fact that you have to choose `Debug As → LeJOS EV3 Program` instead of `Java Application`.

3. An alternative to Wi-Fi or USB is Bluetooth, although it has limited range. Once you've paired the EV3 with your laptop, you can upload code and use the EV3 control panel as you would over Wi-Fi. It may take a while after pairing for your laptop to properly connect to the EV3, though, so you will need to be patient.

4. You can change the EV3's name via the LCD (`System → Change name`). This can be useful if you don't know which robot is yours.

[4]If your robot doesn't appear, it either isn't connected to your laptop or it is being blocked by a firewall, as is the case for the desktop computers in the lab.

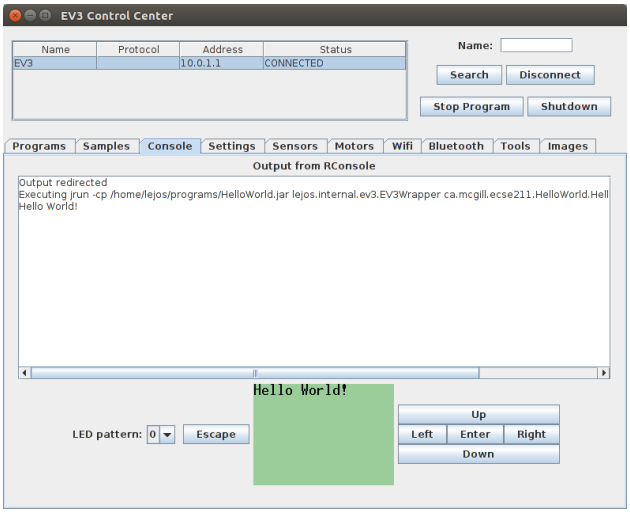


Figure 8: The EV3 control panel.

5. Documentation for all leJOS functions can be found in the `Documentation` folder in the zip file mentioned in step 1 of Section 3. Simply open the `index.html` file within using your favorite browser; the documentation follows the standard Javadoc format. Keep in mind that the documentation covers all of leJOS, including support for older LEGO products like the NXT and RCX so make sure you're looking at the correct section.