ECSE-211 Getting Started Guide

Winter 2018

Last updated January 23, 2018

Contents

1 Background

As the title of this course says, the primary objective this semester 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. That practical experience will come in the form of designing and developing an autonomous robot using an embedded system in the form of the LEGO Mindstorms EV3 kit. While this document will walk you through the set up necessary to start using the EV3 and build your robot, we first need to explain exactly what it is you have to work with.

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. Inside the brick is a Texas Instruments Sitara AM1808 microcontroller, which is based on the ARM 926EJ-S core running at 300 Mhz - a bit of a mouthful, but essentially the same technology that ran the first generation of smartphones. Devices such as the AM1808 are often referred to as a SoC, system on a chip, and contain most of the key components of a computer system on a single silicon die. You will learn more about these devices in later courses such as Microprocessors and Embedded Systems. To keep things really simple for now, you can 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 ECSE 202[1], you wrote simple Java programs, likely using an Integrated Development Environment (IDE) like Eclipse. For this course, we have kept things much the same: your *development environment* consists of Eclipse and Java. As you've no doubt noticed by now, Java will run 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[2]. 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

[1] COMP 202 if you started your first year before Fall 2016

[2] This ability is often referred to as "write once, run everywhere"

runs the Mindstorms environment on top of Linux, all we need to do is install leJOS is put it on an SD card. We will discuss how to do this later.

2 The kit

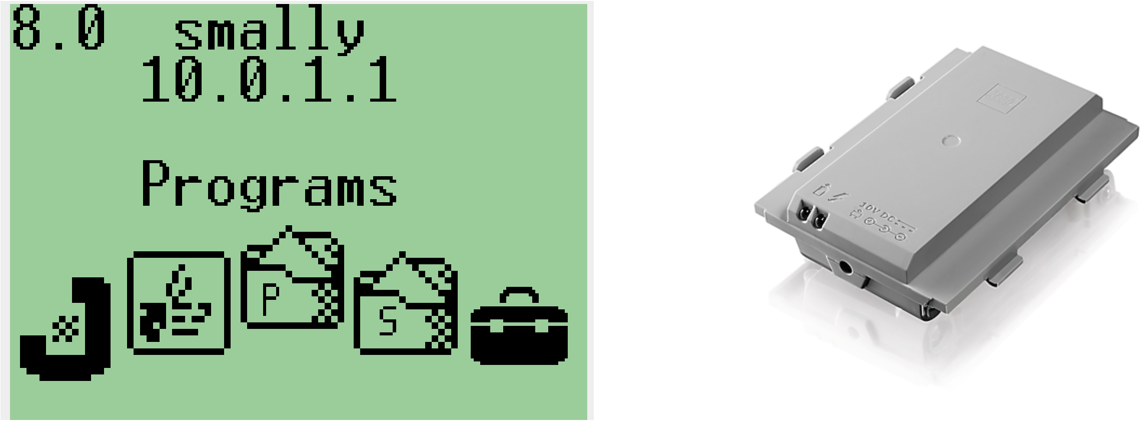
The kit we provide contains the “Brick” mentioned earlier, motors, sensors and various LEGO parts.

2.1 The battery

You will receive both AA batteries and a rechargeable battery pack, the latter of which is shown in Fig. 1b. The brick can run off either, as the battery pack simply slots into the space where you would use AA batteries. To charge the pack, use the included wall adapter, keeping in mind that the pack can be charged both in and out of the EV3 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. 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[3].

We recommend the following:

- Use the rechargeable battery often and leave the AA batteries for emergencies (e.g. the final competition)
- Keep the rechargeable battery plugged in as often as possible
- Start charging the battery once it gets below 7.0V
- Avoid letting the battery get so low that the EV3 shuts off, as this can damage it and reduce its charge capacity permanently.



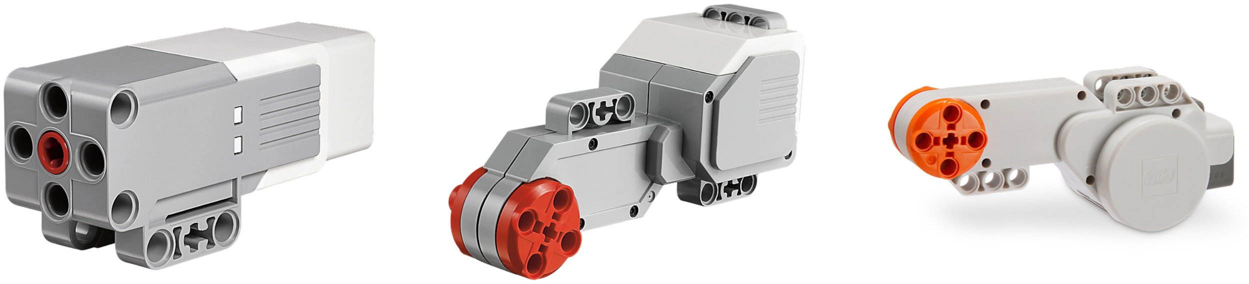
(a) (b)

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

[3] In reality, it actually goes a bit higher.

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.



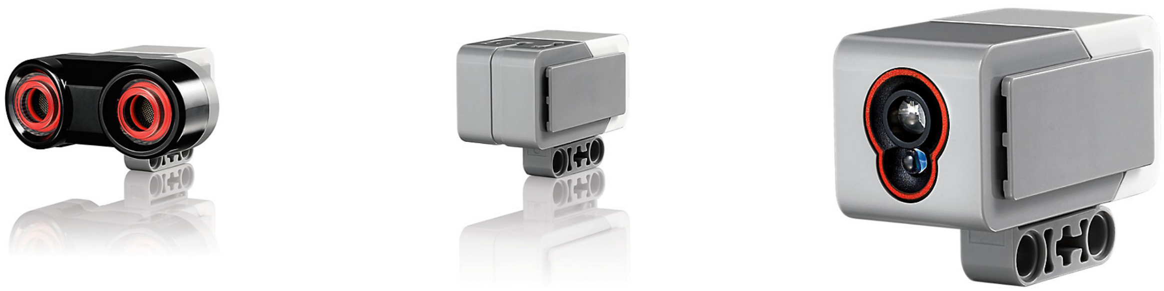
(a) Medium (b) Large EV3 (c) Large NXT

Figure 2: Three types of motors.

2.3 The sensors

While you receive several sensors, we will only cover three important ones: the ultrasonic, colour and gyro 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. Ultrasonic waves are sent out in a cone from the front on the sensor, allowing it to see obstacles that are directly in front and a little bit to each side. 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 colour sensor (Fig. 3c), often referred to as a light sensor, has several modes. It can determine the colour of an object, the intensity of light reflected by said object or the ambient light level. The second mode, measuring the intensity of reflected light, is useful for detecting the black lines on the wooden panels your robots will drive on. Similarly, determining the colour of an object can help identify it during competition. In order to work properly, the sensor needs to be placed at a very specific distance from the object, which you will need to determine. Other than that, it is much more reliable than the ultrasonic sensor. Lastly, the gyro sensor (Fig. 3b) allows you to measure the angle and angular velocity of the robot. It is not used often, but may be useful for this year's project.[4] All sensors connect to the ports labelled 1 - 4 on the bottom of the brick.

[4] Further details will be provided during the semester.



(a) Ultrasonic

(b) Gyro

(c) Light

Figure 3: Useful sensors.

2.4 The "Brick"

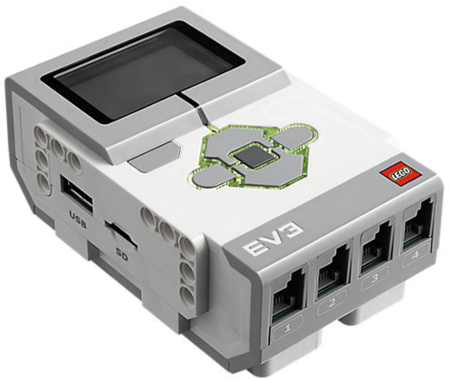


Figure 4: The LEGO "brick".

The "Brick" you receive, shown in Fig. 4, is an embedded system as described in Section 1. The important features are as follows:

- **LCD**: Used to display information.
- **Speaker**: Makes beeps, useful in some labs to let you know what your code is doing. Also capable of playing the *Imperial March* from Star Wars.
- **Micro-SD Card slot**: For the SD card with leJOS on it.
- **USB type-A**: Used by the USB WiFi adapter, necessary for the final project.
- **USB mini-B**: Used to communicate with your computer to upload code.
- **Ports A-D**: Compatible with the RJ-12 wires you are provided for connecting motors.
- **Ports 1-4**: Same as the motor ports, except used for the various sensors.

- **Buttons**: Used to interact with your code. The top left button is referred to as the `Back` button, while the centre button is called `Enter`. The remaining buttons are self-explanatory, being labelled `Left`, `Right`, `Up` and `Down`.

The brick can be turned on by pressing the `Enter` button for a second or two. In leJOS, it can be turned off by pressing the `Escape` button and confirming the shutdown command.

2.5 Defective components

If you believe one of your components to be defective, you can exchange it for a new one at the counter where you originally picked up your kit. Most problems are caused by coding errors or loose wires, though, so you may want to double check those before going to the counter. A TA can help you during lab hours if you have problems with some parts.

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. Install the Java SDK on your machine. At the time of writing, the latest version can be found at `http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html`. We have confirmed Java 8 to work without issue. On Linux, OpenJDK 8 is acceptable. We will not provide support if you encounter issues with other versions. If Java 8 is already installed, make sure both the JDK and JRE are up to date.

2. Install Eclipse from `https://www.eclipse.org/downloads/`. The installer will ask you what version to install; *Eclipse IDE for Java Developers* is sufficient for this course. At the time of writing, we have confirmed the 64-bit Oxygen version to work without issue. As stated above, we will not support other versions.

3. Download the file `leJOS 0.9.1 Complete Package` from MyCourses and extract the contents. The directory structure should look like this:

   - Documentation
   - Mac or Linux
   - SD Card (Method 1)
   - SD Card (Method 2 and 3)
   - Windows

4. If you're using Windows, run `leJOS_EV3_0.9.1-beta_win32_setup.exe` from the Windows folder to install LeJOS on your PC[5][6]. If you're using Mac or Linux, copy the `leJOS_EV3_0.9.1-beta` folder found within the `Mac or Linux` folder to wherever you want to install it; there is no installation program. Keep track of where you install it, though: you will need to specify it in the next step.

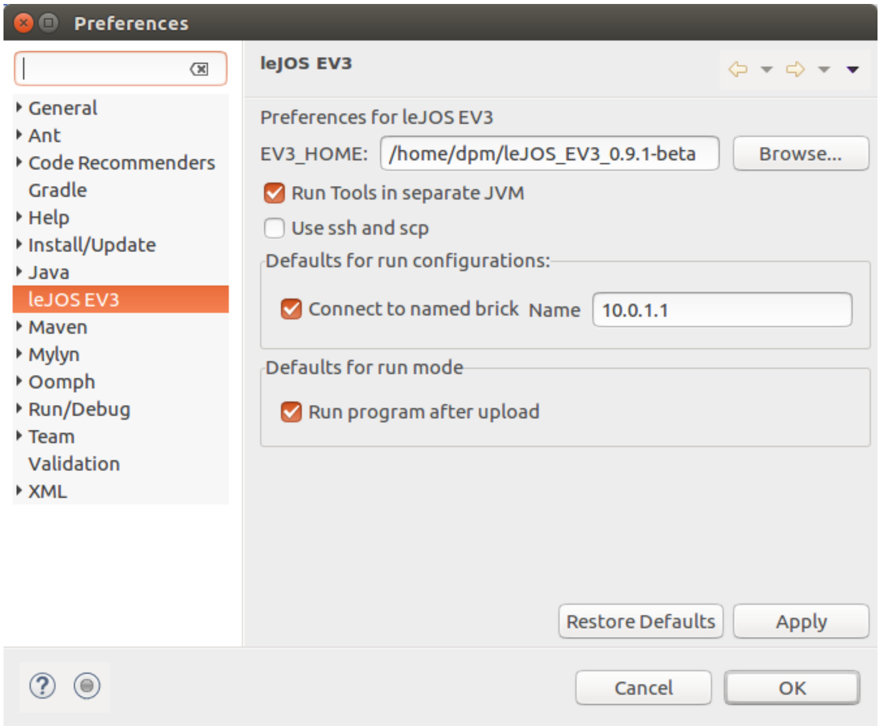
5. Install the Eclipse plugin. Open Eclipse, and navigate to `Help→Eclipse Marketplace`. Search for `leJOS EV3 Plug-in` and install it. You can ignore the warning about unsigned content. After installation, you need to tell the plugin where you installed leJOS as well as how to connect to your brick. The plugin settings page is called `leJOS EV3` and is part of the Eclipse `Preferences` window, which can be accessed via `Window→Preferences` on Windows/Linux or `Eclipse→Preferences` on Mac. 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
   - The box "Run Tools in separate JVM" should be checked

[5] If you have old versions of Java installed, make sure the installer uses the latest one.

[6] Feel free to check the boxes for the source and sample code; they can be easier to understand than the documentation

- Make sure "Use ssh and scp" is *not* checked
- Depending on your system, you may or may not have to check and fill the the "Connect to named brick" field. It usually works without it but specifying the IP as `10.0.1.1` (the default IP address when the brick is plugged into your computer via USB) is recommended.
- "Run program after upload" is up to you



leJOS EV3

Preferences for leJOS EV3

EV3_HOME: /home/dpm/leJOS_EV3_0.9.1-beta

☑ Run Tools in separate JVM

☐ Use ssh and scp

Defaults for run configurations:

☑ Connect to named brick Name 10.0.1.1

Defaults for run mode

☑ Run program after upload

Figure 5: The EV3 settings.

4 Installing leJOS to the SD card

The next step is to install leJOS onto the micro-SD card. Your kit may come with a micro-SD card already, but at minimum you will need to wipe out the old leJOS install from the previous semester. After an entire semester, most leJOS installations become corrupted in some way and should be replaced. Furthermore, we recommend getting your own micro-SD card so that you have a new card known to work well instead of one that may have been abused. 16GB SDHC cards have been confirmed to work; problems have been reported with anything bigger.

We present three ways to install leJOS, ordered in terms of complexity. You can use whichever method you like, as long as it works for your EV3 and computer. Method 1 is the simplest, but appears to only be reliable on Windows systems. Method 2 is not much more complicated but is only available for Windows and Mac. Method 3 is the most complex, but works on all systems and is required in cases of severe SD card corruption where the other methods fail. Regardless of which method you use, if you encounter problems a TA in the lab will be happy to help you.

1. Insert the SD card into a card reader connected to (or part of) your computer.
2. (a) **Method 1**:
   i. In Eclipse, navigate to `leJOS EV3`→`Start EV3SDCard`
   ii. Select your SD card in the option `Select SD drive`. Be careful to select the correct device, as otherwise you could *wipe out data or your operating system on your computer*.
   iii. Click on the button labelled `Zip file` and select the `lejosimage.zip` file in the folder `SD Card (Method 1)`.
   iv. Click on the button labelled `JRE` and select the `ejre-7u60-fvs-.....tar.gz` file in the folder `SD Card (Method 1)`.
   v. Click on `Create` and wait for it to write to the SD card. Fig. 6a shows an example of the tool right before formatting.
   vi. Safely eject the card using the appropriate method for your operating system.

   (b) **Method 2**:
   i. Download the *SD Memory Card Formatter* from `https://www.sdcard.org/downloads/formatter_4/index.html` and run it.
   ii. In the `Select card` field, pick the drive that corresponds to your SD card. As with Method 1, make sure you select the correct device as you could *wipe out data or your operating system on your computer*. For `Formatting options`, you can select `Quick format` while the `Volume label` field can be set to whatever you wish. Finally, click the `Format` button to start the process. An example screenshot of the options can be seen in Fig. 6b.
   iii. When the format is complete, the program should report the file system as FAT32 as shown in Fig. 6c. If it reports something else, the format has failed and you may need to try another method.

   iv. Copy all files from the folder `SD Card (Method 2 and 3)` to the SD card.
   v. Safely eject the card using the appropriate method for your operating system.

(c) **Method 3**:

   i. Use a partition manger or similar tool to 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. You can assign whatever label you wish.

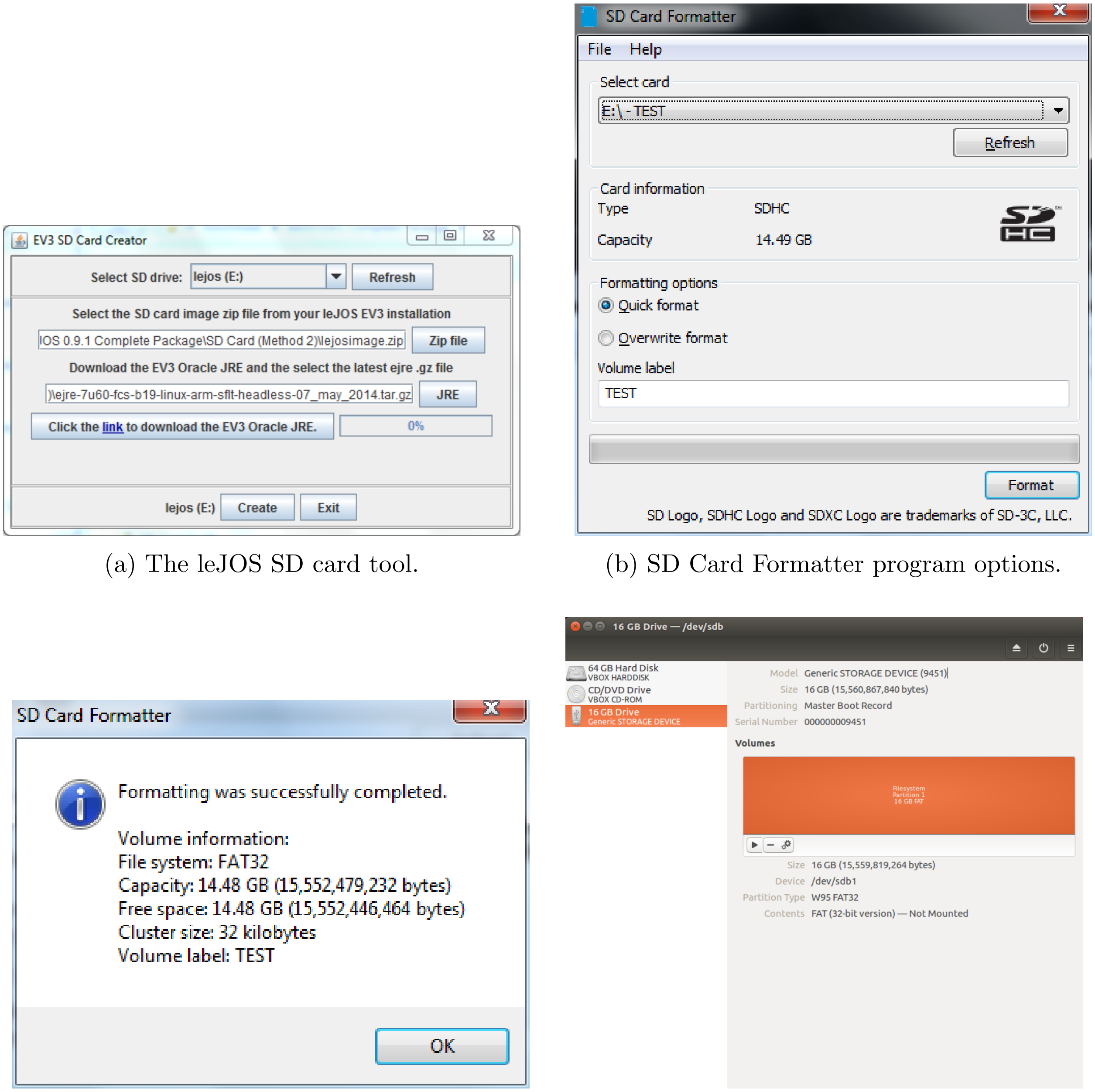
      Your operating system should have a built in tool that can do this, although it may be uncooperative at times. As such, you may need to find a third party tool that can do it for you. The built in tools for Windows and Ubuntu have been shown to work, although it may take some fiddling to convince them to format the card. The Mac tool, unfortunately, formats the SD card in such a way that the EV3 will freeze after installation. For cases of serious SD card corruption, GParted under Linux is recommended as it appears to be the most reliable tool. Regardless of what tool is used, the final partition layout of the SD card should look like that shown in Fig. 6d. If you are not familiar with modifying partitions, you may wish to consult one of the numerous online guides or tutorials on the topic. As with all of the previous methods, *take care not to accidentally delete important data or your operating system.*

   ii. Copy all files from the folder `SD Card (Method 2 and 3)` to the SD card.
   iii. Safely eject the card using the appropriate method for your operating system.

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

4. Turn the brick on and wait for leJOS to go through the installation procedure, which will take about 8 minutes. If you do not see the leJOS logo after a few seconds and your brick instead loads the Mindstorms environment, it is likely that your SD card has not been formatted properly. You will need to repeat the above procedure once more and perhaps try another method or tool.

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



Select SD drive: lejos (E:)

Select the SD card image zip file from your leJOS EV3 installation

lOS 0.9.1 Complete Package\SD Card (Method 2)\lejosimage.zip

Download the EV3 Oracle JRE and the select the latest ejre .gz file

)\ejre-7u60-fcs-b19-linux-arm-sflt-headless-07_may_2014.tar.gz

Click the link to download the EV3 Oracle JRE. 0%

lejos (E:)

(a) The leJOS SD card tool.

Select card

E:\ - TEST

Card information

Type SDHC

Capacity 14.49 GB

Formatting options

● Quick format

○ Overwrite format

Volume label

TEST

SD Logo, SDHC Logo and SDXC Logo are trademarks of SD-3C, LLC.

(b) SD Card Formatter program options.

Formatting was successfully completed.

Volume information:
File system: FAT32
Capacity: 14.48 GB (15,552,479,232 bytes)
Free space: 14.48 GB (15,552,446,464 bytes)
Cluster size: 32 kilobytes
Volume label: TEST

(c) A successful format using the SD Card Formatter.

Model Generic STORAGE DEVICE (9451)
Size 16 GB (15,560,867,840 bytes)
Partitioning Master Boot Record
Serial Number 000000009451

Volumes

Size 16 GB (15,559,819,264 bytes)
Device /dev/sdb1
Partition Type W95 FAT32
Contents FAT (32-bit version) — Not Mounted

(d) The expected SD card partition layout, shown using the Ubuntu Disks utility.

Figure 6: Various SD card related screenshots.

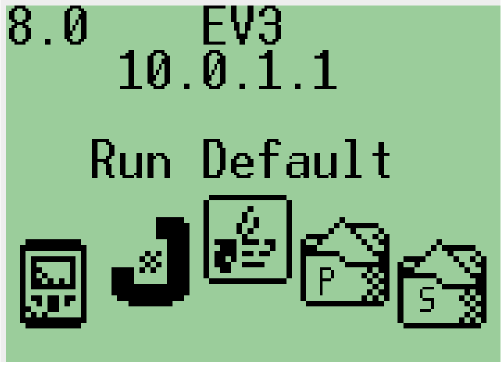


Figure 7: The EV3 screen after leJOS installs.

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. USB is the simplest method and the only one that will work with the desktop computers in the lab, so we will focus on it:

1. Turn the brick on.

2. Plug in the brick to your computer with the provided USB cable.

3. Check your network settings as appropriate for your operating system. There should be a new interface with an IP address in the `10.0.1.x` range in addition to your normal WiFi or Ethernet connections. If you either don't see a new interface or it does not have an IP address in the correct range, your computer cannot see the EV3. This is likely a driver, USB port or firewall issue - a TA in the lab can help if you have difficulty.

4. Create a new Eclipse project as follows:

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

   (b) Select `leJOS EV3 Project`

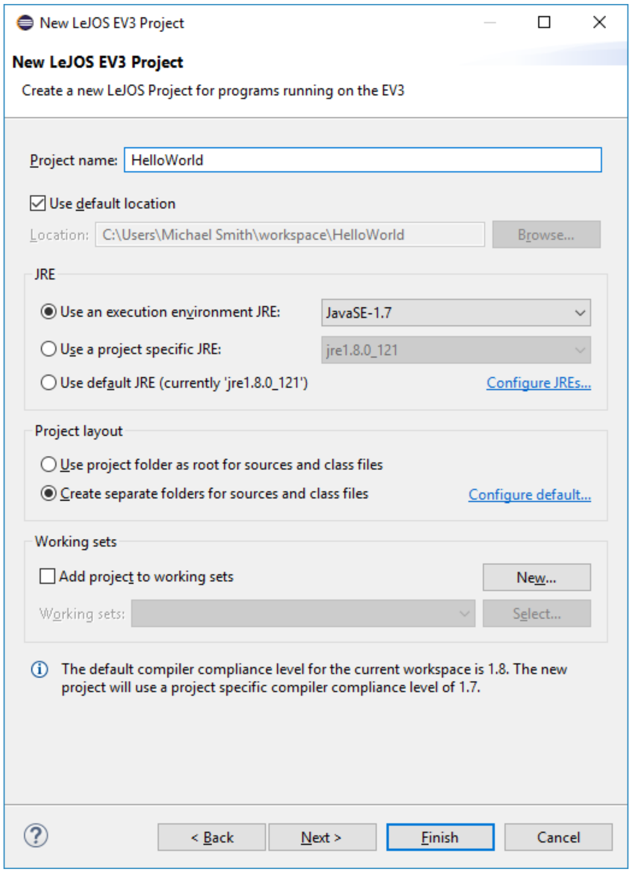
   (c) Name it `HelloWorld` and select `JavaSE-1.7` for `Use an execution environment JRE`[7]. Click `Finish`. An example of the page is shown in Fig. 8.

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

   (e) Add some code so that your program prints to the LCD and waits for a button to be pressed to exit[8]. Listing 1 provides an example.

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

[8] 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.



New LeJOS EV3 Project

Create a new LeJOS Project for programs running on the EV3

Project name: HelloWorld

☑ Use default location

Location: C:\Users\Michael Smith\workspace\HelloWorld

JRE

● Use an execution environment JRE: JavaSE-1.7

○ Use a project specific JRE: jre1.8.0_121

○ Use default JRE (currently 'jre1.8.0_121')

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

□ Add project to working sets

Working sets:

ⓘ The default compiler compliance level for the current workspace is 1.8. The new project will use a project specific compiler compliance level of 1.7.

Figure 8: Creating a new EV3 project.

```
package ca.mcgill.ecse211.HelloWorld;

import lejos.hardware.Button;

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

        public static void main(String[] args) {
                // Print to LCD
                System.out.println("Hello World!");
                // Wait for a button press before exiting
                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.

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 (Step 2 in 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 re-install the battery.

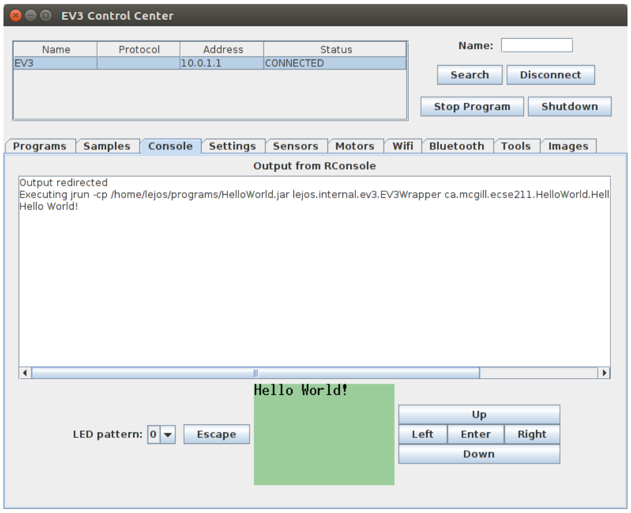
The last two steps don't allow leJOS to shutdown nicely, and can cause the SD card to get corrupted. If you experience weird behaviour after trying one of the last two steps, you may need to reformat your SD card and re-install leJOS.

7 Useful information

While the above discusses all the essentials for your first few labs, there are several points that can come in handy later:

1. The IP address displayed in the middle of the LCD screen (`10.0.1.1` in Fig. 7) is the address you use to connect to the brick. This is what you should specify in the settings page (Fig. 5) for the setting `Connect to named brick`. It is usually `10.0.1.1` but it can change if you use methods other than USB such as WiFi.

2. 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`[9]. 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 watch the output on your laptop rather than the small LCD, useful when you need to examine a lot of data. Fig. 9 shows what the console looks like when running the `HelloWorld` program. Feel free to explore the other tabs as well.



EV3 Control Center

Name	Protocol	Address	Status
EV3		10.0.1.1	CONNECTED

Name:

Output from RConsole

```
Output redirected
Executing jrun -cp /home/lejos/programs/HelloWorld.jar lejos.internal.ev3.EV3Wrapper ca.mcgill.ecse211.HelloWorld.Hell
Hello World!
```

```
Hello World!
```

LED pattern: 0

Figure 9: The EV3 control panel.

3. 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`.

4. Alternatives to using the USB cable include Bluetooth and WiFi. They can be useful for understanding what your code is doing while it's running on the wooden floor.

[9] 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.

- **Bluetooth**: Once you've paired the EV3 with your laptop, you can upload code and use the EV3 control panel as you would over USB. It may take a while after pairing for your laptop to properly connect to the EV3, though, so you will need to be patient.
- **WiFi**[10]: There is an access point in the lab with the following settings:
  - **SSID**: DPM
  - **Password**: dddpppmmm

  Power off the EV3, plug in the USB WiFi adapter provided to you, turn the EV3 back on and use the WiFi menu on the LCD to find the access point and connect to it[11]. Connect your laptop to the access point and note the new IP address shown on the third line of the EV3 LCD; it should be in the `192.168.2.x` range. Set this IP address in the EV3 settings page (Fig. 5) and in the EV3 control panel, which will allow you to treat the EV3 as if it's connected via USB.

  Note that once WiFi is connected, the USB port may not work correctly e.g. you may not be able to upload code. Whether or not this actually happens is random, but if it does occur it can always be fixed by unplugging the USB WiFi adapter and rebooting the EV3, allowing you to use USB again.

5. If there are too many users in the lab, Bluetooth can fail and the WiFi access point can crash or otherwise exhibit very strange behaviour so we recommend USB if possible.

6. You can change the EV3's name via the LCD (`System`→`Change name`). This can be useful if several groups use Bluetooth and you don't know which robot is yours. This name can also be used instead of an IP address in the `Connect to named brick` field (Fig. 5) to identify your robot. It is not always reliable, however, so don't be surprised if it doesn't work on your machine.

7. Documentation for all leJOS functions can be found in the `Documentation` folder in the zip file mentioned in step 3 of Section 3. Simply open the `index.html` file within using your favourite 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.

[10] You will be required to use WiFi in the final project. Further details and relevant code will be provided then.

[11] Other access points like your phone or home router will work as well. The McGill WiFi network will not as the encryption it uses it not compatible with the EV3.