Open Day: Lego Robot Programming

Dr Ramsay Taylor

Introduction

The aim of this exercise is to give you some practical experience of programming. During this session you should get to compile and run an existing program, understand what it does, then modify the program to do something that you want it to do. Ultimately, all the programming that you will ever do comes down to getting the computer to do what you want by giving it the right instructions.

Playing with Lego is purely incidental :)

Getting Started

The Student Ambassadors should help you through these stages — its the programming that's the interesting bit!

SETTING UP THE ROBOT
Your team should collect a robot from the staff. Each robot has a name — they aren't very imaginative: it will be a letter and a number such as A5 or G4.

The robot includes an EV3 "brick" and a box of components. In the box should be a pre-built set of wheels with two large motors. Attach the EV3 brick to the wheel structure and plug the motor wires into sockets A and B. Also in the box are various sensors. We will be using the colour sensor, which has a window on the front with a light and a sensor. Attach the colour sensor to socket 1 and find a way of attaching it to the front of the robot.

CONNECTING THE ROBOT
1. Make sure that the robot is turned on (press the dark-grey button on the front of the EV3 micro-computer until the light comes on.
2. Wait for a menu to appear on the robot.
3. On your PC screen, right click on the Bluetooth icon at the bottom right corner (the *system tray)* and select Show Bluetooth Devices.
4. If your robot is present in the list already proceed to stage 7 — look for the name on the sticker, which should be A5, or F7 or something similar.
5. If not, right click on the Bluetooth icon or on the Bluetooth device window and select Add a device
6. When asked whether a number appears on the device select *Yes* and *Next*, **even though no number appears**. Continue to select *Next* and/or *Close* through several more information screens
7. Right click on your robot in the list of Bluetooth devices and select Connect Using Access Point

The robot is now connected to the PC, so we can start controlling it with programs.

COMPILING AND RUNNING THE SAMPLE PROGRAM

Download the zip file from http://staffwww.dcs.shef.ac.uk/people/R.Taylor/OpenDays/robots.zip. Make a folder for the files somewhere sensible (the Student Ambassadors can make suggestions) and unpack the zip file. This should contain ColorFinder.java, which is the program we will be working with, and a load of files ending in .jar — Java Archives, which contain all the extra software you need to connect to the robots.

Click on the start menu, under programs find the JEdit folder, and open the editor. Open the ColorFinder.java file and have a quick look. This is a java program with a lot of comments to explain what's going on (comments are lines that start with `//`, the rest of the line is then ignored by the compiler and can contain helpful description).

Now, in the same section of the start menu, run the "Command prompt with Java JDK". This should give you a black command prompt window. Change directory to wherever you unpacked the zip file. Once you are in the right place run `setupconsole.bat`, which should run for a moment and then return to the prompt. This has set up various bits of the *environment* so that the java compiler and virtual machine know where all those .jar files are.

In the black command window type `javac ColorFinder.java` (note the American spelling of colour…) and press enter. This should also run for a few moments and then return to the prompt. If it prints lots of error messages ask your Student Ambassador what went wrong! This has *compiled* the Java program. The *source code* that you can see in the editor is (fairly) readable by humans, so you can understand it and modify it. However, the computer needs instructions in a *binary* form (1s and 0s in particular patterns that match its *instruction set*). The compiler converts the readable form to the binary form[1].

Assuming you connected the robot successfully you should now be able to run `java ColorFinder` (note: no `.java`). This will work for a moment, then it should say `EV3 found on IP xx.xx.xx.xx` where the xxs are probably 10.0.1.<something>. Then it will sit there for a little while as it sets up a connection to the robot. Eventually the robot's colour sensor should light up and it should start to rotate one way and then the other. For reasons passing understanding it sometimes fails the first time, so if it waits a really long time and then prints an error message just try it again…

The sample program instructs the robot to look left and right and search for something red. If you put something red close enough to the colour sensor then the robot should stop looking and make a rising pair of beeps. If it gets all the way left and right without finding anything red it will make a falling pair of beeps.

Exercise

If you can get the basic program running you can try and modify it to do something more interesting. Find the `main` method in the program — this is where the JVM will start running the program, so this is where you should add things that you want to have happen.

There are some methods defined at the bottom of the file that should allow you to do standard things without needing to manage all the details. The adding `goForward(300);` to the main method will cause the robot to go forward 300 "units" (the units are, technically, degrees of rotation of the motors, so 300 is less than one full turn of the wheels). The methods `turnLeft` and `turnRight` will rotate the robot on the spot (by moving one motor forward and the other backward). You have already seen the `scan` method, that will turn small steps in a given direction

[1] Technically, Java compiles to *bytecode* which is a binary form, but not the one used by any real computer. When you run `java ColorFinder` you are running the Java Virtual Machine (JVM), which *interprets* the byte code and executes the appropriate instructions on the real computer. This allows one Java byte code program to run on any machine for which you have a JVM. There are lots of JVMs — there is almost certainly one in your phone!

and check for the colour that you choose; `scan` stops as soon as it finds the colour, and leaves the robot pointing towards that direction.

To start with, try re-arranging the code thats there, or changing the *parameters* (like the number of steps it turns, or the colour it is looking for) and confirm that you can get the robot to do something different.

Remember: if you change the program you will need to save the file and re-run the java compiler (`javac`) before you re-run `java ColorFinder`

Once you can happily modify the program and get the robot to do something different try any or all of these ideas:

- Make the robot scan repeatedly and beep each time. Ask the student ambassadors about `while` loops.
- Make the robot stop scanning on a range of different colours and then make a different noise for different colours. Ask the student ambassadors about *boolean expressions.*
- Make a better search strategy — just scanning 20 steps left and then 40 right can take a while if the red object is just off to the right. Perhaps 1 left, then 2 right, then 3 left, then 4 right… Or maybe you can think of something better?
- Make the robot drive forwards when it detects red. Combine this with repeated scanning and driving and you should be able to get the robot to *follow* something red.
- Having got your robot to follow something red, join up with another team or the student ambassadors and see if you can get one robot to follow another (while the front one follows something). Since the colour sensors have a very short range in our brightly-lit lab you may have to go incredibly slowly for this to work. Next time you see those self-driving cars on the news and think they look a bit slow and boring, hopefully you will remember how hard this was :)