Programming EV3 with Java using LeJos

March 22, 2017

1 Introduction

Lego Mindstorms EV3 is a robotic kit with a graphical programming language that can be used by programming beginners to learn programming. It is developed by LEGO and National Instruments. Though it is powerful in nature, it is actually capable of so much more. The limited graphical programming language is just not enough for people to develop their talent in programming. Luckily, there are already community based projects that are available. LeJos and ev3dev are one of them. Using these projects, a bigger potential of the EV3 can be unleashed.

2 Setting up

LeJos will be used in order to be able to program the EV3 with Java (programming language). In order to use the LeJos, it must be set up in both the Eclipse plugin in the computer which the programming will take place and the EV3.

2.1 Setting up the EV3

A blank SD card will be needed to install LeJos and the card must be between 2GB and 32GB. Note that SDXC cards will not work. It is strongly recommended to use Windows to perform the installations. These are the crucial steps that must be done.

1. Format the SD card.
2. Install LeJos on the SD card.
3. Mounting the SD card on EV3.

2.1.1 Formatting the SD Card

The SD card must be formatted with a FAT32 partition. It is strongly recommended to format the SD card even if it is a new card because there might be hidden partitions in the card that can be the cause of problems later on. To perform the formatting, the following program can be used. Link : `https://www.sdcard.org/downloads/formatter_4/`

2.1.2 Installing LeJos

Before any further steps, it is necessary for the computer in which the programming is going to take place to have Java installed. Both Java 7 or 8 is fine. Then, download the LeJos installer from the following link. Link : `https://sourceforge.net/projects/ev3.lejos.p/files/` Follow the normal procedures of installation until it has installed successfully. On the second step of the installation, choose the appropriate Java JDK folder that is in your computer when prompted. By right, it should be installed without any problem. For more information on the installation, refer to this link. Link : `https://sourceforge.net/p/lejos/wiki/Windows%20Installation/` Once the installation has completed, launch the program and you should see the EV3 SD Card creator utility pops out. If you have not downloaded the JRE for EV3 yet, simply follow the link provided in the utility to download the JRE. Then, include the JRE file in the utility, insert the SD card and start the process.

2.1.3 Mounting the SD Card

Once the above two steps are completed, insert the SD card into the EV3 along with a USB wifi dongle and start the EV3. It should now begin the installation of the necessary packages.

2.2 Setting up the Eclipse Plugin

The best way to develop a program for leJOS EV3 is through the Eclipse IDE. In order to do so, a plugin is needed. The Eclipse IDE can be downloaded from this following link. Link : `http://www.eclipse.org/downloads/` Open up the Eclipse IDE and choose the option "Install New Software" from the Help menu. Then, click add, name is appropriately and insert the following link for the repository URL. Link : `http://lejos.sourceforge.net/tools/eclipse/plugin/ev3` Once you have followed the normal procedure for the installation, it should ask to restart the Eclipse IDE. More information on the plugin installation part can be found in this link. Link : `https://sourceforge.net/p/lejos/wiki/Installing%20the%20Eclipse%20plugin/`

3 Programming using Eclipse

If everything is done correctly, a new option in the menu bar (leJOS EV3) should be seen. When creating project for the leJOS EV3, press SHIFT + ALT + N on the keyboard and choose Project. Do not choose Java Project. It would not work. Instead, choose LeJOS EV3 Project that can be found under the LeJOS EV3 folder. Type a project name and under the JRE section, select JavaSE-1.7 for the "Use an execution environment JRE" section. Once that is done, click Finish the programming part can be done.

4 Programming

4.1 Controlling the Motor

The leJOS EV3 library offers a wide range of build-in functions that can be used to control the hardwares of EV3. One of it is the motor hardware. The following is the code to enable the motor to run for 5 seconds and stop.

```
import lejos.robotics.RegulatedMotor;
import lejos.utility.Delay;
import lejos.hardware.motor.EV3LargeRegulatedMotor;
import lejos.hardware.lcd.LCD;

public class Motor{
    public static void main(String[] args){

    //Creates an instance of EV3LargeRegulatedMotor class from the library.
    //Parameter as the port where is the motor is plugged in.
    RegulatedMotor m = new EV3LargeRegulatedMotor(MotorPort.A);

    m.forward(); //Continuous forward movement of the motor.

    LCD.drawString("Motor is moving", 0, 4); //Display on the EV3

    Delay.msDelay(5000); // Pause the program for 5 seconds.

    m.stop(); //Stops the motor movement.
    }
}
```

To run the above program, click the Run option in the menu bar, choose "Run As" and click on the LeJOS EV3 Program. Make sure the EV3 is connected to the same network as your computer and the upload should start now.

4.2 Ultrasonic Sensor

In the assumption of the Ultrasonic Sensor is plugged in to the port number 1, the following code is made.

```
import lejos.hardware.port.Port;
import lejos.hardware.sensor.EV3UltrasonicSensor;
import lejos.hardware.sensor.SensorModes;
import lejos.robotics.SampleProvider;
import lejos.utility.Delay;
import lejos.hardware.Button;
import lejos.hardware.ev3.LocalEV3;
import lejos.hardware.lcd.LCD;

public class Ultra{
    public static void main(String[] args){

        //instantiate the port
        Port port = LocalEV3.get().getPort("S1");

        //instantiate the Ultrasonic EV3 sensor by passing the port parameter
        SensorModes sensor = new EV3UltrasonicSensor(port);

        //SampleProvider (SP) wil get an instance of this sensor in
        //measurement mode
        SampleProvider distance = sensor.getMode("Distance");

        //All of the sensors returns in float type
        //Initialize an array of float for samples
        float[] sample = new float[distance.sampleSize()];

        while(true){
           Delay.msDelay(2); //Pause for 2 miliseconds between each loop
           distance.fetchSample(sample,0); //fetch the sample from SP
           String s = String.valueOf(sample[0]); //Convert the float to string
           LCD.drawString(s, 0, 4); //Displays the value
           if(Button.ESCAPE.isDown()){
              //Exits the program when ESC button in EV3 is pressed
              System.exit(0);
           }
        }
    }
}
```

4.3 Simple Server

To make the learning process more interesting, a small server can be created and run in EV3 while using a normal Java application to communicate with it. Perhaps an application with GUI and buttons that is programmed to activate certain sensors or motors can be built. To demonstrate the connectivity between the client and server, the following program is built.

4.3.1 Server Side

```
import java.io.*;
import java.net.*;
import lejos.hardware.lcd.LCD;
import lejos.utility.Delay;
```

```
public class ServerSide{
    public static void main(String[] args){
        try{

            //Creates a server that listens on port 3000
            ServerSocket server = new ServerSocket(3000);

            //Creates a socket for the connection
            Socket s = server.accept();

            //Displays message when a client is connected
            LCD.drawString("Connected",0,5);

            //Pauses the program for 3 seconds before exiting
            Delay.msDelay(3000);

        }catch(Exception e){
         System.out.println(e);
        }
    }
}
```

The above program must first be run in the EV3 before running a client program.

4.3.2 Client Side

```
import java.io.*;
import java.net.*;

public class ClientSide{
    public static void main(String[] args){
        try{
            //Replace the "TheEV3IPAdress" with the real ip address
            //Establishes the connection to the server (EV3)
            Socket s = new Socket("TheEV3IPAddress", 3000);

            //These are to send messages to the server
            //It would not work because there is no input stream in the server
            DataOutputStream dout = new DataOutStream(s.getOutputStream());
            String message = "Testing";
            dout.writeUTF(message);
            dout.flush();
        }catch(Exception e){
         System.out.println(e);
        }
    }
}
```

Run the client program as a Java Application. Once the connection is established, the EV3 should display it's message and closes the application.

5 Conclusion

In a nutshell, a lot more can be developed using these tools. All of the things that are mentioned above are just the basic parts of what can be achieved more. These guides should be helpful for anyone who wishes to start their own projects using the EV3 Lego Mindstorm.