MIYbot Software

Programming the MIYbot

The MIYbot is easily programmed thanks to the fact that the main “brains” is the Adafruit Pro Trinket microprocessor circuit board. The Pro Trinket is part of the popular open-source Arduino family of embedded processor boards based upon the Atmel 328 processors. The Pro Trinket is heavily documented here: https://learn.adafruit.com/introducing-pro-trinket/overview and should be reviewed carefully.

As mentioned previously, the Pro Trinket is an Arduino; this means it can be programmed using the Arduino integrated development environment (Arduino IDE). Simply stated, the Arduino IDE is software that runs on Windows, Mac and Linux and allows the user to write, edit, compile and upload programs to Arduino (and thus Pro Trinket) circuit boards.

Getting Started

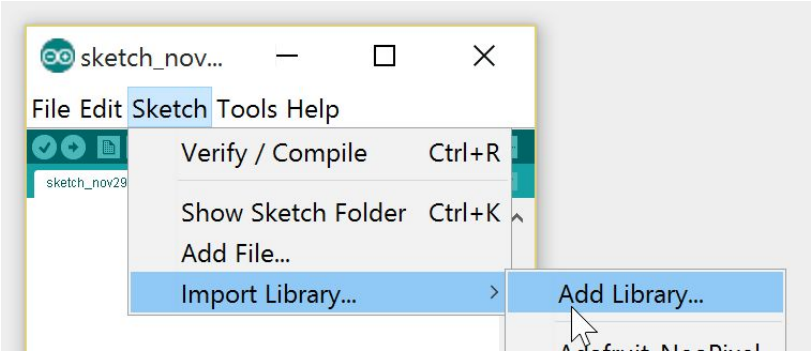
First, since the MIYbot is based upon the Pro Trinket, the instructions for programming a Pro Trinket should be followed. Begin by visiting web page https://learn.adafruit.com/introducing-pro-trinket/starting-the-bootloader and downloading the latest version of the Arduino specific to the Pro Trinket. Follow the instructions for uploading code via the USB bootloader and not the FTDI cables. You will also have to download and install the USB drivers using the directions on the same web page.

Once the Arduino IDE and drivers are installed, test the programming of the Pro Trinket by following the “Blink” instructions on page https://learn.adafruit.com/introducing-pro-trinket/setting-up-arduino-ide. If all goes well you will be program the #13 LED on the Pro Trinket to blink on and off. One common problem when programming the Pro Trinket is failing to press the reset button prior to compiling and uploading your code. Make sure you can upload and run the Blink code successfully prior to continuing. If you have trouble getting this working, re-read the Adafruit Pro Trinket web pages and also query the Adafruit forums.

Installing the MIYbot libraries

The MIYbot robot relies on several libraries to run the ultrasonic sensors, the line sensors and the buzzer. You will need to install these libraries into the Arduino IDE.

For the Ultrasonic Sensor, download the latest NewPing library zip file. The download can be found here http://playground.arduino.cc/Code/NewPing Once the zip file is downloaded, the library is installed by Arduino IDE menu item Sketch->Import Library->Add Library and choosing the NewPing.zip file you downloaded.



For the Line Sensor, download the latest QTR-RC Arduino library zip file. The download can be found here https://github.com/pololu/qtr-sensors-arduino Once the zip file is downloaded, the library is installed by

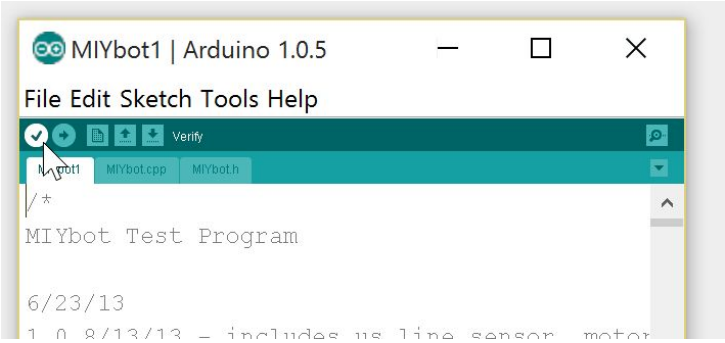
Arduino IDE menu item Sketch->Import Library->Add Library and choosing the QTRSensors.zip file you downloaded. More information on these sensors can be found here: https://www.pololu.com/product/2459

The buzzer needs to use the library TimerFreeTone because of conflicts with the servos and the PWM routines needed for the tri-colored LED to work. Download the latest TimerFreeTone library here: https://bitbucket.org/teckel12/arduino-timer-free-tone/downloads More information about the TimerFreeTone library can be found here: https://forum.arduino.cc/index.php?topic=235774.0 Once the zip file is downloaded, the library is installed by Arduino IDE menu item Sketch->Import Library->Add Library and choosing the TimerFreeTone.zip file you downloaded.

Installing the MIYbot software.

The MIYbot software can be downloaded on github page: https://github.com/bpwagner/MIYbot. You will find three files, MIYbot.cpp, MIYbot.h and MIYbot1.ino in a folder called MIYbot1. Put these three files in the same folder named MIYbot1 anywhere on your computer. You can load the MIYbot software into the Arduino IDE by double clicking the MIYbot1.ino file or using File-Open and choosing MIYbot1.ino. All three files need to be in the same folder!

Try compiling the software by clicking the check-mark icon. If it does not compile, make sure the libraries are all installed properly. You can also upload the software to your Pro Trinket if you want.



About the MIYbot software

The MIYbot is software that has evolved over the past three years. It is still a work in progress so you may see new features added and bugs fixed periodically. The software is contained in three files, MIYbot.cpp, MIYbot.h and MIYbot1.ino. MIYbot.cpp and MIYbot.h are library files used to abstract the entire robot so the programming of the MIYbot for novice programmers is easy. The MIYbot1.ino file is the Arduino “main” program that controls the robot.

If you look at the MIYbot.h file you will find the compiler #defines that describe the Arduino pins that the robot uses as well as many constants that are useful for programming (like the frequencies for musical notes) the MIYbot. You will also see the function definitions for the robot functions like motorRun(), setRGB() and beep(). The actual code for these functions is contained in MIYbot.cpp.

The MIYbot1.ino file contains the functions needed to run the MIYbot. The two functions void setup() and void loop() are common to all Arduino programs. Setup() is used to initialize the variables as needed and start the Arduino program. In the MIYbot’s case, the code initializes a few variables, the serial monitor and the servo configuration. Then it flashes the blue, green and red LEDs and beeps twice.

The loop() function works like a simple menu for the robot. Basically, the robot will cycle through many colors of the rainbow using one button and then runs a function when the other button is pushed. This way many programs can be written and run on the MIYbot. The remaining functions are colors: red(), orange(), yellow(), green(), blue(), purple(), white(), pink() and aqua(). As the menu cycles through the colors and one is chosen, the corresponding function is called.

Currently the colors are programmed as follows:
red() - drive around with the ultrasonic sensors, avoiding obstacles
orange() - test the ultrasonics. LEDs change colors when sensors 'see something'
yellow() - test the line sensors. Cover the line sensor with a finger to see LED colors change
green() - SumoBot fight - not programmed yet
blue() - To be programmed by student
purple() - To be programmed by student
white() - To be programmed by student
pink() - turn on servos so they can be centered using the tiny screw on the bottom of the servo.
aqua() - play the theme to Mario on the buzzer!

There is a small adjustment screw on the bottom of the servo that needs to be set prior to installing the servos. To adjust this screw, load the MIYbot software and run the pink() program. This will tell the servo to go to its center position. Adjust the screw on the bottom of the servo right or left using a jeweler's screwdriver until the servo stops moving. Now you can attach the servo to the body plate.

One final thing. The configuration of the servos needs to be set so the robot drives forward when it is programmed to drive forward. Depending on how the servos are physically mounted to the robot, this configuration can be one of 4 ways. The ways are both servos forward, left servo forward and right servo reversed, left servo reversed and right servo forward and both servos reversed. The configuration is changed near the top of the MIYbot1.ino file in the lines below. If your robot spins around instead of driving forward, you probably need to fix this!

```
//*****************************************
// Please set this value to 0, 1, 2, or 3
// Controls what initial direction of the servos
// 0 = both servos fwd
// 1 = left fwd, right reverse
// 2 = left reverse, right fwd
// 3 = both servos rev
//
int Servo_Config = 2;
//
//
//*****************************************
```

Conclusion
The MIYbot software is only at the beginning. Because the MIYbot is built on the Arduino platform, it has the potential to keep evolving. My hope is that others in the Arduino community will add to the MIYbot to make it more awesome!