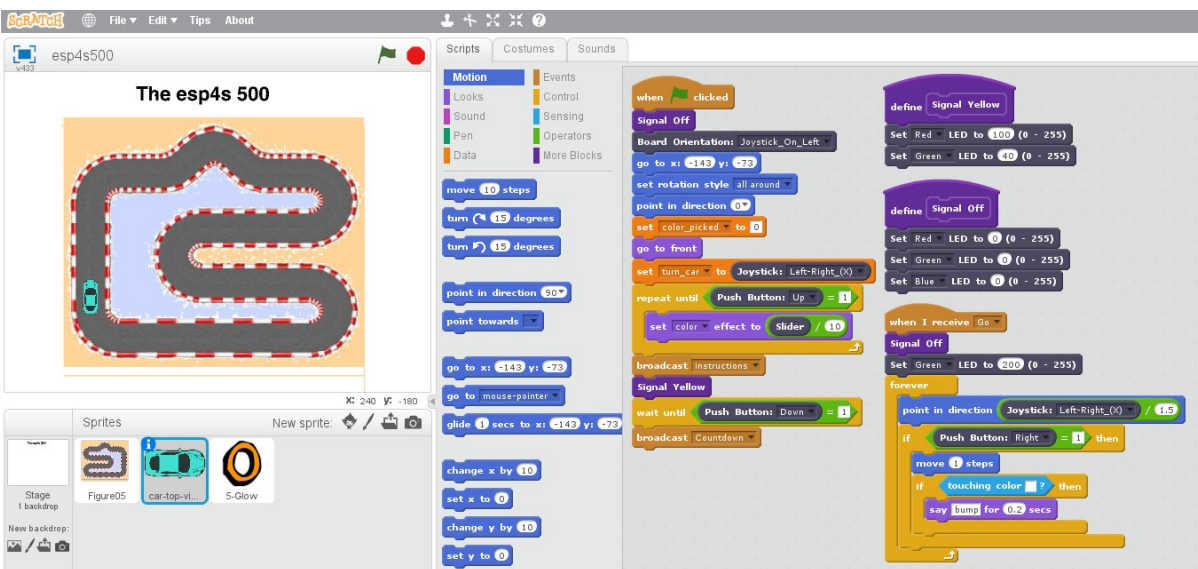


# Esp4s-aio

## A Scratch/Snap! HTTP Hardware Extension for the Arduino Esplora



**Copyright © 2015 Alan Yorinks. All rights reserved.**

**This manual is distributed WITHOUT ANY WARRANTY, without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. This manual is provided as is, without any warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.**

# Table of Contents

Introduction.....	1
An Out of the Box Hardware Platform.....	1
Esplora Sensors.....	2
Esplora Actuators.....	2
An Out of The Box Software Platform.....	2
Installation.....	3
esp4s.....	3
Python.....	3
pySerial.....	3
The Arduino Sketch.....	3
Running esp4s.....	6
Using Scratch.....	6
Scratch Projects Included.....	8
Using Snap!.....	8
Block Descriptions.....	10
Set Board Led.....	10
Select and Set RGB LED Brightness.....	10
Play Note.....	11
Play Continuous Tone.....	11
Set TinkerKit Output Level.....	11
Board Orientation.....	12
Temperature Units.....	12
Slider.....	12
Light.....	12
Temperature.....	13
Sound.....	13
Push Button.....	13
Joystick.....	13
Accelerometer.....	14
TinkerKit Input.....	14
Troubleshooting.....	14
Getting Help.....	15
Other Open Source Projects From Mr. Y's Lab.....	15

# Introduction

For many, nothing can be more fun than writing programs that control and monitor physical devices such as joysticks, LEDs, sound sensors, temperature sensors and the like.

But before the first line of code can be written, circuits need to be designed, parts need to be procured, and the hardware needs to be assembled and tested.

In addition, custom software is often required to provide a programming interface for the custom hardware. This software, like the hardware, needs to be designed, coded, and tested before integrating it with the hardware.

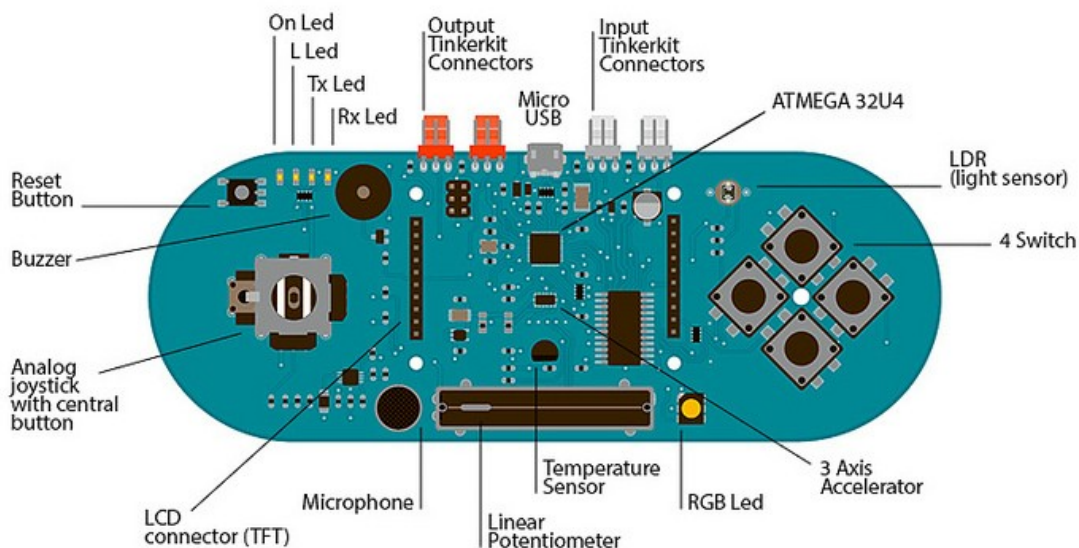
Designing and assembling both the hardware and matching software can be quite time consuming and is not without its obstacles.

Is there quicker way to writing that *first line of code*?

Yes! By combining an Arduino Esplora microcontroller with the esp4s library and either the Scratch or Snap! programming languages, that *first line of code* can be written in minutes!

## An Out of the Box Hardware Platform

The [Arduino Esplora](#) integrates sensors, actuators and a microcontroller all on a single board, making it ideal for quickly getting started. Nothing to design or test! Just take it out of the box, power up and you are good to go. The Esplora comes with a full set of sensors and actuators listed below.



## Esplora Sensors

- Joystick with center pushbutton.
- 4 pushbuttons laid out in a diamond pattern.
- Slider (linear potentiometer).
- Microphone for monitoring the amplitude of sound of the surrounding environment.
- Light sensor for monitoring ambient light levels.
- Temperature sensor to monitor the ambient temperature.
- 3 axis accelerometer to monitor device motion.
- 2 TinkerKit connectors to add additional sensors.

## Esplora Actuators

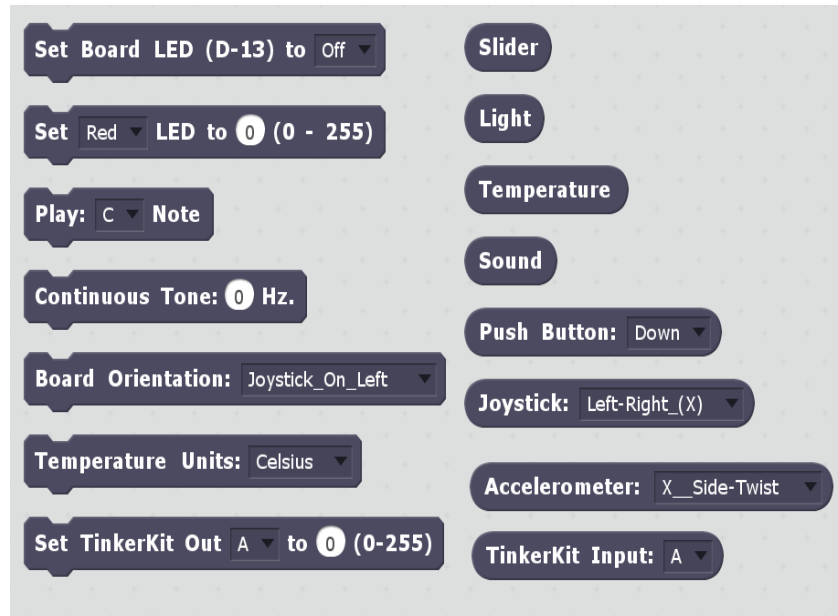
- Buzzer to generate sounds.
- RGB (Red, Green, Blue) LED
- Small Yellow LED
- 2 TinkerKit connectors to add additional actuators

## An Out of The Box Software Platform

Both Scratch and Snap! are wonderful graphical programming languages that simplify the programming effort. Both of these languages alleviate the need to deal with troublesome syntax issues. As a result, programmers quickly and intuitively get to the business of crafting programs. But there is still one problem to solve. Neither Scratch nor Snap! come equipped with programming blocks to control or monitor the Esplora.

And so, **esp4s** was born. It's a Scratch and Snap! extension that adds both custom Esplora programming blocks to the program editor, and a communications bridge to control and monitor the Esplora.

Below is a screen shot of the full set of esp4s programming blocks. All of these blocks are described in detail in the Block Description section of this document.



## Installation

### esp4s

Download and unzip the esp4s distribution to any directory you wish.

### Python

Before using the esp4s library, you need to have Python 3.4 installed on your computer.

Make sure that you have version 3.4 and not version 2. Version 2 is not compatible with the library. You can download python from: <http://python.org/>.

### pySerial

After installing Python, the next step is to install pySerial. You can download the latest version of pySerial here: <http://pyserial.sourceforge.net/>

Follow the instructions on the web page for installation instructions for your operating system.

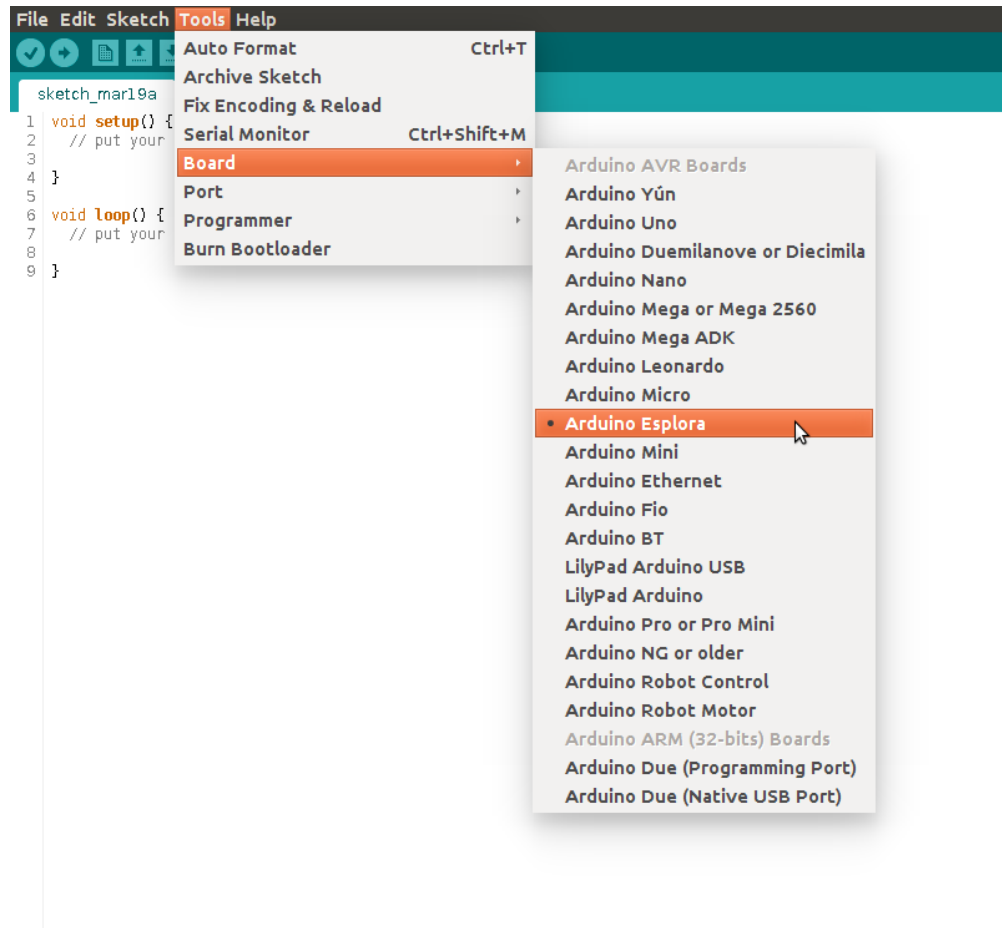
**Note: you may need administrative privileges to perform the install.**

### The Arduino Sketch

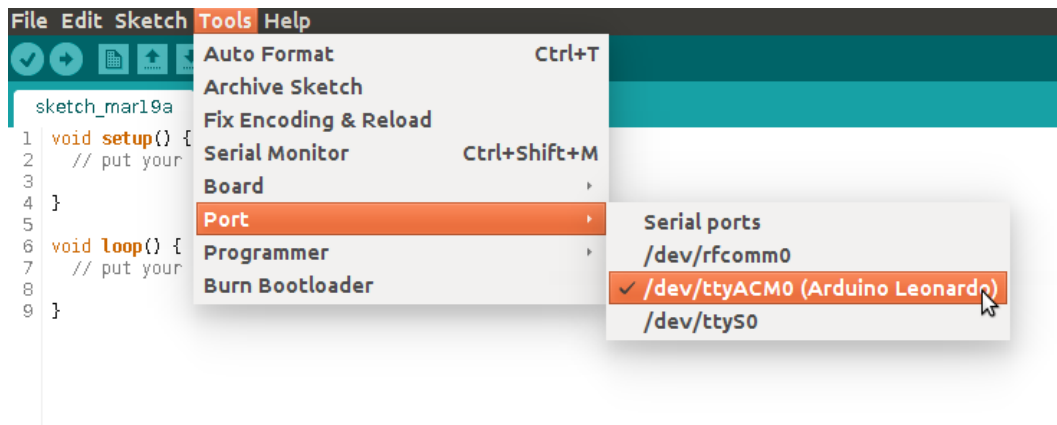
First you will need to install the Arduino Integrated Development Environment (IDE) from <http://arduino.cc/en/Main/Software>

Install all software including USB drivers if prompted by the Arduino installation program.

Next, open the Aduino IDE. Select Tools/Board and then select Esplora in the drop down list.



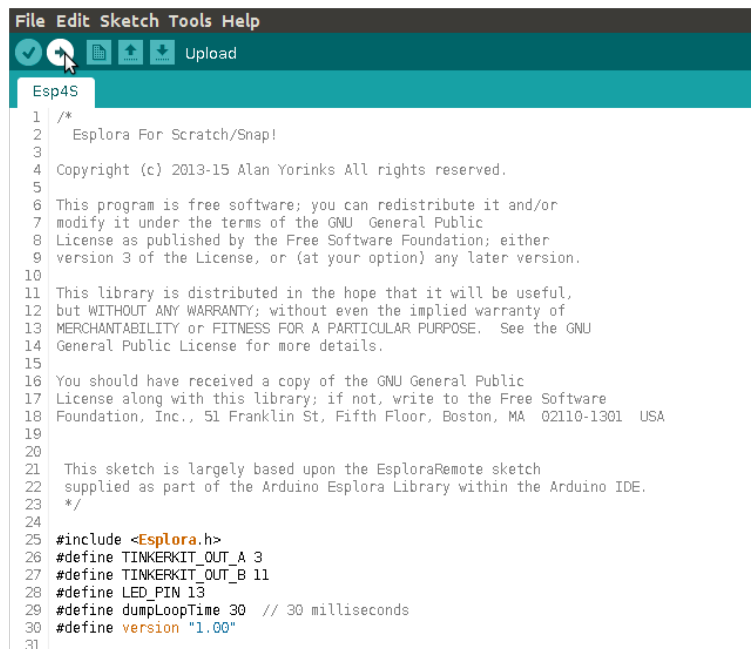
Then, from the Tools/Port menu, select the COM port for your computer. The IDE may show Leonardo as the identifier for the port.



Finally, upload the esp4s Arduino sketch to the Espora by doing the following:.

- Select File/Open and then using the file chooser, go to the directory where you extracted the esp4s files and select Esp4s.ino in the Arduino\_Sketch/Esp4s directory.
- Select the Upload button (right pointing arrow). When the upload completes, close the Arduino IDE.

You only need to do this once ,as long as you do not overwrite the sketch with another sketch at some other time.





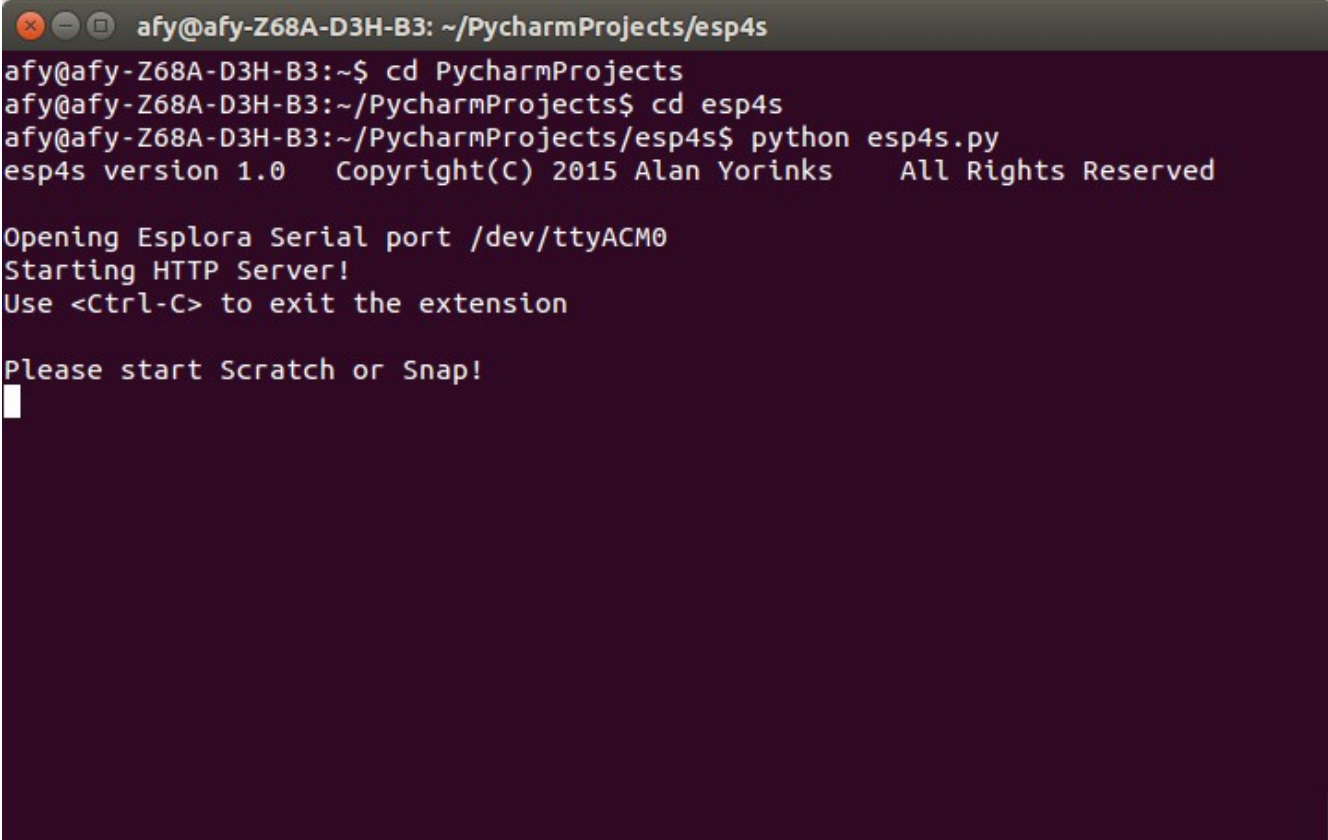
## Running esp4s

To use the default com port, open a command terminal and go to the directory that contains the extracted esp4s files.

Type: `python eps4s-aio.py`

To use a different com port (for example com3), specify it on the command line:

Type: `python eps4s-aio.py com3`

A terminal window with a dark purple background and white text. The window title bar shows a red close button, a yellow minimize button, and a green maximize button, followed by the text 'afy@afy-Z68A-D3H-B3: ~/PycharmProjects/esp4s'. The terminal content shows the following commands and output:

```
afy@afy-Z68A-D3H-B3:~$ cd PycharmProjects
afy@afy-Z68A-D3H-B3:~/PycharmProjects$ cd esp4s
afy@afy-Z68A-D3H-B3:~/PycharmProjects/esp4s$ python esp4s.py
esp4s version 1.0   Copyright(C) 2015 Alan Yorinks   All Rights Reserved

Opening Esplora Serial port /dev/ttyACM0
Starting HTTP Server!
Use <Ctrl-C> to exit the extension

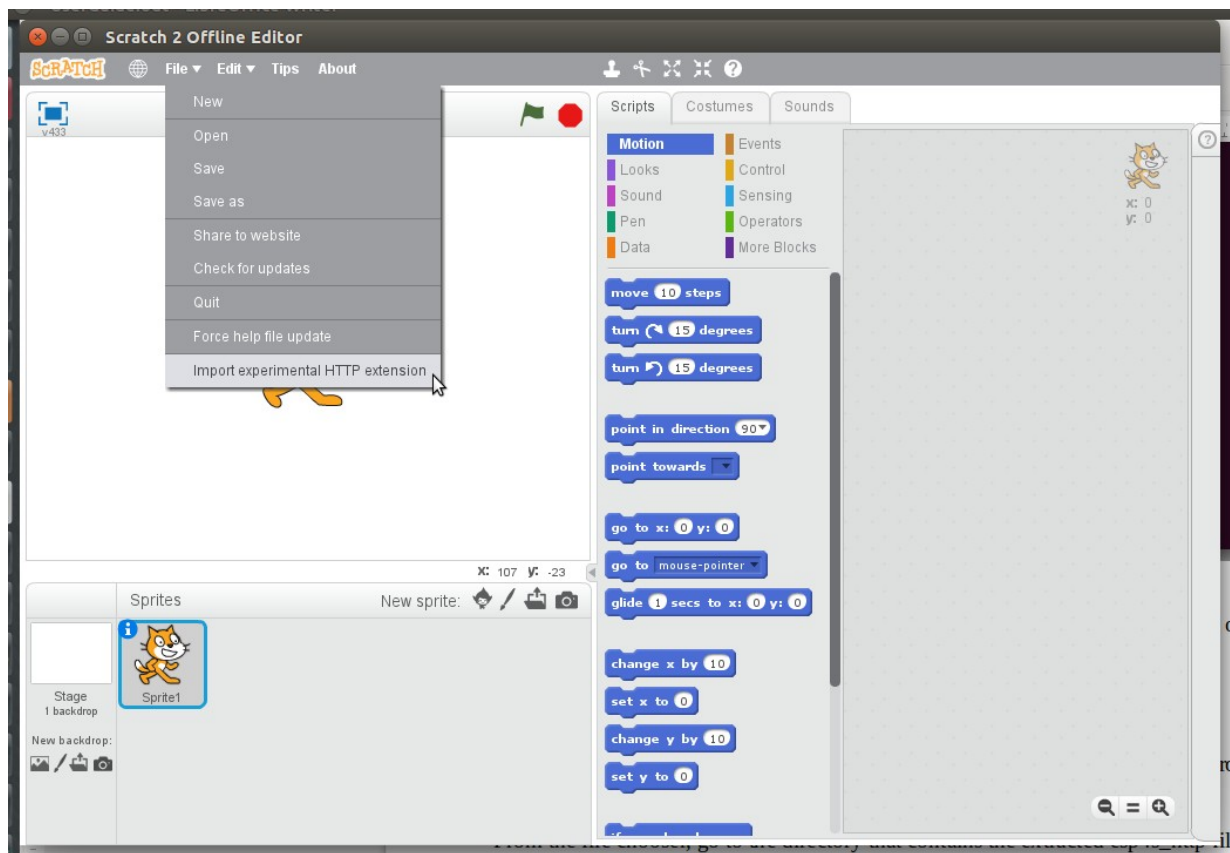
Please start Scratch or Snap!
█
```

In a few seconds you should see the welcoming message and instructions to start the off-line version of Scratch or the on-line version of Snap!.

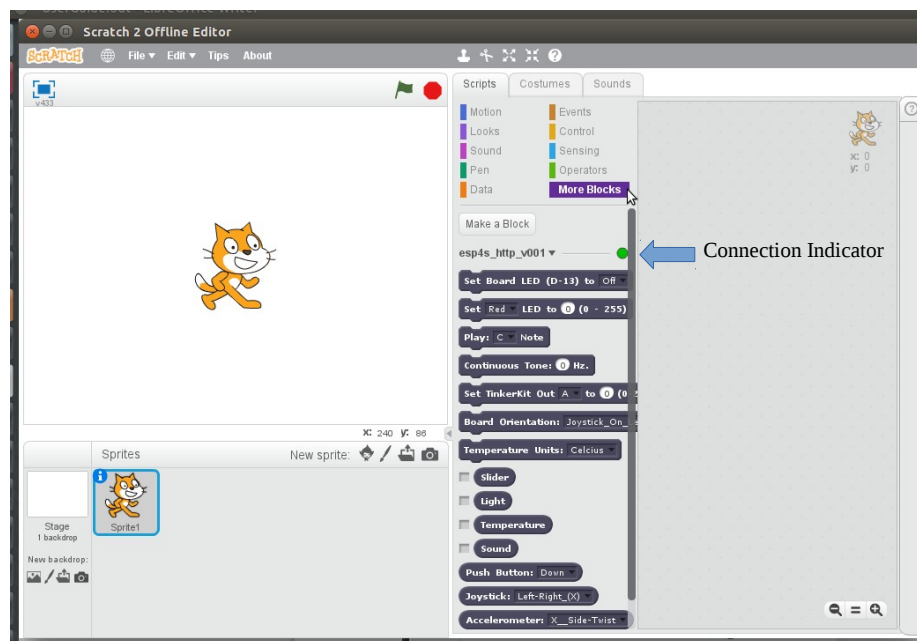
## Using Scratch

First, start eps4s. Then, if you haven't already done so, download and install Scratch from <http://scratch.mit.edu/scratch2download/>

Start Scratch 2.0 off-line. Then while holding down the Shift key, click on File. From the drop down list select “Import experimental HTTP extension”.



From the file chooser, go to the directory that contains the extracted esp4s files, and select ScratchFiles/ExtensionDescriptors/ep4s.s2e. If you then click on the More Blocks selector, you should see the esp4s blocks. The connection indicator should be green and you can start programming.



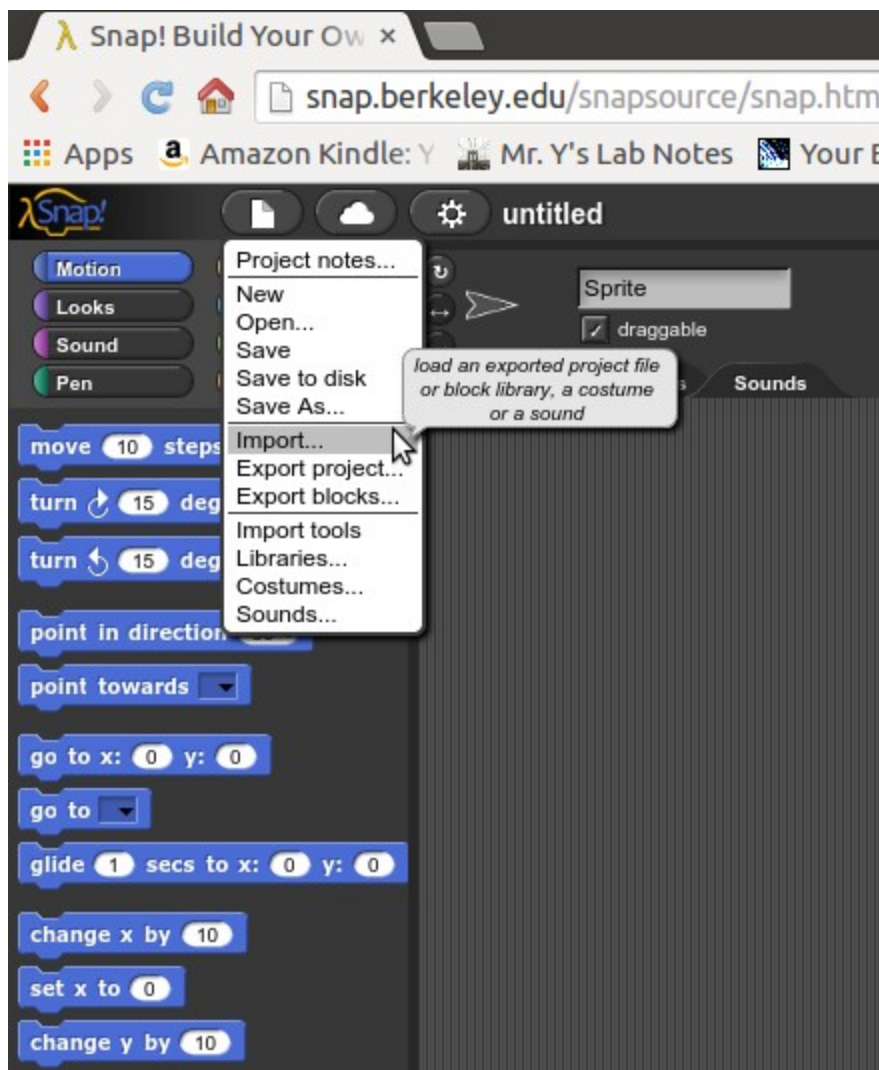
## Scratch Projects Included

There are two projects included in the ScratchFiles/ScratchProjects directory. One project is a race car game, the *eps4s500*, and the other is *twinkeTwinkle* that plays the first few notes of Twinkle Twinkle Little Star. To load and run the programs, start eps4s, and then open Scratch and from the Scratch File/Open menu, select the program you wish to run.

## Using Snap!

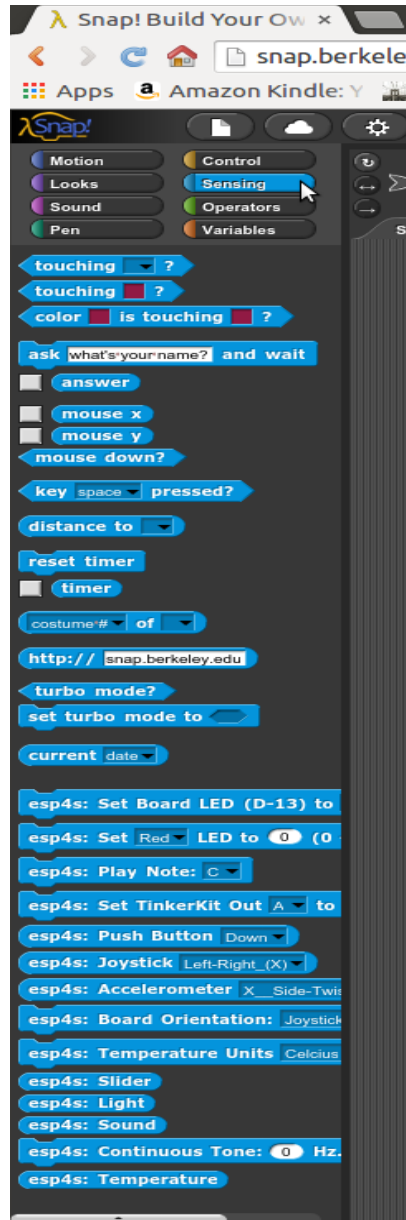
First start eps4s, then using your browser (either Chrome or Firefox), go to the Snap! Editor website <http://snap.berkeley.edu/snapsource/snap.html>

Select the File icon and then Import...



Using the file chooser, go to the directory where you extracted the esp4s files, and select Snap!\_Files/esp4s\_blocks.xml.

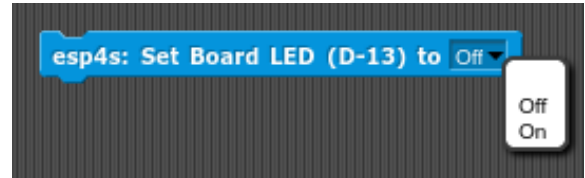
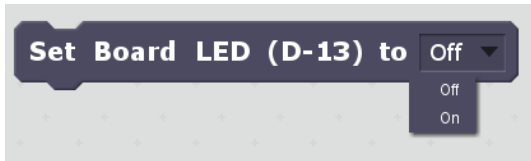
Click on Sensing and you will see all of the blocks identified with “esp4s:”.



## Block Descriptions

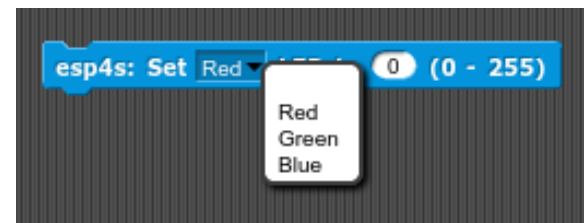
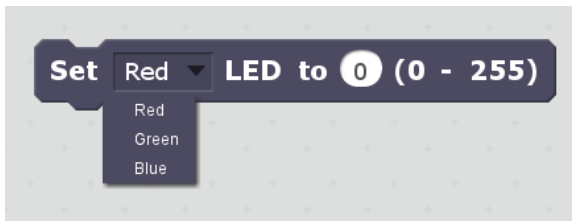
The black blocks are the Scratch blocks, and the blue blocks are for Snap!.

### Set Board Led



This block controls the small yellow “L” Led. Select either On or Off from the drop down list.

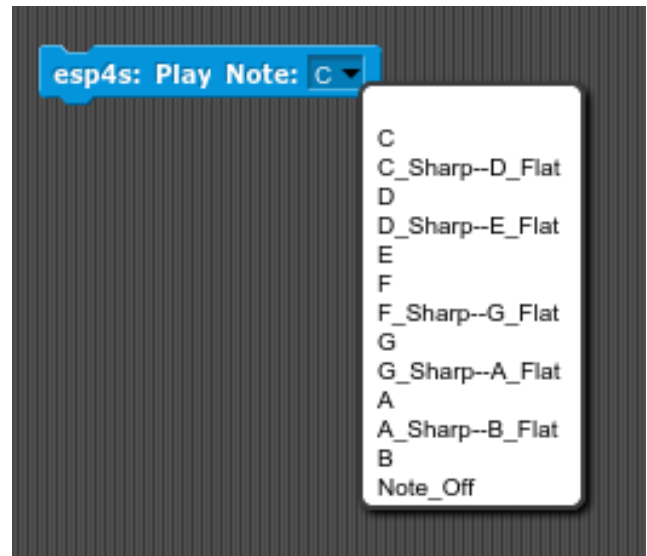
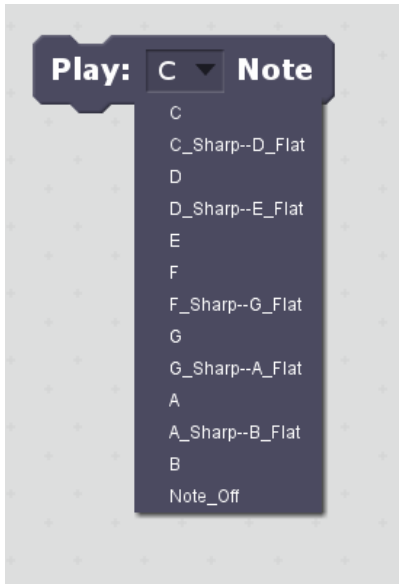
### Select and Set RGB LED Brightness



Select the LED color from the drop down list and then enter a brightness level. 0 is Off, and 255 is maximum brightness. You can turn on more than one LED at a time to mix colors.

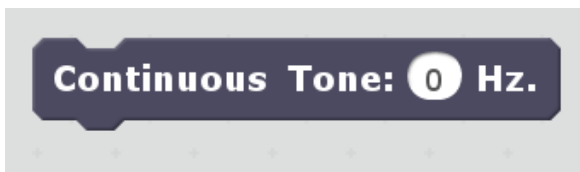
NOTE: If you use Play Note or Continuous Tone, the Red LED will no longer light until you reset the Esplora. For an explanation see: <http://arduino.cc/en/Reference/EsploraTone>

## Play Note



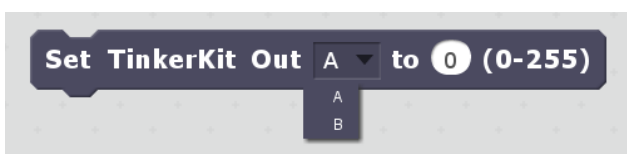
Select the note from the drop down list. The note will play continuously until you select Note\_Off.

## Play Continuous Tone



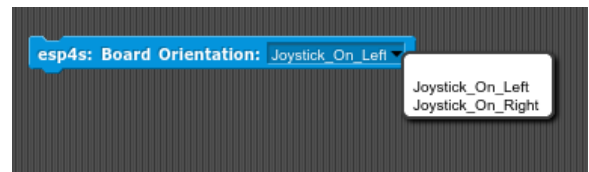
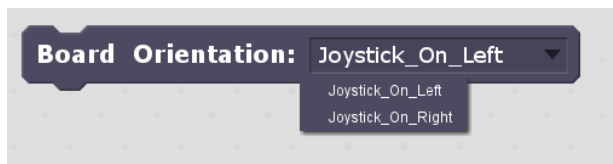
Enter the frequency of the tone you wish to play expressed in Hertz. To turn the tone off, set the frequency to zero or use the Play Note block and select Note\_Off.

## Set TinkerKit Output Level



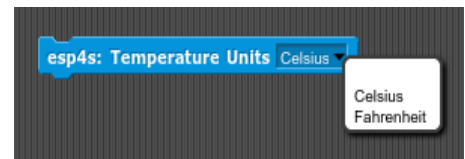
The Esplora has two TinkerKit output connectors to allow you to set the output level for either of the TinkerKit output channels. The output range is from 0 to 255. For more information on TinkerKit devices, go to [http://store.arduino.cc/index.php?main\\_page=index&cPath=16](http://store.arduino.cc/index.php?main_page=index&cPath=16).

## Board Orientation



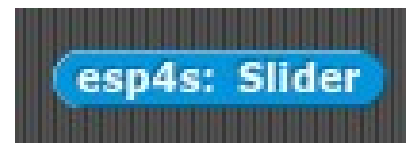
This block allows you to select board orientation so that the Joystick can be positioned to the right or left side of the board. This block maintains consistent operation for items such as the joystick, pushbuttons and slider, independent of board orientation.

## Temperature Units



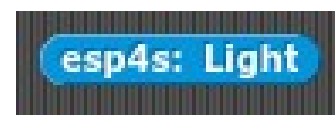
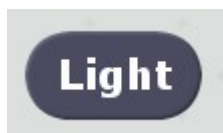
This blocks allows you to have temperature reported in degrees Celsius or degrees Fahrenheit.

## Slider



The Slider block reports the current setting of the slider. The extreme left position returns a zero and the extreme right position returns 1023. These positions are maintained when using the board orientation block.

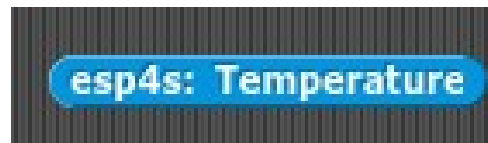
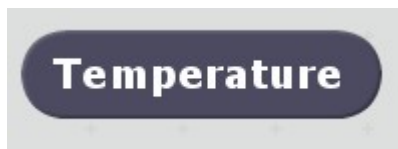
## Light



The Light block reports the current light level. Since light levels can vary greatly within a room, take a reading of the ambient light, and again when a light source is aimed at the sensor. Use low and high

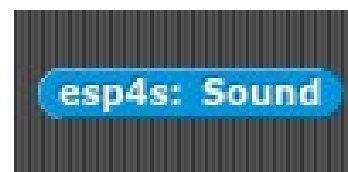
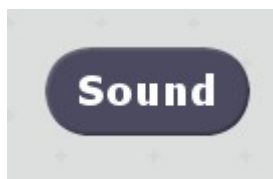
values as a basis for your application.

## Temperature



The Temperature block reports the ambient temperature. You can select the units, Celsius or Fahrenheit by using the Temperature Units block.

## Sound



The Sound block reports the ambient sound level received by the microphone. The range of values is 0 to 1023 (the loudest).

## Push Button



This block allow you to select and monitor any of the four push buttons. When a button is pressed, it returns a “1” otherwise it returns a “0”. The buttons maintain their orientation when using the Board Orientation block.

## Joystick





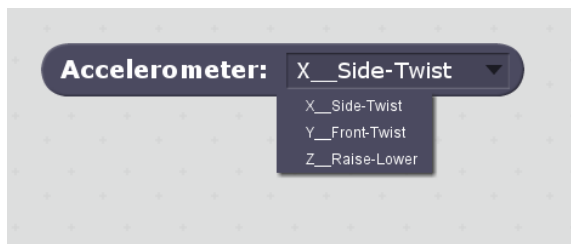
This block allows you to monitor the joystick x and y positions, in addition to the state of the pushbutton. The values returned are:

- X: -511 = maximum left, +512 = maximum right
- Y: +511 maximum Up, -511 Maximum down
- Button: 0 if not pressed and 1 if pressed.

Using the Board Orientation block assures that the values are consistent independent of board orientation.

## Accelerometer

This block allows you to select and monitor X motion (side to side), Y motion ( front to back) or Z motion (moving towards the floor or ceiling).



## TinkerKit Input



This block reports the current value of a TinkerKit sensor. You can choose either channels A or B, and the value returned is between 0 and 1023. For more information on TinkerKit devices, go to

[http://store.arduino.cc/index.php?main\\_page=index&cPath=16](http://store.arduino.cc/index.php?main_page=index&cPath=16)

## Troubleshooting

The most common problems occur as a result of forgetting to upload the esp4s Arduino sketch onto the Esplora and not plugging the Esplora into your computer for power and communication. If you need additional help, contact us at the link below.

## Getting Help

If you have any issues or need some help, you can contact us at:

[MisterYsLab@gmail.com](mailto:MisterYsLab@gmail.com)

If you would like to keep up with what we are doing, go to our blog: <http://mryslab.blogspot.com/>

or contact me on Twitter: Alan Yorinks@BrassFigLigee

## Other Open Source Projects From Mr. Y's Lab

Here is a list of links to some of our other open source projects:

[https://github.com/MrYsLab/s2a\\_fm](https://github.com/MrYsLab/s2a_fm)

<https://github.com/MrYsLab/xi>

<https://github.com/MrYsLab/PyMata>