fablab
waag society

so you want use:

Blynk and I2C

before you start:

Find a:

- -ESP 8266 based board
  example: WeMos D1 Mini
- -Computer with the Arduino IDE
- -A GY-85 or other I2C sensor
- -A phone the can run blynk

Get the following libraries:

- -ESP8266 core for Arduino
- -Blynk
- -GY-85 (or library for your sensor)

Have a basic underinstanding of blynk

- - blynk.cc
- -http://fablab.waag.org/project/iot-rapid-prototype-blynk

fablab waag society

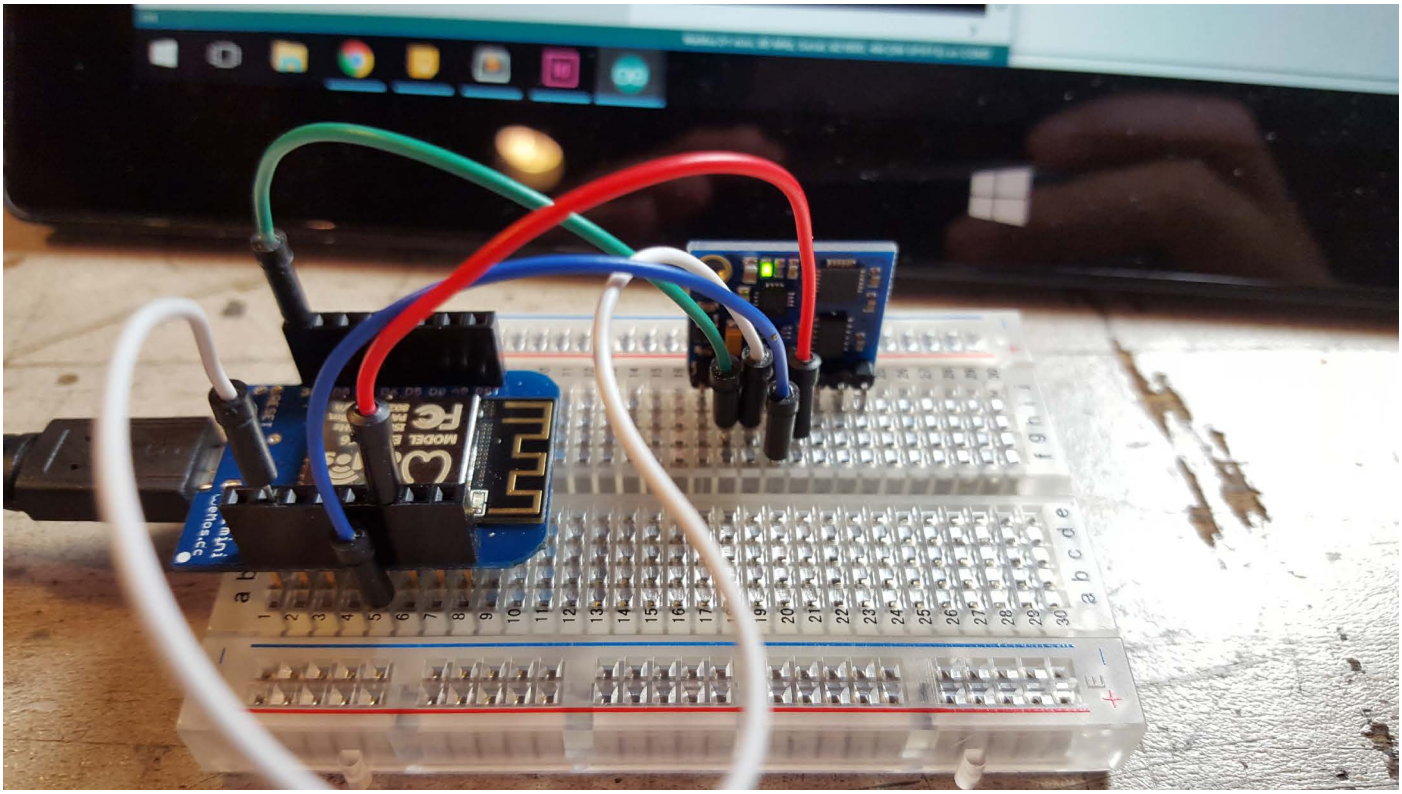
so you want to use:

Blynk and I2C

Sensing the world

Since the ESP8266 only has one Analogue to Digital converter, it's handy (and in most cases crucial) to get data from sensors communicating over I2C. In this case we're using the GY-85, a board with an accelerometer, gyroscope, and magnetometer start getting data from out I2C sensor, we first need to connect to it.
Start by opening the example form the GY-85 library (examples > GY-85 > example). You need to modify line 52 so it reads '`Wire.begin(D1,D2);`' during `void setup()`' This tells the (in this case) WeMos D1 Mini that we are using D1 for SDA and D2 for SCL.

Wiring it up!



To wire up our sensor, we only need to provide it with power and the I2C interface:

D1 mini	GY-85
5V	Vin
G	GND
D2	SCL
D1	SDA

Upload the sketch and open the serial monitor, it should be outputting the values read by the sensor.

adding Blynk

Once you have your sensor working, you can start sending the X axis accelerometer data to a phone using Blynk. In order to do this, we're going to add to the Blynk example for the esp8266: Examples > Blynk >BoardsAndShields > ESP8266_standalone.

The first thing we want to do is add:

```
#include "GY_85.h"
#include <Wire.h>
GY_85 GY85;
```

This adds the libraries and object for the GY-85. Next, we want to add some virtual pins so that we can send data to our phones using Blynk.

```
#define PIN_ACCEL V5
#define MAX_PIN V6
#define MIN_PIN V7
```

Then we need to set up some variables for sampling. Since the internet is 'slow' (slower than the serial connection from an Arduino to a computer when using Blynk) we need a way to 'slow down' the stream of information. We'll do that by keeping a bit of the data from the past, and then handing it off processed in 3 ways: the average, the minimum, and the maximum. First we set up an array to hold these values:

```
//set up the variables for the running average
const int numSamples = 100;
int counter = 0;
int runningArray [numSamples];
```

Remember to also add your Blynk auth token. During the 'setup' method, also start up the GY-85:

```
Wire.begin(D1,D2);
delay(10);
GY85.init();
```

Now we're going to create our methods for calculating the average, minimum, and maximum values:

```
int avgRun(){
  int sum = 0;
  for(int i=0;i<numSamples;i++){ //go thru all of the samples
    sum += runningArray[i];//add the samples to the sum of samples
  }
  sum = sum/numSamples;//divide the sum by the number of samples to get
an average
  return sum;
}
int minRun(){
  //to calculate the minimum, take the first value in the array
  int nmin = runningArray[0];
   for(int i=0;i<numSamples;i++){
    //compare it to the other numbers in the array
    if(nmin>runningArray[i]){
      //and if it's smaller, it's the new minimum
      nmin = runningArray[i];
    }
  }
  return nmin;
}
```

```
int maxRun(){
  //to calculate the maximum, take the first value in the array
  int nmax = runningArray[0];
   for(int i=0;i<numSamples;i++){
    //compare it to the other numbers in the array
    if(nmax<runningArray[i]){
      //and if it's larger, it's the new maximum
      nmax = runningArray[i];
    }
  }
  return nmax;
}
```

Now, we need to tie Blynk reading the virtual pins, to our values:

```
BLYNK_READ(PIN_ACCEL)
{
  Blynk.virtualWrite(PIN_ACCEL, avgRun());
}

BLYNK_READ(MAX_PIN)
{
  Blynk.virtualWrite(MAX_PIN, maxRun());
}

BLYNK_READ(MIN_PIN)
{
  Blynk.virtualWrite(MIN_PIN, minRun());
}
```

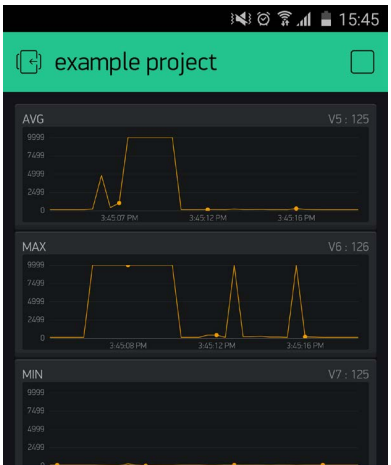
Finally in the loop, we need to record the value of the sensor and let Blynk run:

```
void loop()
{
  Blynk.run();

  //check to make sure you loop
  if(counter > numSamples - 1){
    counter = 0;
    Serial.println(avgRun());
    }
  runningArray[counter] = GY85.accelerometer_x( GY85.
readFromAccelerometer());
  counter ++;

}
```

The final thing we need to do is set up out Blynk app to have 3 graphs, one for each of the virtual pins.



full code:`