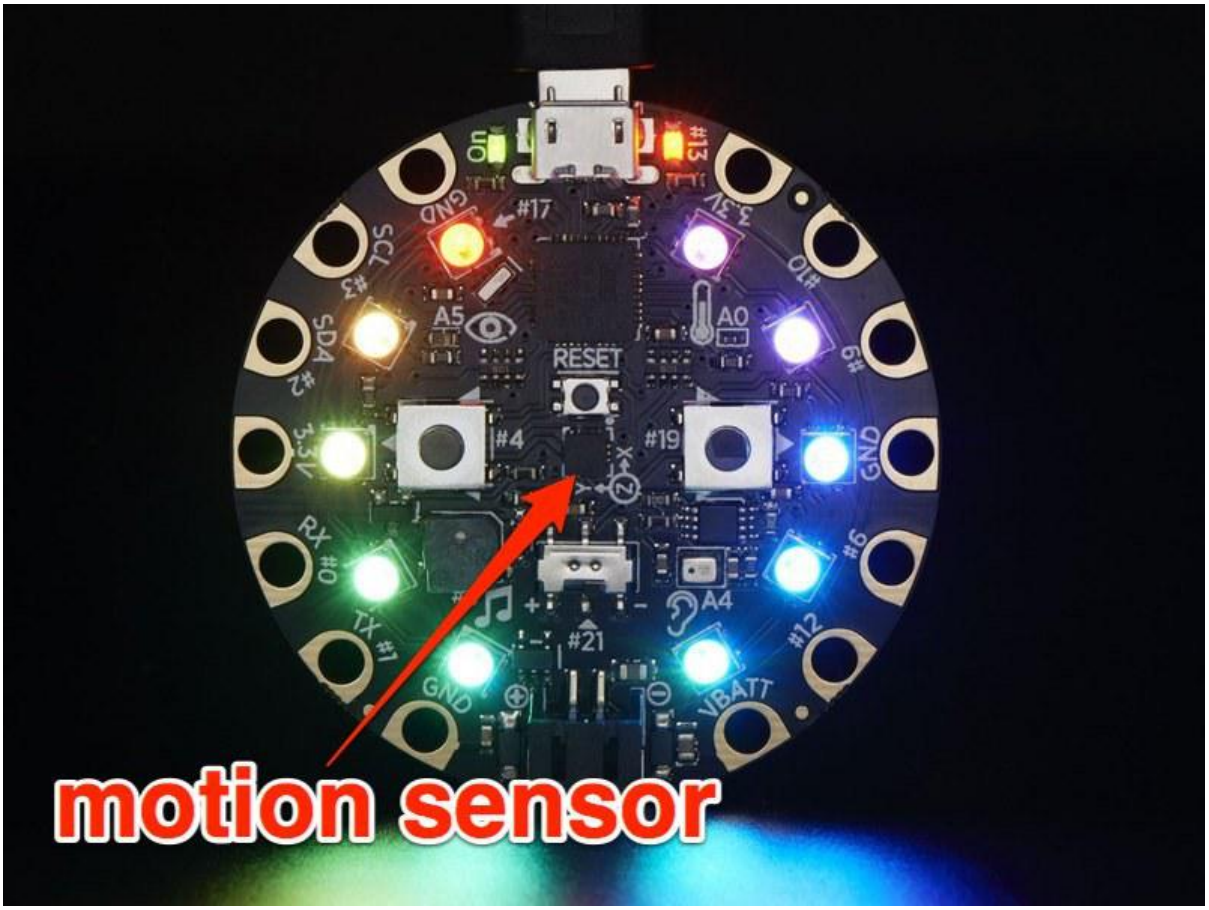
Circuit Playground's Motion Sensor

Created by Erin St Blaine



https://learn.adafruit.com/circuit-playgrounds-motion-sensor

Last updated on 2021-11-15 06:52:36 PM EST

Table of Contents

Intro & Setup

Circuit Playground is a really fun, really easy to use micro controller that is a great way to add sensors and interactivity to your project without a whole lot of fuss and soldering.

It's got a built in motion sensor, a sound sensor, a light sensor, and a temperature sensor, as well as a built in speaker and 10 neopixels, ready to use right out of the box. I'm just starting to explore the fun things this board can do.

Today we're going to focus on the motion sensor accelerometer.

Before You Start

If this is your first foray into the world of arduino-based microcontrollers, you'll need to install some software first. Head over to the Circuit Playground Lesson 0 guide (https://adafru.it/tGC) for detailed installation and setup instructions.

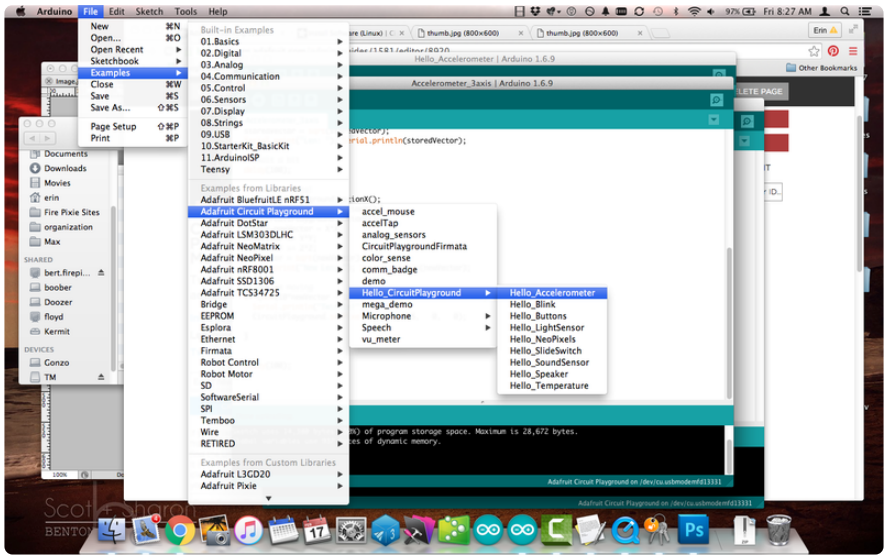
You'll only need to do all this once, so be a little patient if it seems like a lot! This is actually the hardest part.

Learning the Sensor

Once you've got everything installed and your computer can talk to the Circuit Playground, it's time to start playing with the sensor.

Plug your Circuit Playground into your computer and select the Circuit Plaground under Tools > Boards. Then select the Circuit Playground as the Port.

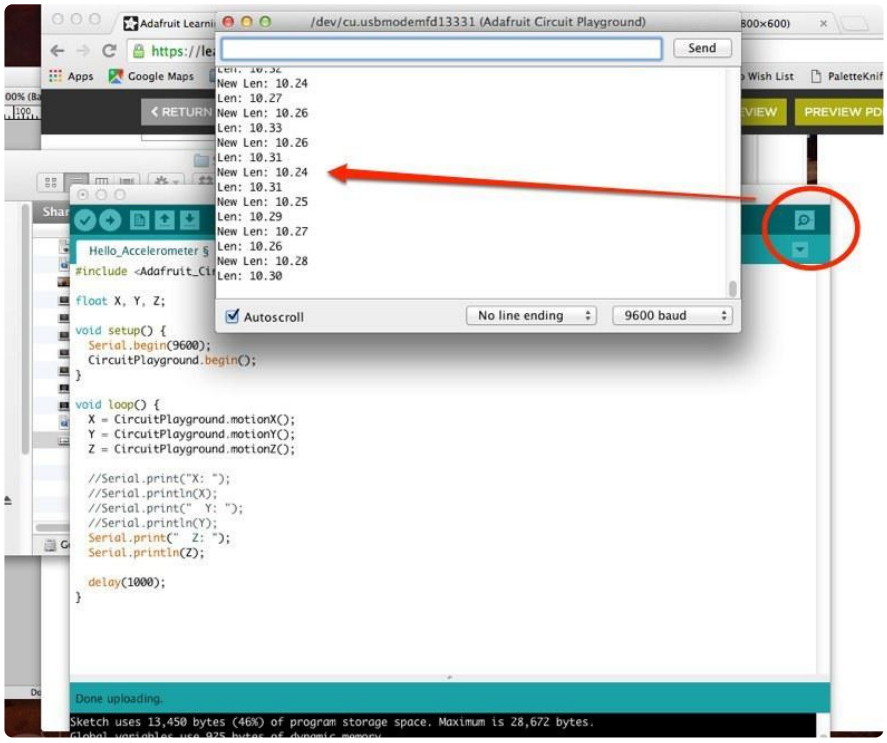
Next, open Examples > Adafruit Circuit Playground > Hello_Circuitplayground > Hello_Accelerometer.



Upload the code to your Circuit Playground by clicking the Upload button.

Now open your serial monitor in Arduino. The serial monitor gives you feedback from the board so you can test your code at a really basic level and make sure everything is working.

If all is well, you should see some numbers scrolling by in this window. You'll see values for x, y, and z -- these represent the three different directions this board can move. Keeping the circuit playground plugged in, pick it up and move it around a bit and watch to see how the numbers change.



Let's simplify even more. Go into your code window and comment out all the serial commands that talk about Y and Z -- we'll just focus on X for now.

While you're in there, change X's second "serial print" to a "serial print line" command. This will make a new line for each value in the serial monitor so it's easier to read. Upload the code again.

```
Serial.print("X: ");
Serial.println(X);
//Serial.print("  Y: ");
//Serial.println(Y);
//Serial.print("  Z: ");
//Serial.println(Z);
```

Take another look at your serial monitor. You'll see just one reading for X appear each second. Play around with moving the circuit playground through space again and watch what happens with the values on the screen.

Try moving the board in all three dimensions and see which one changes X the most consistently. With the board oriented the way I want it in my project, it looks like X changes the most dramatically when I turn the board clockwise or counter clockwise.

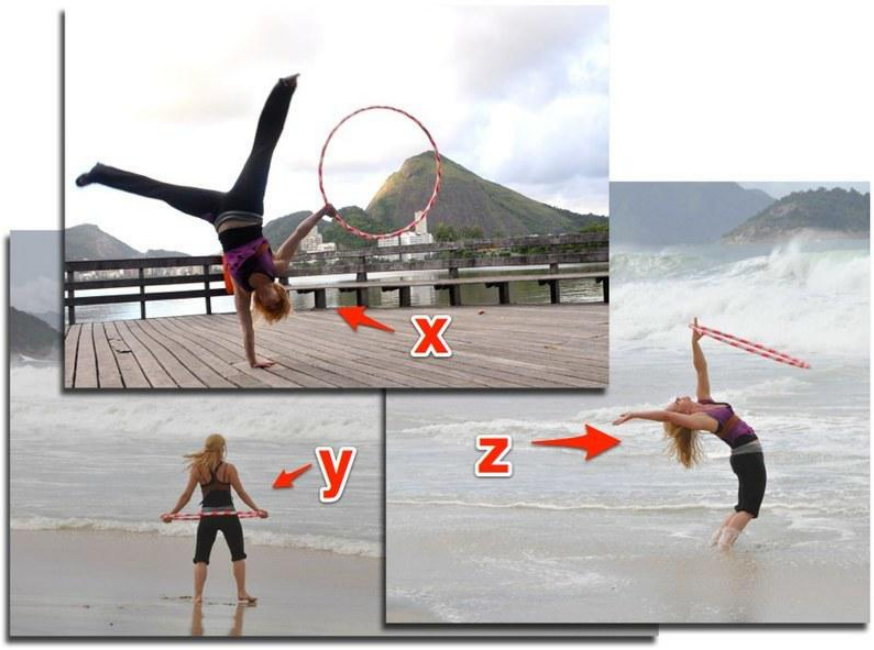
Go back to your code and uncomment only the Y lines, and upload again. Then get a feel for which axis of movement the Y direction controls. Looks like it's related to the board twisting left to right.

Try once more with Z. Z appears to control the back and forth motion.

Conclusions

To help imagine what this means in real life, I held the board against my hip, where it's going to live in my project, to visualize the real world orientation.

I learned that the data for X will measure my side to side motion, like when I do cartwheels. Y will measure whether I'm spinning around in circles. Z will be able to tell me if I'm doing forward bends or back bends.



Twinkle

I want to use a combination of all three directions of motion, so the lights come on whenever I move any which way. Rather than starting from scratch and having to do my own math, I've borrowed some code from the Sparkle Skirt guide (https://adafru.it/umD) and modified it to work with the Circuit Playground.

```
#include &lt;Adafruit_CircuitPlayground.h&gt;

float X, Y, Z;
#define MOVE_THRESHOLD 3

void setup() {
  Serial.begin(9600);
  CircuitPlayground.begin();
}

void loop() {
  //CircuitPlayground.clearPixels();
  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();
  Z = CircuitPlayground.motionZ();

   // Get the magnitude (length) of the 3 axis vector
  // http://en.wikipedia.org/wiki/Euclidean_vector#Length
  double storedVector = X*X;
  storedVector += Y*Y;
  storedVector += Z*Z;
  storedVector = sqrt(storedVector);
  Serial.print("Len: "); Serial.println(storedVector);

  // wait a bit
  delay(100);

  // get new data!
  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();
  Z = CircuitPlayground.motionZ();
```

```
  double newVector = X*X;
  newVector += Y*Y;
  newVector += Z*Z;
  newVector = sqrt(newVector);
  Serial.print("New Len: "); Serial.println(newVector);

  // are we moving
  if (abs(10*newVector - 10*storedVector) &gt; MOVE_THRESHOLD) {
    Serial.println("Twinkle!");
    //CircuitPlayground.setPixelColor(0, 255,   0,   0);
    delay(1000);
  }


  delay(100);
}
```

Upload this code to your circuit playground and watch the serial monitor as you move the board around. The word "Twinkle" will appear whenever the board moves far enough. Change the move threshold number to make it more sensitive or less sensitive.

Lastly, let's get out of the serial monitor and into the world. We can easily make one of the neopixels on the board twinkle whenever the move threshold is reached.

Uncomment the "clear neopixels" line of code at the beginning of the loop, and the "CircuitPlayground.setPixelColor" line near the bottom. Comment out the "delay(1000);" line just below the Set Pixel color command.

Upload your code again, and move your circuit playground around to make the light twinkle.

Twinkle ALL The Pixels!

Here is the complete Sparkle Skirt tutorial code modified to twinkle all 10 neopixels on the Circuit Playground. Go nuts!

```
#include &lt;Adafruit_CircuitPlayground.h&gt;

float X, Y, Z;
#define NUM_LEDS   10

// Here is where you can put in your favorite colors that will appear!
// just add new {nnn, nnn, nnn}, lines. They will be picked out randomly
//                                    R   G   B
uint8_t myFavoriteColors[][3] = {{200,   0, 200},   // purple
                                 {200,   0,   0},   // red
                                 {200, 200, 200},   // white
                                };
```

```
// don't edit the line below
#define FAVCOLORS sizeof(myFavoriteColors) / 3

// mess with this number to adjust TWINklitude :)
// lower number = more sensitive
#define MOVE_THRESHOLD 10

void setup() {
  Serial.begin(9600);
  CircuitPlayground.begin();
}

void loop() {
  //CircuitPlayground.clearPixels();
  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();
  Z = CircuitPlayground.motionZ();

   // Get the magnitude (length) of the 3 axis vector
  // http://en.wikipedia.org/wiki/Euclidean_vector#Length
  double storedVector = X*X;
  storedVector += Y*Y;
  storedVector += Z*Z;
  storedVector = sqrt(storedVector);
  Serial.print("Len: "); Serial.println(storedVector);

  // wait a bit
  delay(100);

  // get new data!
  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();
  Z = CircuitPlayground.motionZ();
  double newVector = X*X;
  newVector += Y*Y;
  newVector += Z*Z;
  newVector = sqrt(newVector);
  Serial.print("New Len: "); Serial.println(newVector);

  // are we moving
  if (abs(10*newVector - 10*storedVector) &gt; MOVE_THRESHOLD) {
    Serial.println("Twinkle!");
    flashRandom(5, 1);  // first number is 'wait' delay, shorter num == shorter
twinkle
    flashRandom(5, 3);  // second number is how many neopixels to simultaneously
light up
    flashRandom(5, 2);
  }
}

void flashRandom(int wait, uint8_t howmany) {

  for(uint16_t i=0; i&lt;howmany; i++) {
    // pick a random favorite color!
    int c = random(FAVCOLORS);
    int red = myFavoriteColors[c][0];
    int green = myFavoriteColors[c][1];
    int blue = myFavoriteColors[c][2];

    // get a random pixel from the list
    int j = random(NUM_LEDS);
    //Serial.print("Lighting up "); Serial.println(j);

    // now we will 'fade' it in 5 steps
    for (int x=0; x &lt; 5; x++) {
      int r = red * (x+1); r /= 5;
      int g = green * (x+1); g /= 5;
      int b = blue * (x+1); b /= 5;
```

```
      CircuitPlayground.setPixelColor(j, r, g, b);
      delay(wait);
    }
    // &amp; fade out in 5 steps
    for (int x=5; x &gt;= 0; x--) {
      int r = red * x; r /= 5;
      int g = green * x; g /= 5;
      int b = blue * x; b /= 5;

      CircuitPlayground.setPixelColor(j, r, g, b);
      delay(wait);
    }
  }
  // LEDs will be off when done (they are faded to 0)
}
```