Light Paintbrush with Circuit Playground

Created by John Park

https://learn.adafruit.com/lightpaint-cplay

Last updated on 2024-03-08 02:30:43 PM EST

Table of Contents

Overview 3

- Painting with Light

Code the Light Paintbrush 4

- Download the Code

MakeCode for Circuit Playground Express 7

- Lots of Coding Options
- MakeCode
- Input
- Colored LEDs
- Brightness Control

Assemble the Light Paintbrush 17

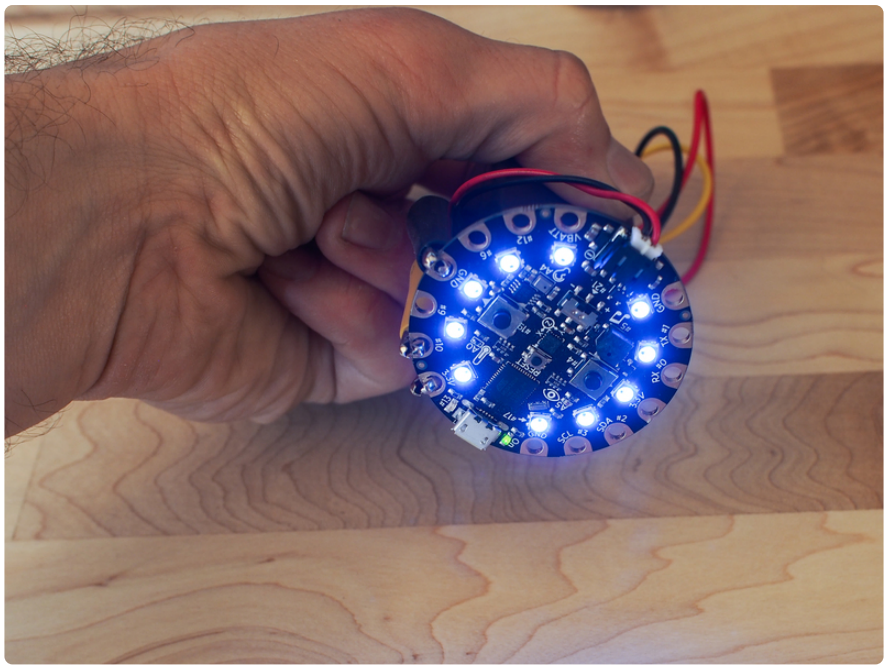
Paint with Light 23

Overview

Painting with Light

Streaks of colorful light, suspended in midair -- ever since people began to capture images on film we have had a fascination with light painting and light drawing. By simply moving a light source in front of an open shutter, motion over time could be captured on camera.

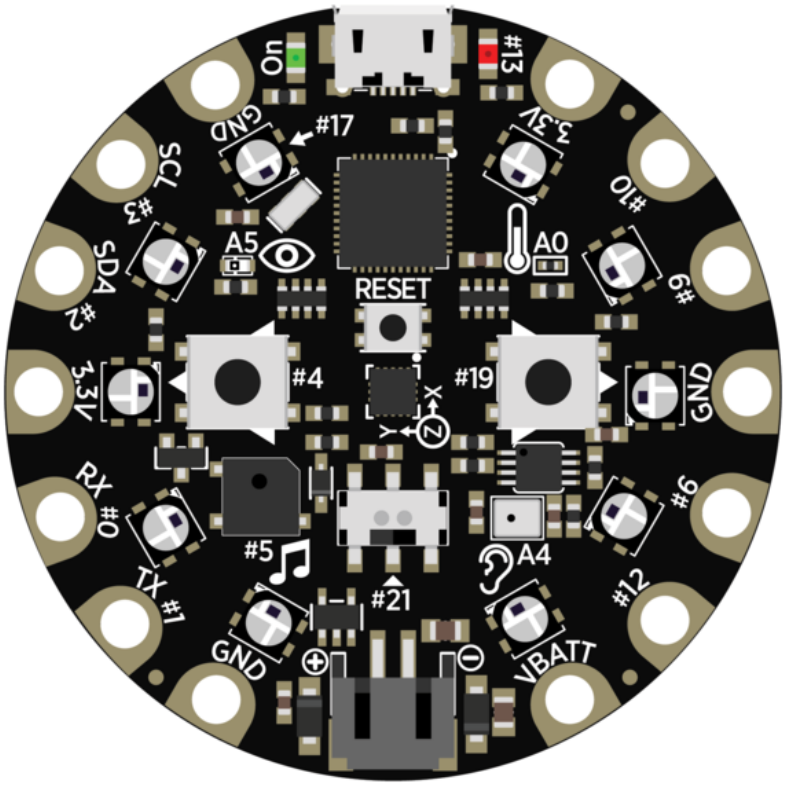
We have a rich history of light painting projects here at Adafruit, as well. Check out some of these excellent projects (https://adafru.it/wOf) for incredible effects that involve persistence of vision images painted with light, and even thermal heat map painting (https://adafru.it/sje).



A Circuit Playground and a camera or smartphone capable of taking long exposure photos is enough to get started. By using the on-board accelerometer, you'll be able to change colors by rotating the Circuit Playground. As an added touch, you can use a potentiometer to easily adjust the brightness of the NeoPixels.

Code the Light Paintbrush

Before you get started, make sure you've been through the basic tutorials on using the Circuit Playground. You'll need to have installed the Arduino IDE, added the Circuit Playground board to Arduino, and added the Circuit Playground library. This excellent guide, Introducing Circuit Playground (https://adafru.it/pAP) and Circuit Playground lesson #0 (https://adafru.it/Cgt) will show you how to do all of that and get you going in no time!



If you are using the Circuit Playground Express, be sure to check out the next page of the guide to see how to program the Light Paintbrush using MakeCode

Plug your Circuit Playground into your computer now using a USB cable, and launch the Arduino IDE. Double check that you've selected the board and port as in the Lesson #0 tutorial and are ready to upload code.

The code for the Light Paintbrush has two jobs:

- Read the on-board accelerometer to determine which color to set the NeoPixels
- Read the potentiometer connected to pin #10 to set the NeoPixel brightness

```
// ----------------------------------------------------------------------------
//Circuit Playground Light Paintbrush
// John Park
// for Adafruit Industries
//
// Use the Circuit Playground as a long exposure photography light paintbrush
// tilt to change color
// adjust brightness with the 10k potentiometer
//
 /*      +X
 *        __
 *       /  \
 * +Y   |    |   -Y
 *       \__/
 *
```

```
 *       -X
 */
//
// MIT license, check LICENSE for more information
//
//
// ----------------------------------------------------------------------

#include &lt;Adafruit_CircuitPlayground.h&gt;
#include &lt;Wire.h&gt;
#include &lt;SPI.h&gt;

int sensorPin = A10;
int sensorValue = 0;
int brightVal = (20);

int X;
int Y;

int red = 0;
int green = 255;
int blue = 0;

void setup() {
  CircuitPlayground.begin();
  CircuitPlayground.clearPixels();
  CircuitPlayground.setBrightness(brightVal);
}

void loop() {
  sensorValue = analogRead(sensorPin);
  brightVal = map(sensorValue, 0, 1023, 0, 225);

  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();

  if ( (X&gt;=1) &amp;&amp; (Y&lt;1) ) {      //+x, -y = GREEN
    red = 0 ;
    green = 255;
    blue = 0;
  }
  else if ( (X &lt;1 ) &amp;&amp; (Y&lt;1) ){ //-x, -y = BLUE
    red = 0;
    green = 0;
    blue = 255;
  }
  else if ( (X &gt;=1) &amp;&amp; (Y&gt;=1) ){ //+x, +y = RED
    red = 255;
    green = 0;
    blue = 0;
  }
  else if ( (X &lt;1) &amp;&amp; (Y&gt;=1) ){  //-x, +y = YELLOW
    red = 255;
    green = 255;
    blue = 0;
  }

  for (int i=0; i&lt;10; ++i) {
    CircuitPlayground.setBrightness(brightVal);
    CircuitPlayground.strip.setPixelColor(i, red, green, blue);
  }
  CircuitPlayground.strip.show();
}
```

Download the Code

You can press the button below to download the code. Unzip and move the directory CircuitPlaygroundLightPaintbrush to your Arduino sketch directory, then open it in the Arduino IDE and upload it to your Circuit Playground.

CircuitPlaygroundLightPaintbrush.zip

https://adafru.it/sjf

Next you'll assemble the Light Paintbrush.

MakeCode for Circuit Playground Express

MakeCode only works with the Circuit Playground Express. If you have the original Circuit Playground Classic, check out the Arduino version of this code!

Lots of Coding Options

There are many options for programming the Circuit Playground Express to work as a Light Paintbrush! You can hand-code it with the traditional, text-based Arduino IDE, as seen in the previous page of this guid, you can try your hand at the exciting new CircuitPython methods, or you can harness the incredibly user-friendly, yet powerful block-based interface of Microsoft's MakeCode web app (https://adafru.it/wmd)!! (You can even go full circle and hand type code inside the MakeCode interface with Javascript!)

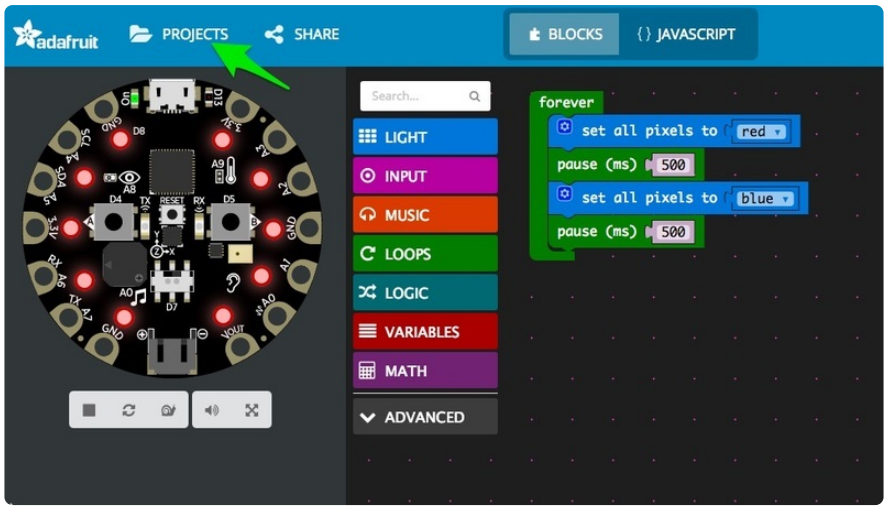
By simply dragging and dropping elemental code blocks, you can create Circuit Playground Express software in no time, and even test it out with the virtual Circuit Playground Express simulator built into the page.

Be sure you've gone through the tutorial here on the basics of MakeCode (https://adafru.it/wB5) for Circuit Playground Express.

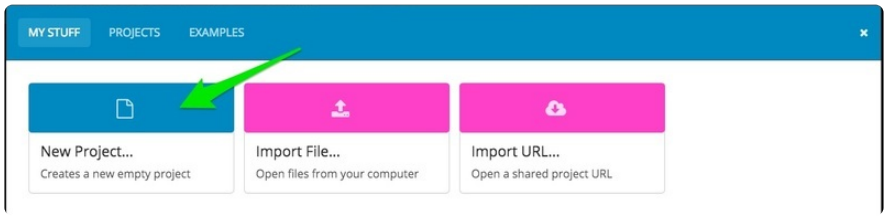
MakeCode

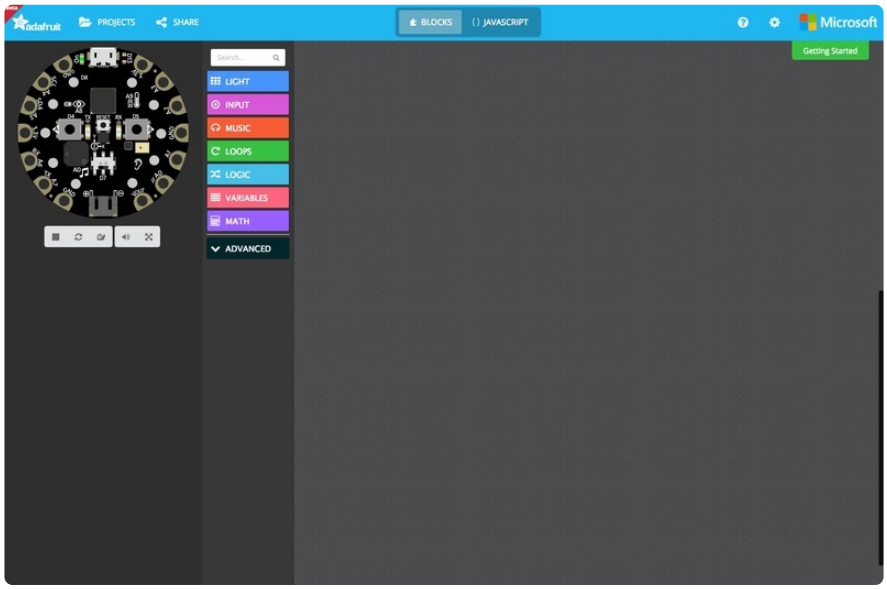
You're now ready to begin coding your own program with the MakeCode interface to use the Circuit Playground Express as your tilt-reactive Light Paintbrush.

Click the **Projects** button so you can start a new project.



Click the **New Project...** button to create the new empty project.



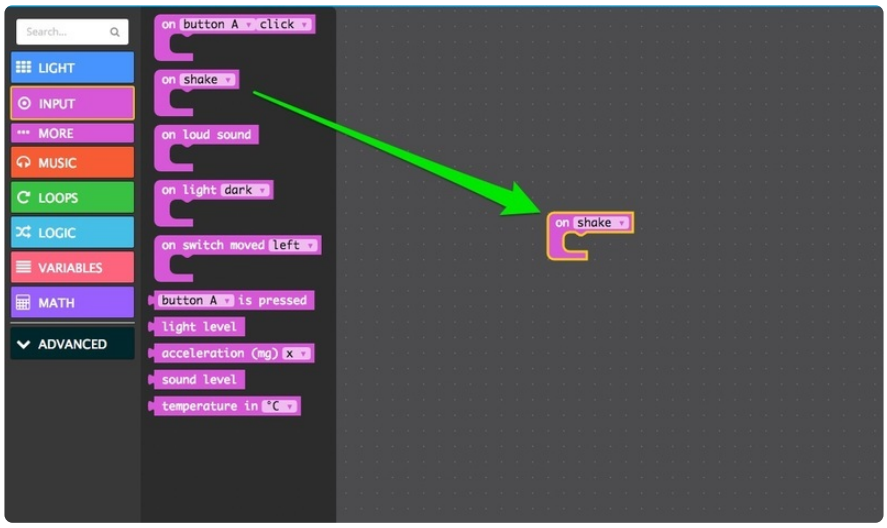


Input

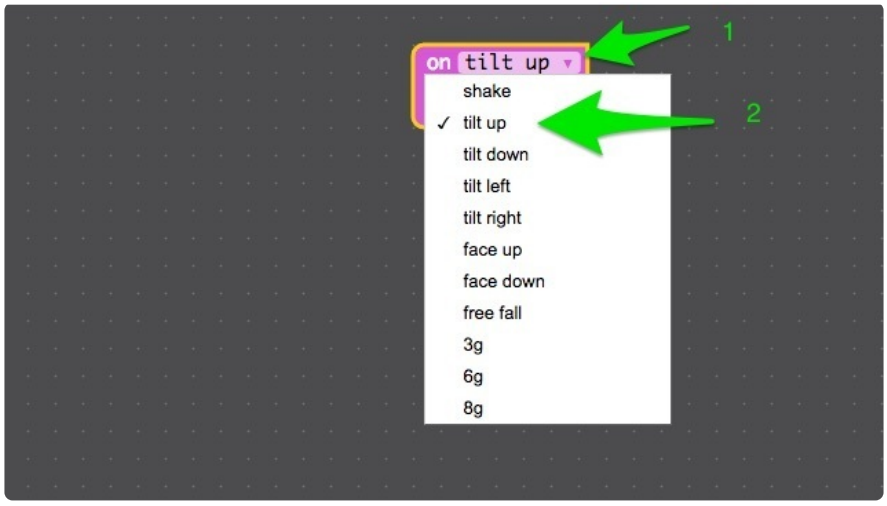
Your Light Paintbrush will display different colors depending on the angle to which you rotate it. To do this, you'll create inputs that check the rotation angle, and then set the entire NeoPixel ring to a particular color.

The first block you'll add is is an input block that can read the tilt orientation of the accelerometer on the Circuit Playground Express. Click on the **Input** category and

choose the **on shake** block, then drag it to the canvas. (All of the accelerometer-based inputs contained within this one block.)

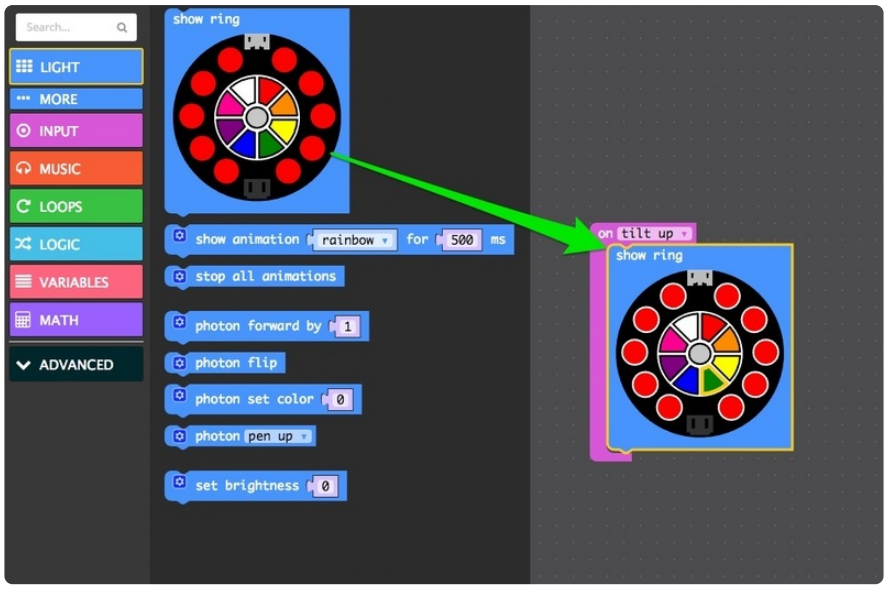


Click on the **shake** dropdown, and select the **tilt up** options from the list.

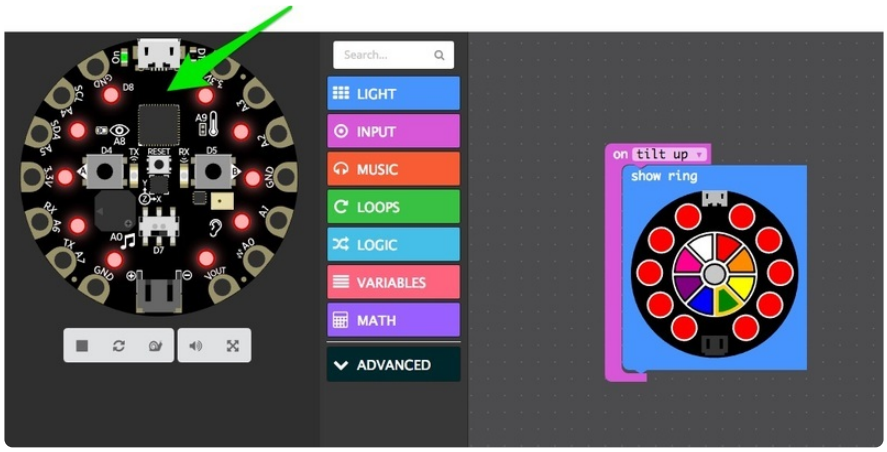


Colored LEDs

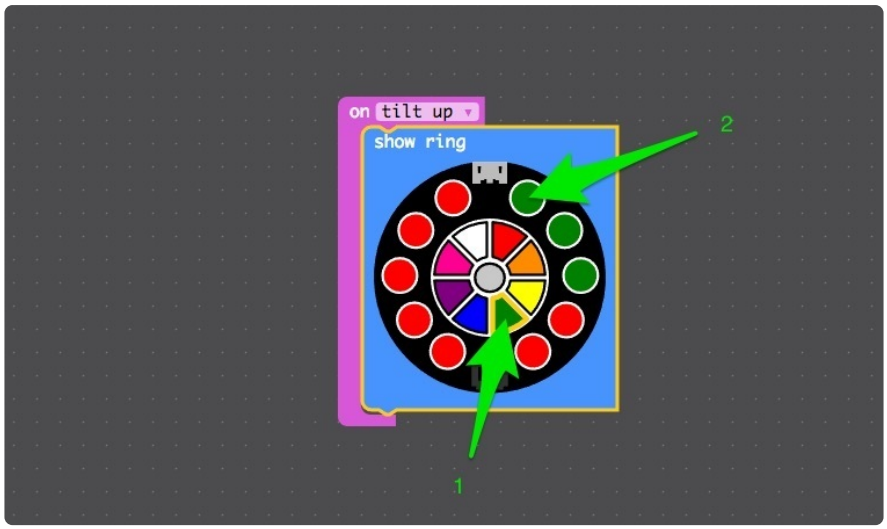
To light the LEDs, you'll add a **show ring** block from the **Light** category, and then drag it inside the **on tilt up** block.



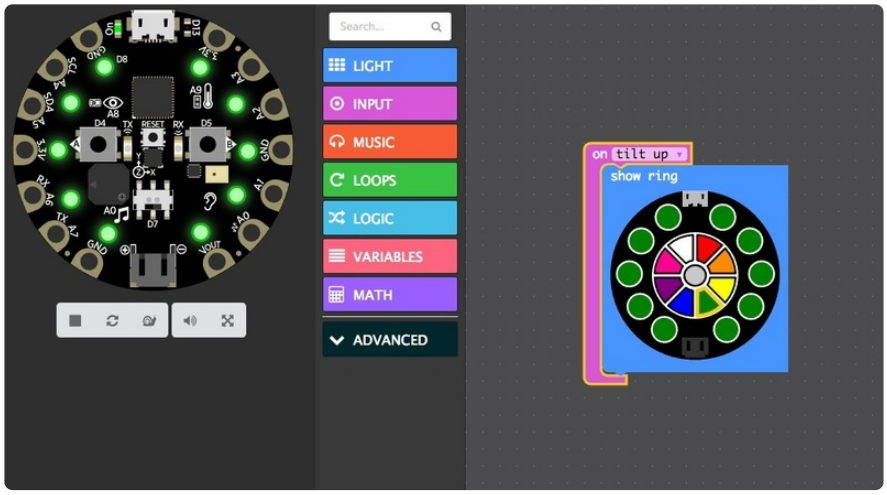
You can test this in the simulator by hovering your mouse cursor over the top of the virtual Circuit Playground Express. You'll see the board tilt a bit and the LEDs will turn red. The simulator is a great way to iterate on your program without needing to upload to the real board every time.



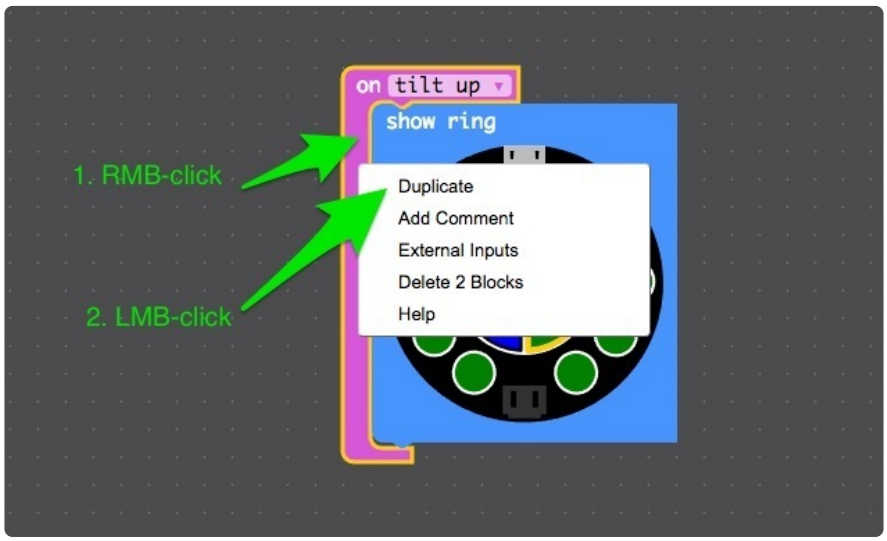
The default **show ring** color is red, but we'll go ahead and change the **on tilt up** color to green. You can do so by first clicking the green "pie wedge" color picker, and then clicking each of the circular LED dots.



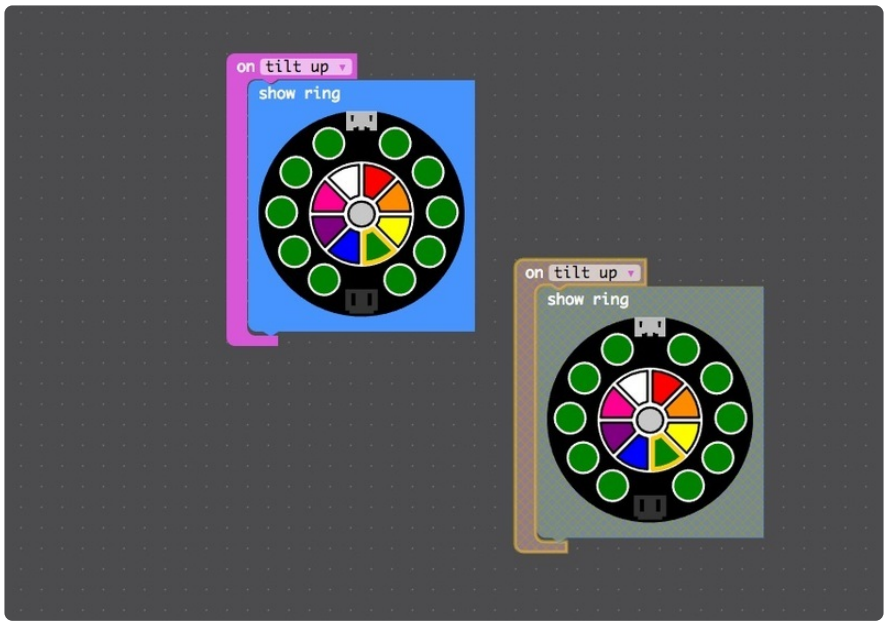
Now when you hover your cursor over the top of the board in the simulator, the LEDs will turn green. Success!



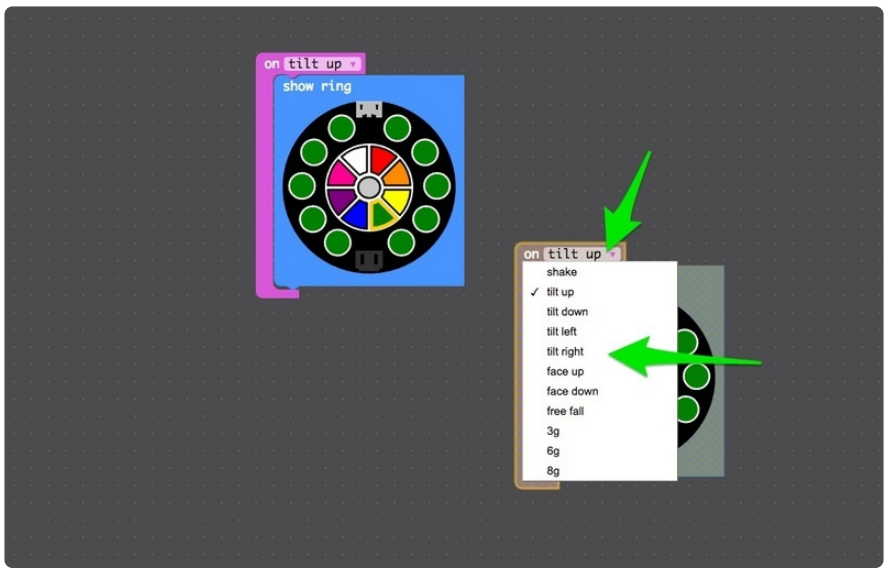
To create the other three tilt direction/color block sets, right-mouse-button (RMB) click the **on tilt up** block and then left-mouse-button (LMB) click the **Duplicate** command from the pop-up list.



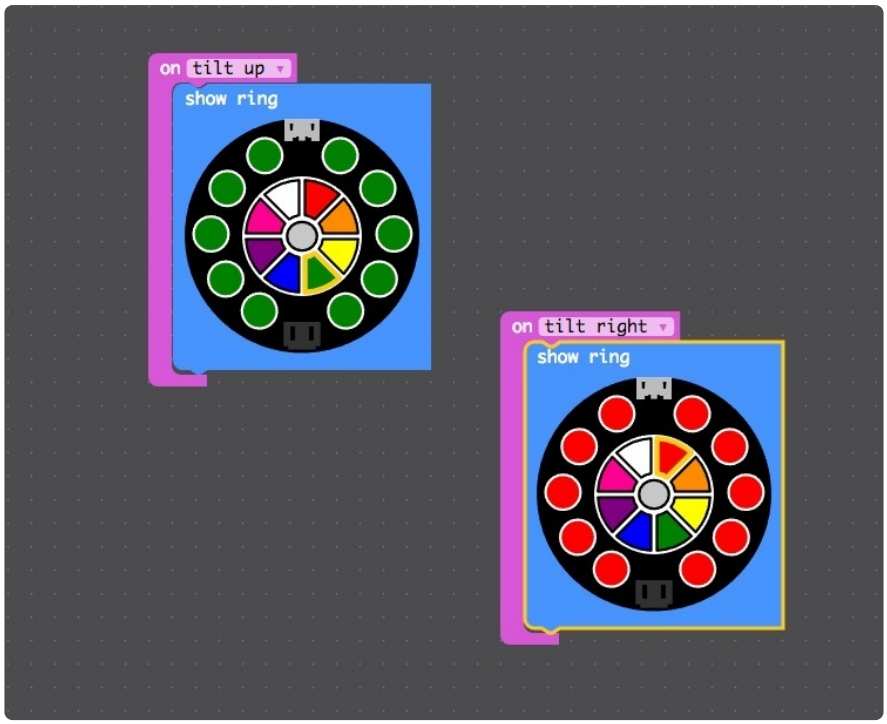
You'll notice that the new copy of the block is greyed out, meaning it will not run as part of your program. This is because only one instance of the same input can be used -- otherwise the program would get horribly confused in dealing with conflicts.



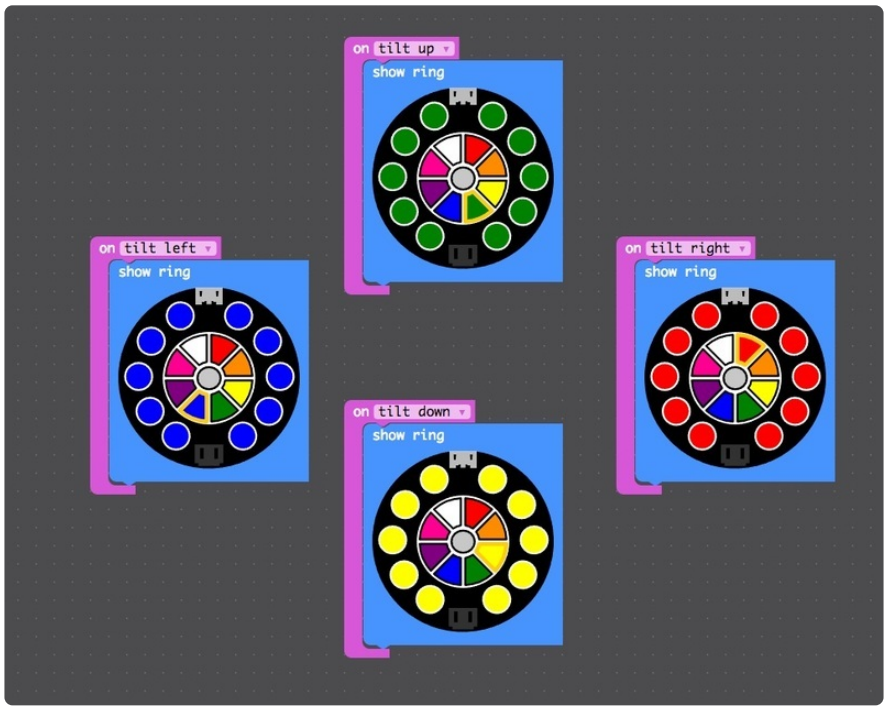
Click on the **tilt up** option on the second block set and choose **tilt right.**



Using the same method as before, change the LED colors to red on the **on tilt right** block set.



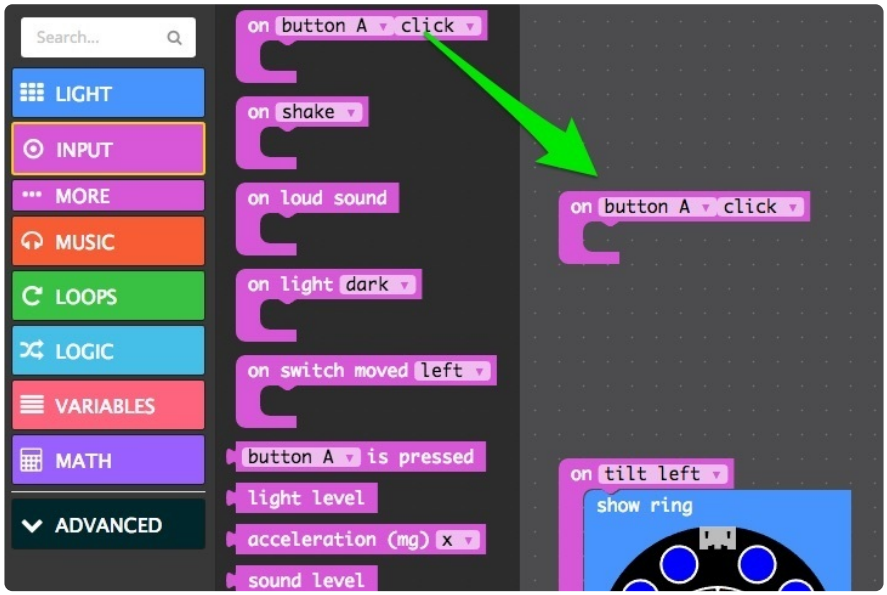
And now, complete the set of gestures by duplicating the block set two more times, and adjusting them as seen here.



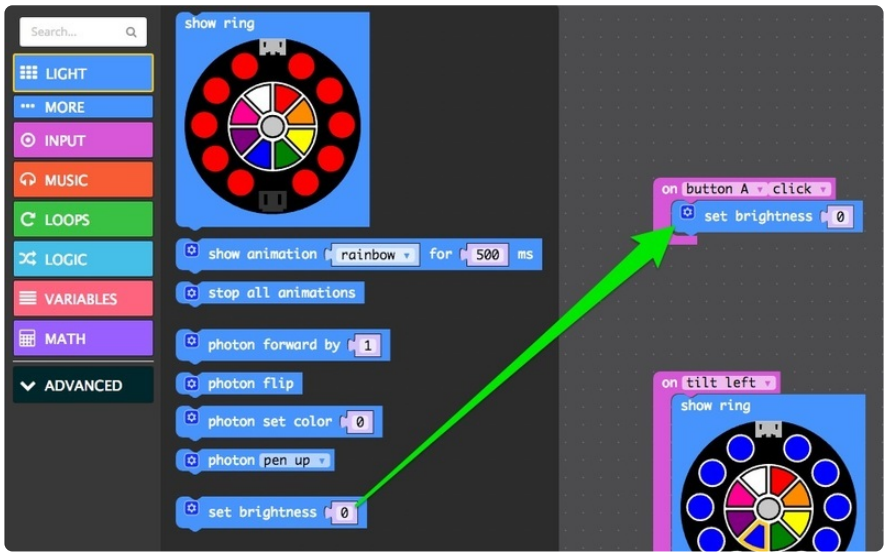
That's all there is to creating the core functions of you motion reactive, color changing Light Paintbrush! Test it out on the simulator, and then we'll add one more feature -- a set of hight/low brightness controls.

Brightness Control

To adjust brightness between high and low, you'll click the A and B buttons. Go to the **Input** block and add an **on button A click** block.

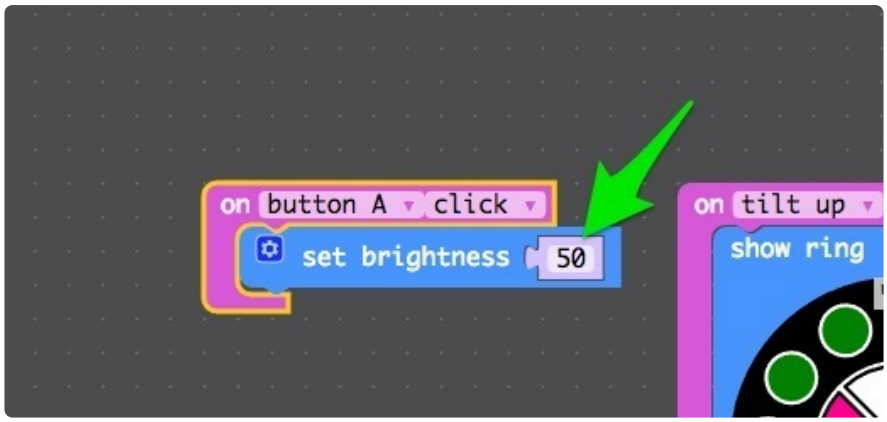


Click on the **Light** category and add a **set brightness 0** block inside the **on button A click** block.

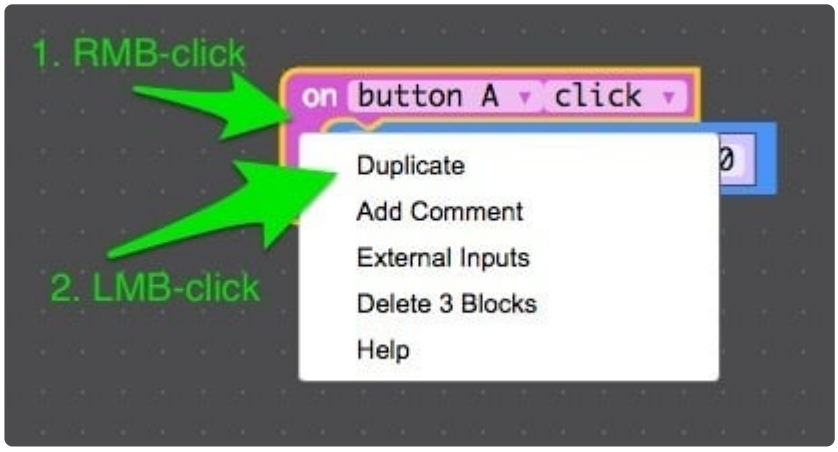


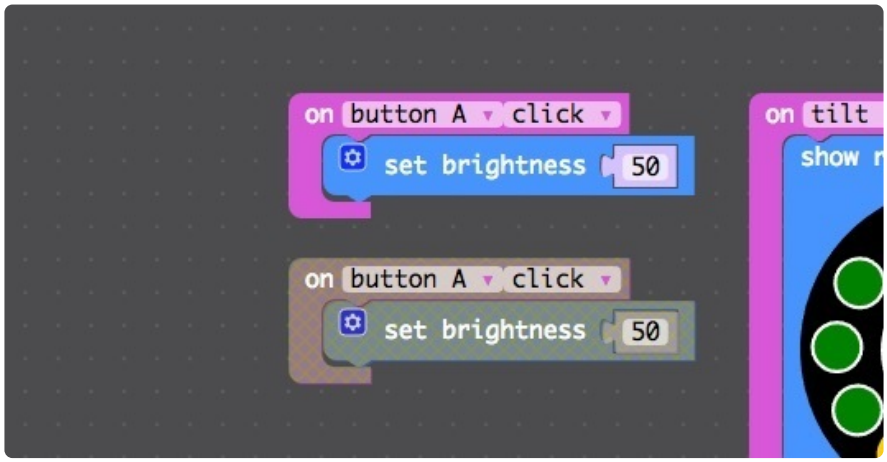
The "set brightness" command sets the brightness universally for all of the NeoPixel LEDs

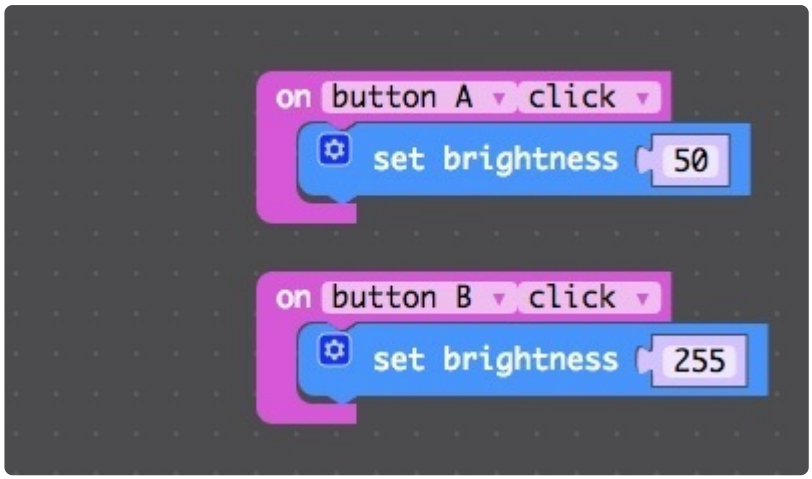
Change the "low" brightness setting to 50. You can adjust this value later as you see fit.



Duplicate the **on button A click** block, and then switch the input on the new copy to **button B,** and set the brightness value to 255 (which is blindingly bright, so be careful!)

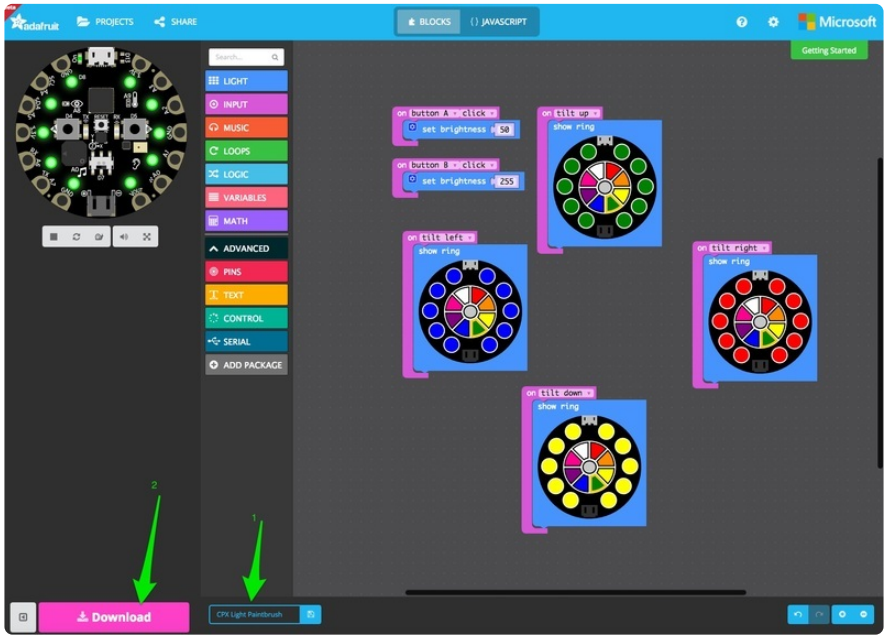




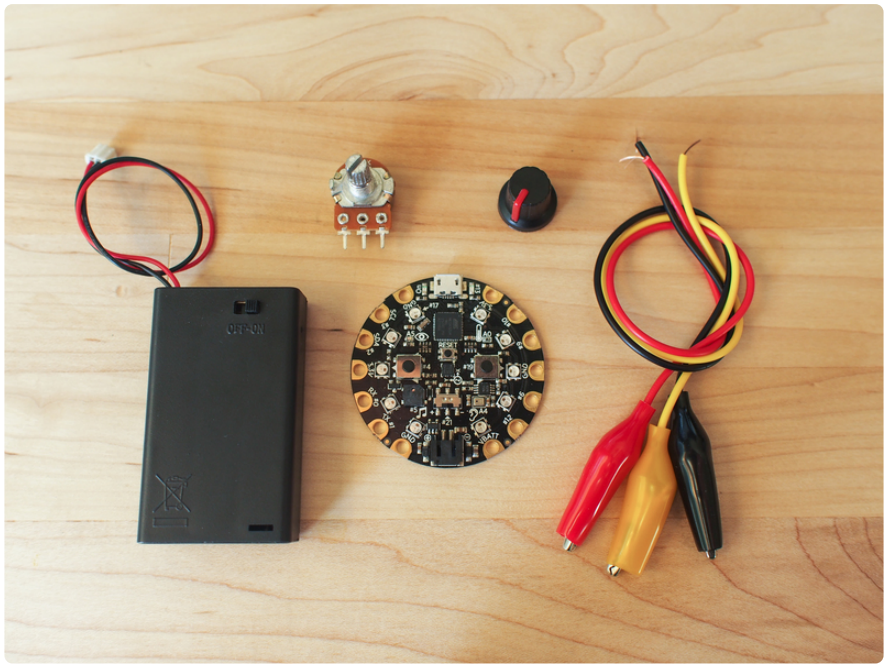


You won't see the new brightness value take effect until the next time the "show ring" is called by tilting the Light Paintbrush

Your program is complete! You can now dowload the .uf2 file, and upload it to your Circuit Playground Express by plugging it into your computer over USB, and pressing the reset button on the Circuit Playground Express to place it into bootloader mode, and then dragging the .uf2 file onto the CPLAYBOOT drive on your computer.



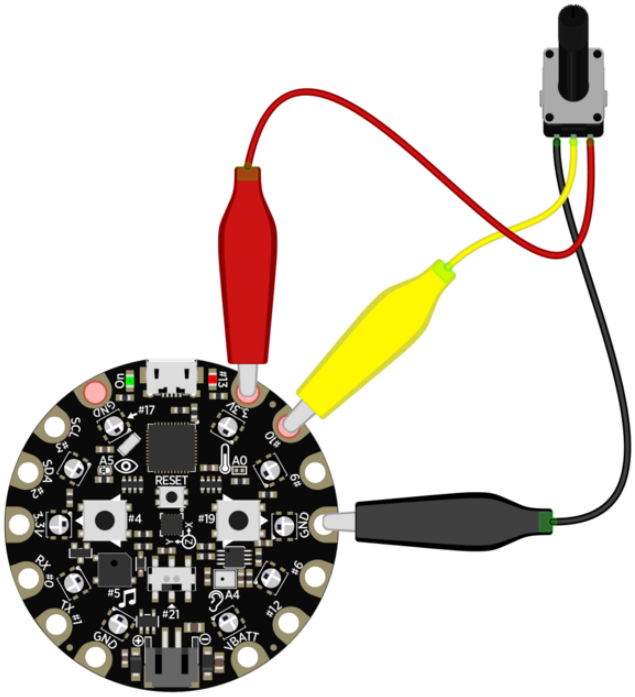
Assemble the Light Paintbrush



With the Arduino software loaded onto the Circuit Playground Classic, your Light Paintbrush will need power and one additional sensor -- a potentiometer -- to control brightness. You'll use the battery holder as both power supply and handle for your Light Paintbrush. This is expedient, inexpensive, and a great example of form following function. You could build a nice enclosure for the project if you like, but the Light Paintbrush works well without one.

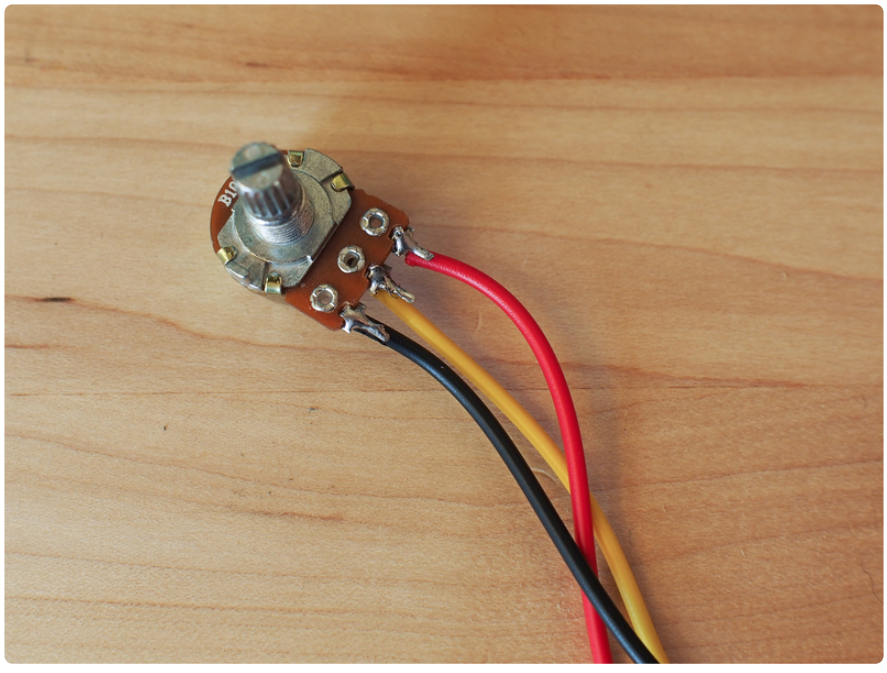
For a very quick, simplified version of this project you can skip adding the brightness controlling potentiometer, just remember to comment out that section of the Arduino sketch to avoid random dimming!

If you made the Circuit Playground Express version of the Light Paintbrush with MakeCode, you will use the buttons for brightness instead of the added potentiometer.



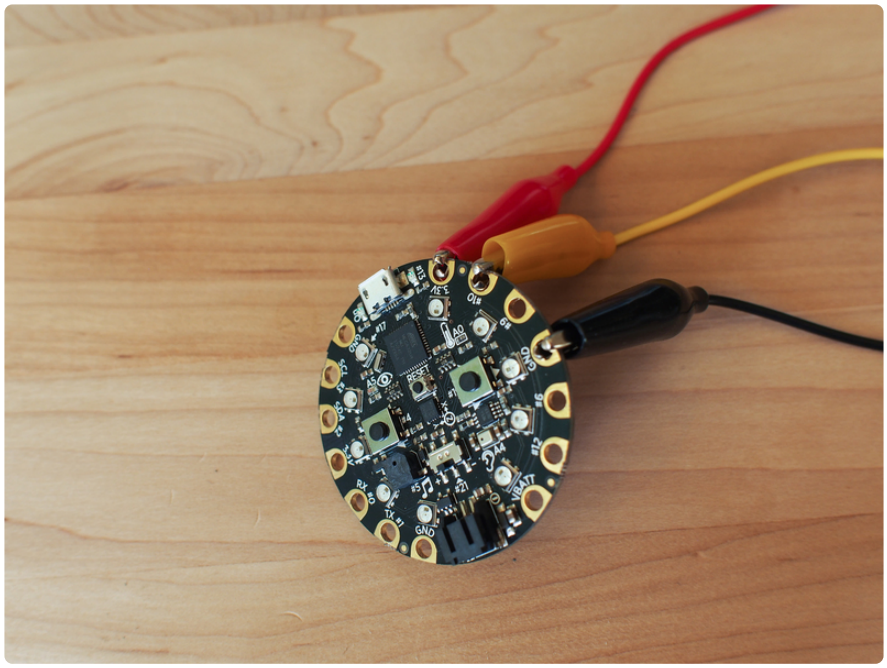
The first build step is to attach leads to the potentiometer. The potentiometer uses three connections to the Circuit Playground Express -- ground, signal, and voltage. The ground and voltage are references for the varying resistance of the middle signal leg as the knob is turned. The Circuit Playground reads these changes as analog values from 0 to 1023, which are then remapped to NeoPixel brightness values in the software.

Cut one clip end off of each alligator clip lead, and strip away a bit of insulation to expose the wire. Tin the potentiometer legs with solder, tin the wires, and then solder the wires to the legs in the order shown here.

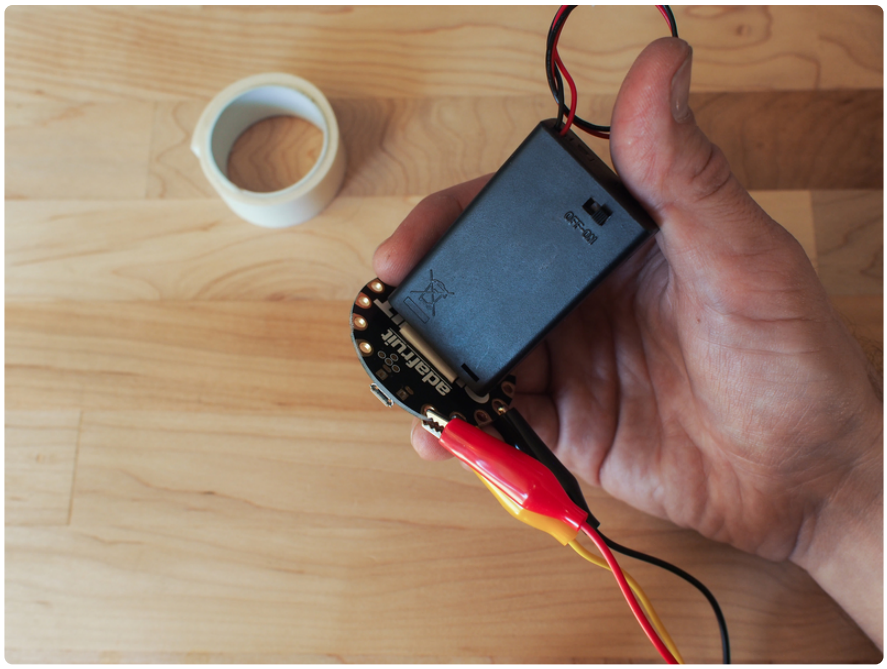


Then, clip the leads to the pads on the Circuit Playground Express:

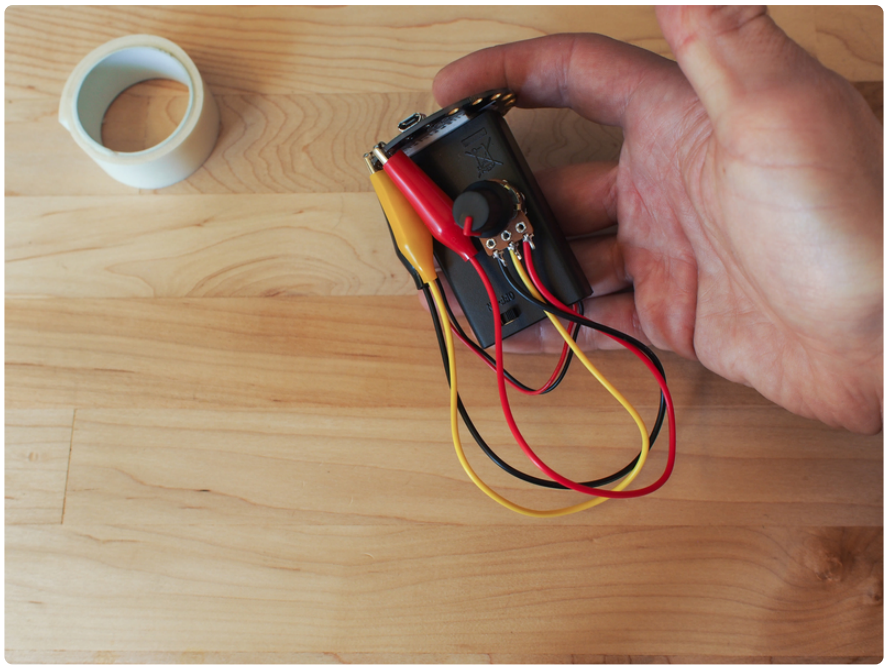
- black to GND
- yellow to #10
- red to 3.3V

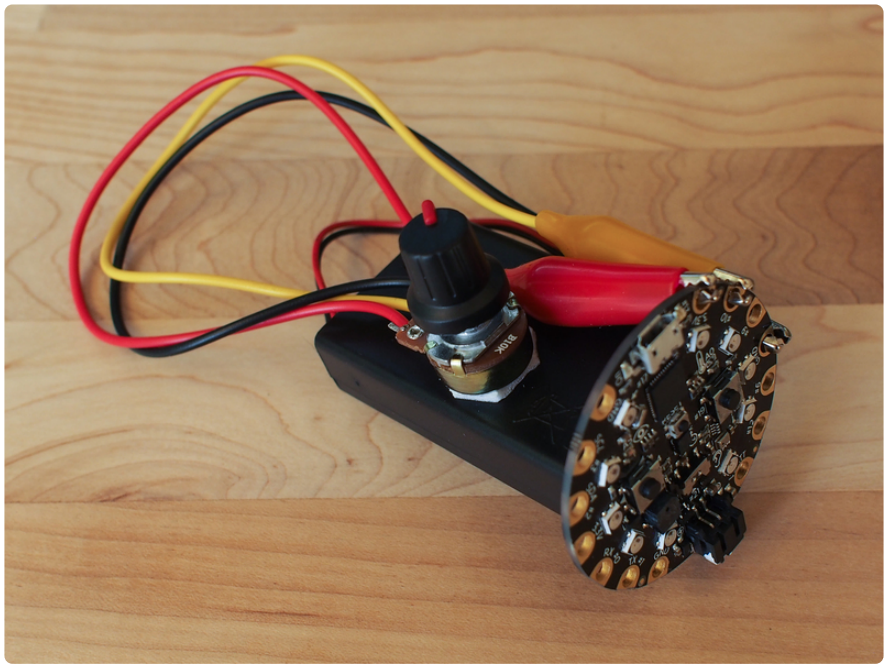


Next, use a small piece of foam double-stick tape to connect the battery case to the back of the Circuit Playground.



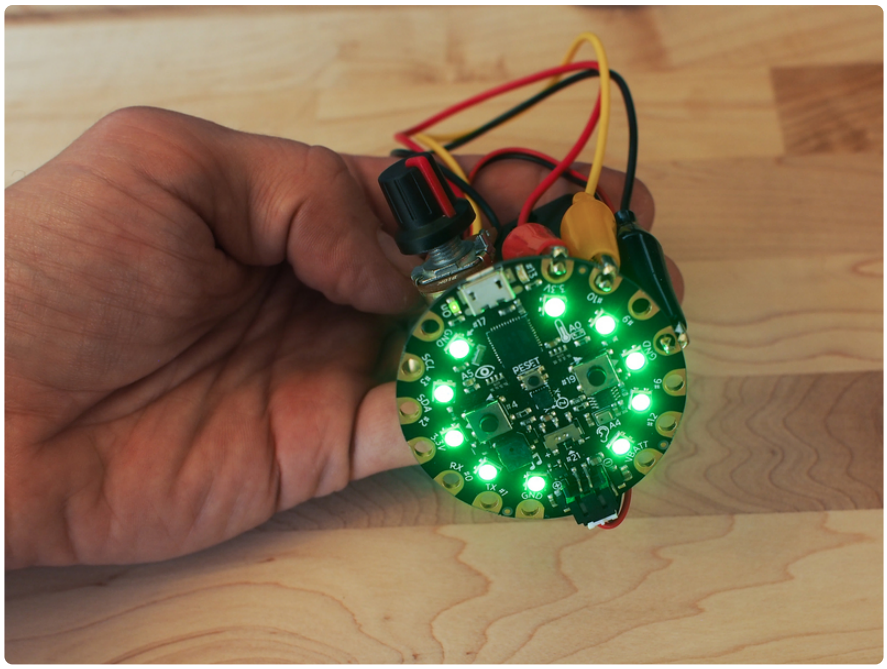
Finally, use some foam tape to attach the potentiometer to the battery case.

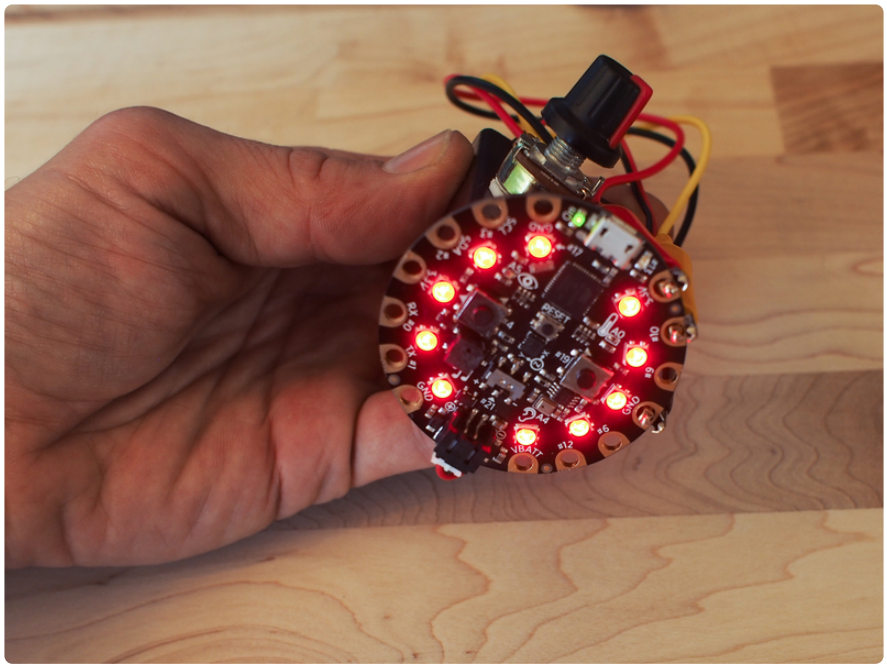


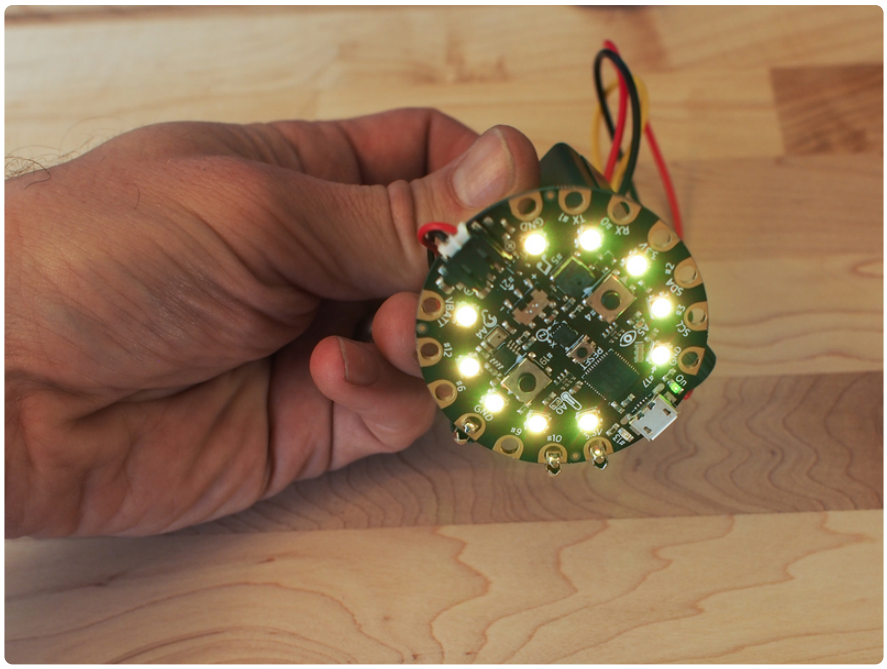


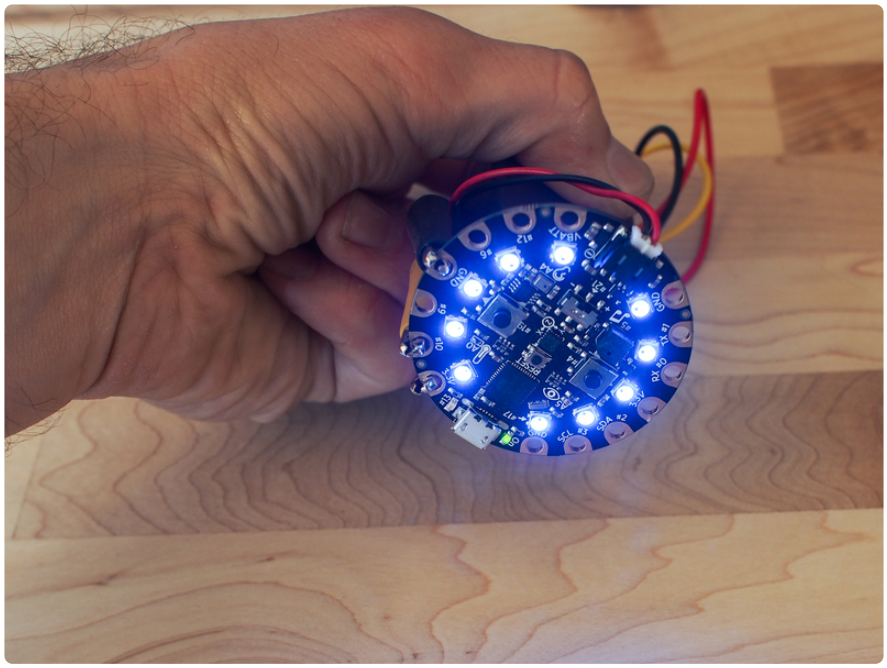
That's it, your Light Paintbrush is complete!

Turn it on and try tilting the brush to change colors, and adjust brightness with the knob.









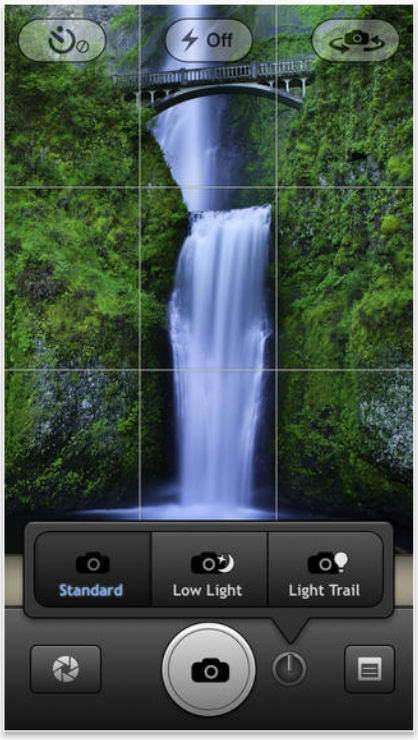
Now comes the artistic part -- time to make some light paintings!

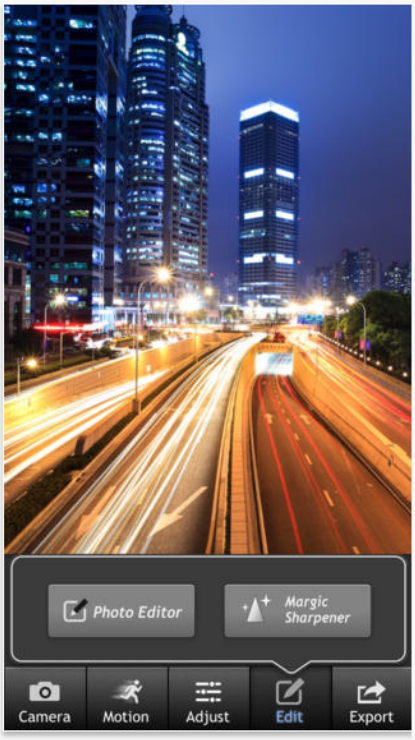
Paint with Light

Now that you've built it , in order to paint with your Light Paintbrush you'll need a dark place and a camera or smartphone with "bulb" or long exposure mode.

On a camera, look for a "B" on the dial or check the far end of the shutter values. Bulb may require a held shutter, so it can be convenient to use a shutter in the 10-to-30 second range (notated as 10", 20", 30" and so on), or a remote trigger.

For smartphones there are many apps available -- just search for "long exposure" or "light trails" in the app store of your choice.

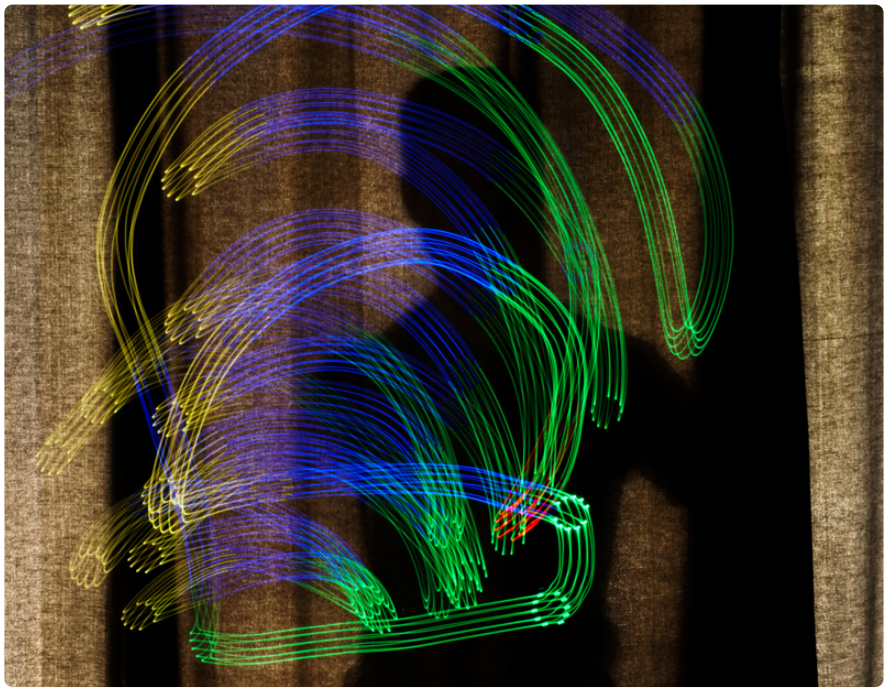


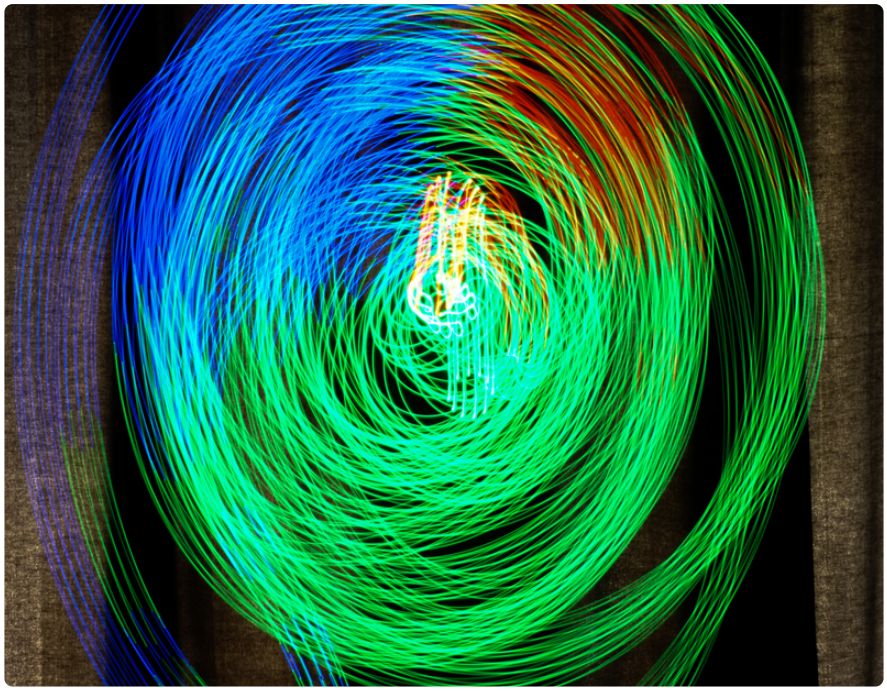


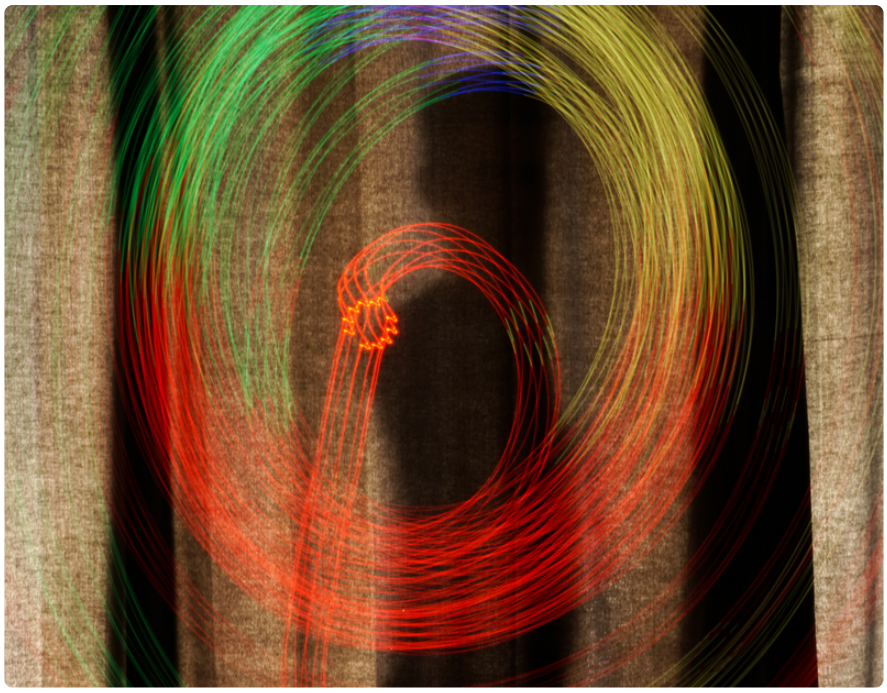
Now, head to a darkened room or outdoors at night. Set your camera or smartphone on a table or tripod, put it in long exposure mode with a small aperture to keep the scene fairly dark, and press the shutter.

Turn on your Circuit Playground Light Paintbrush, set the brightness knob to a medium setting, and move the Light Paintbrush in some circles and arcs. When you're done, press the camera shutter again and check your photo.

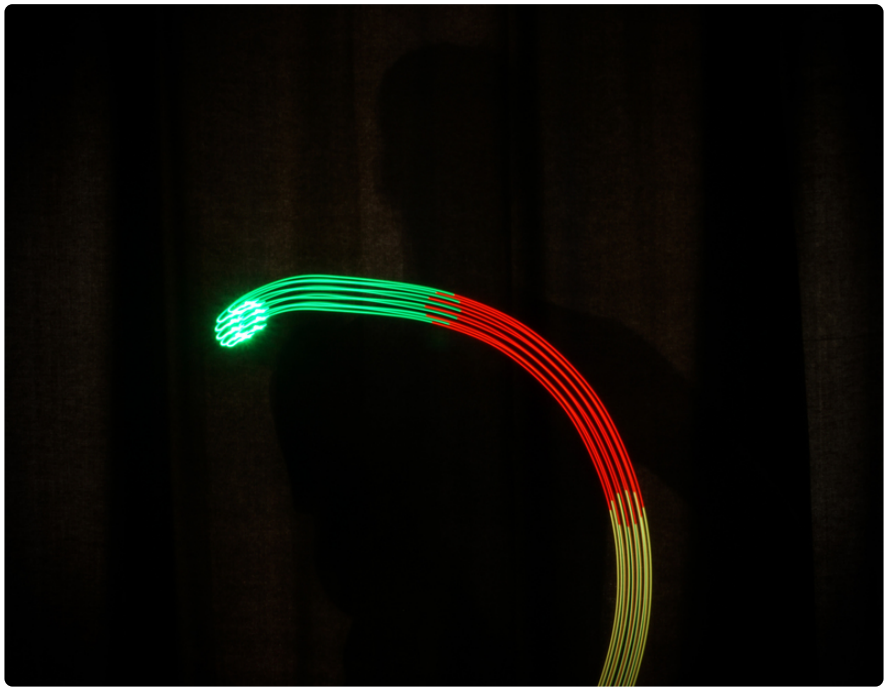
It may take some tuning to get the brightness and focus just right -- it's very helpful to have an assistant -- but once you do you'll see beautiful streaks of light with colors changing depending on the rotation arc of your arm or how you twist your wrist.

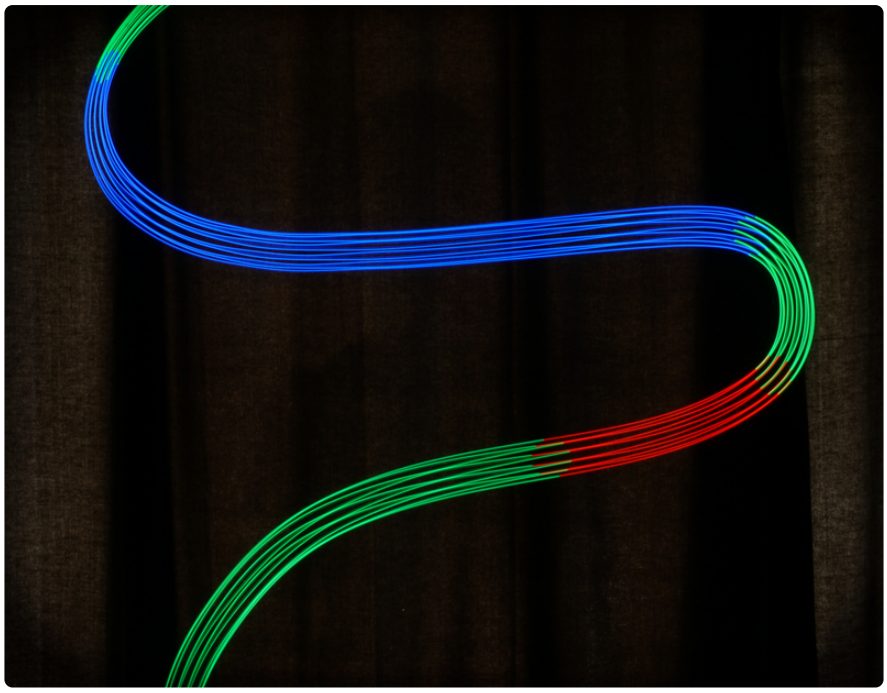




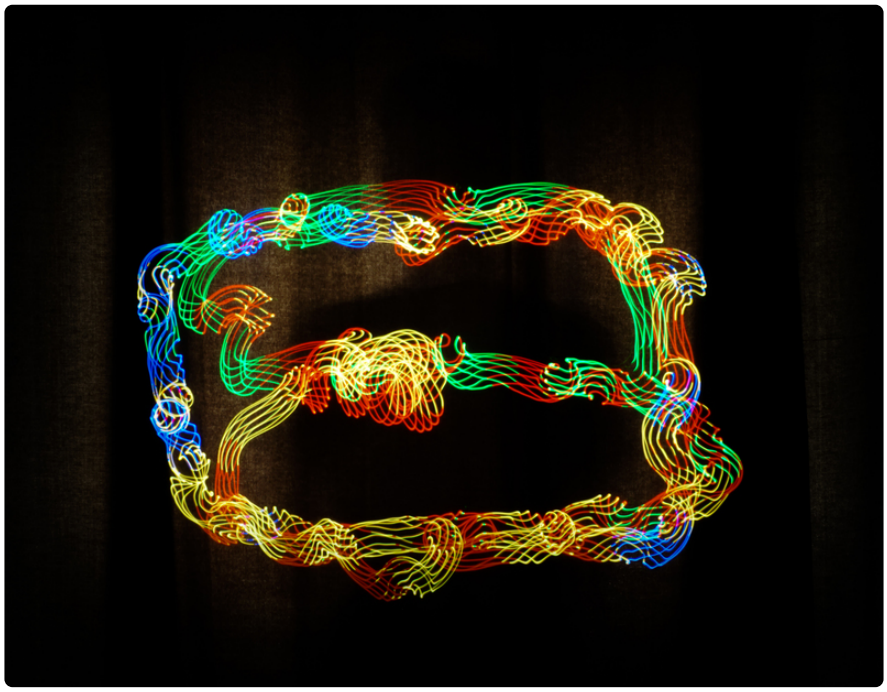


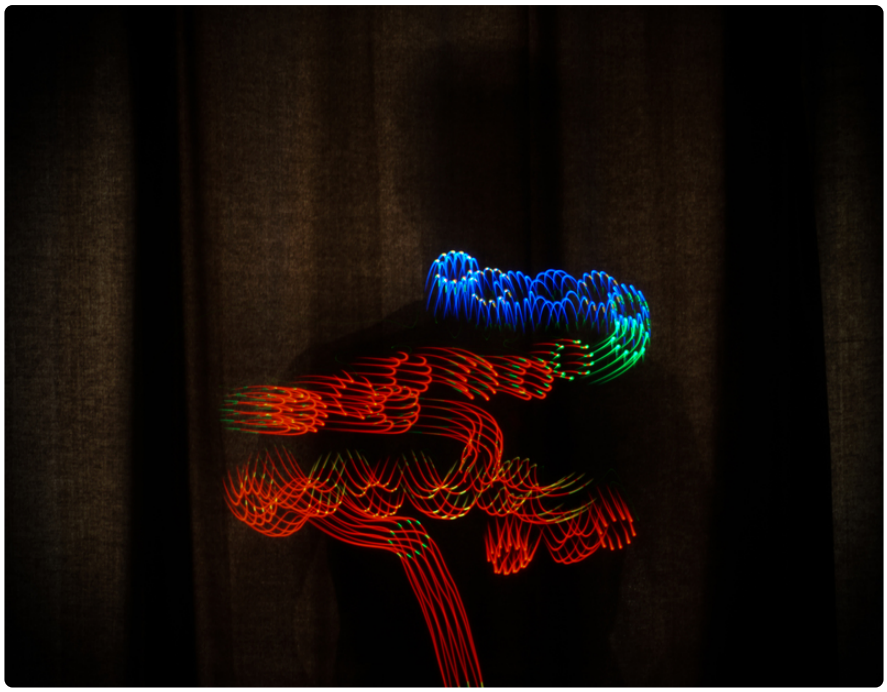
This image was a single streak during a five second exposure.



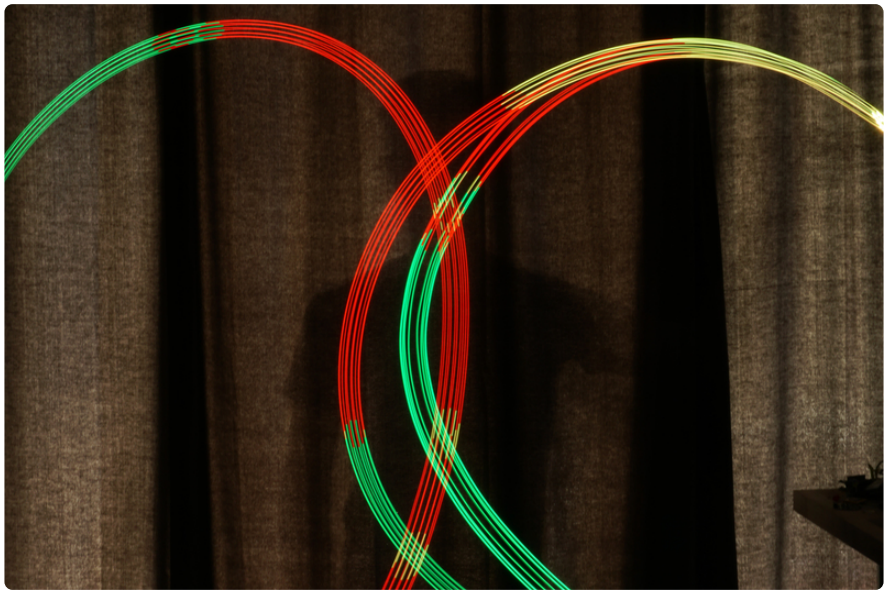


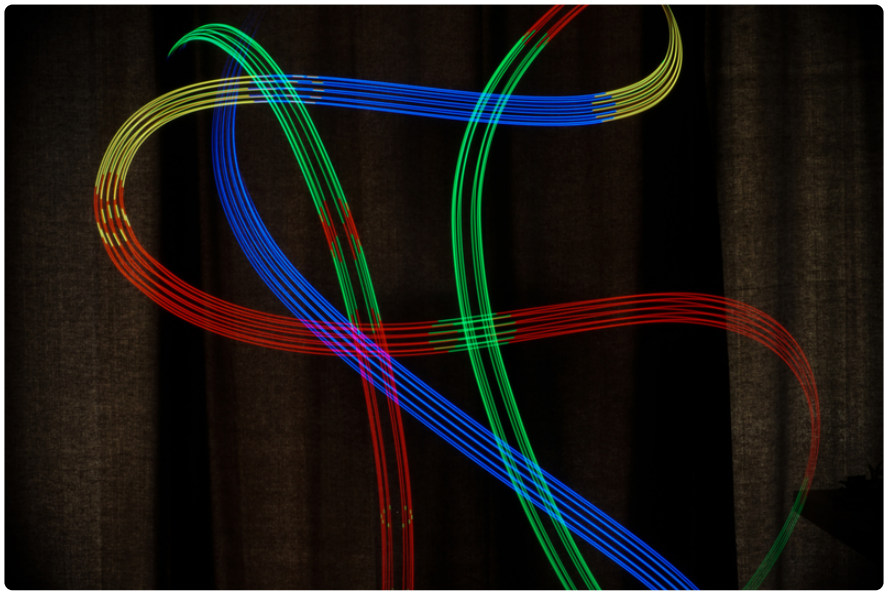
For these jellyfish-like images, the brightness knob was twisted on and off while moving and twisting the brush.

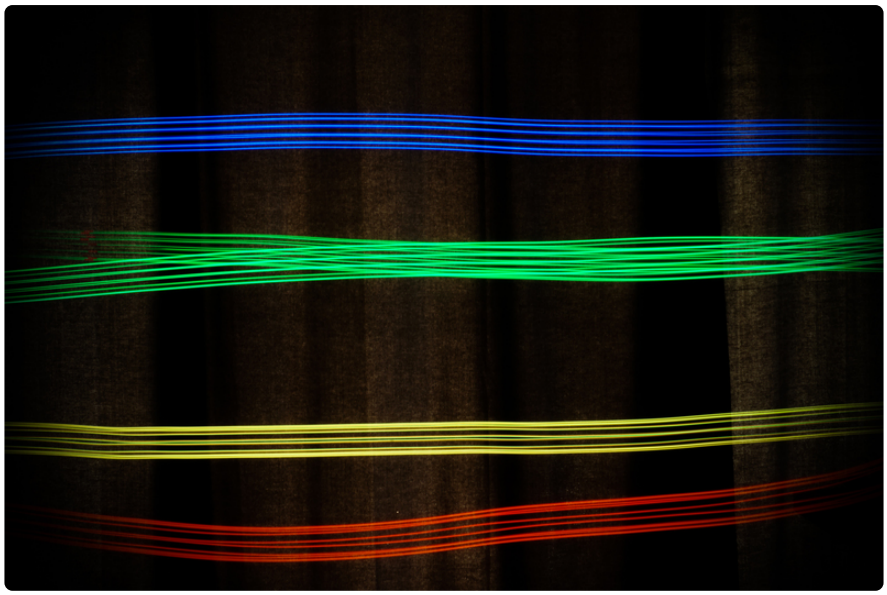


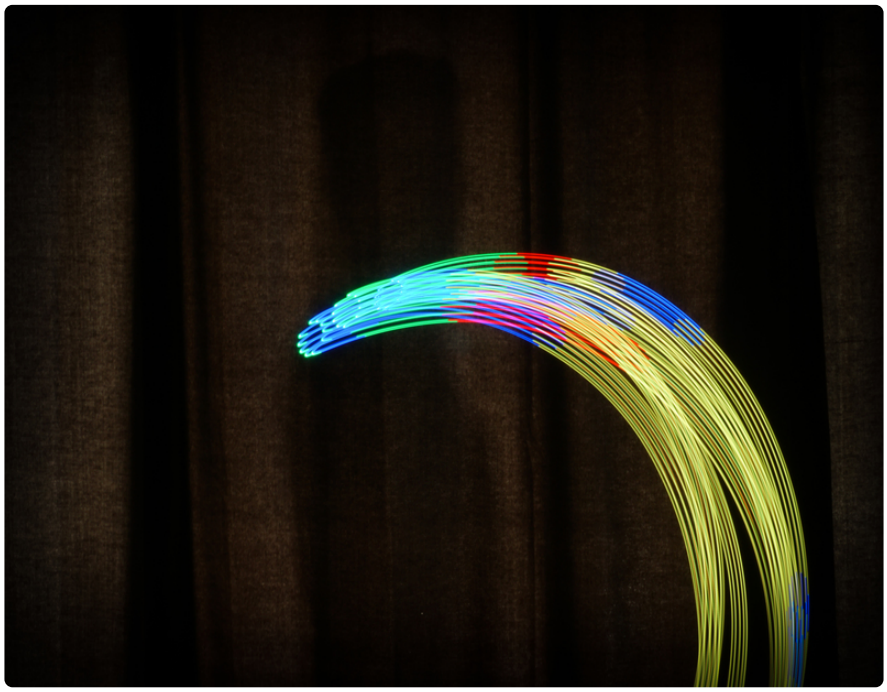


Less can be more.

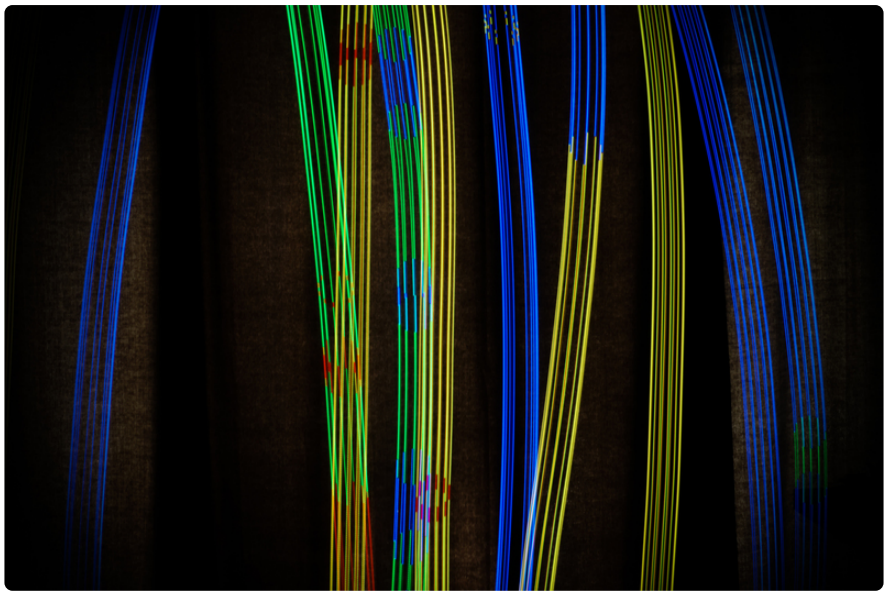




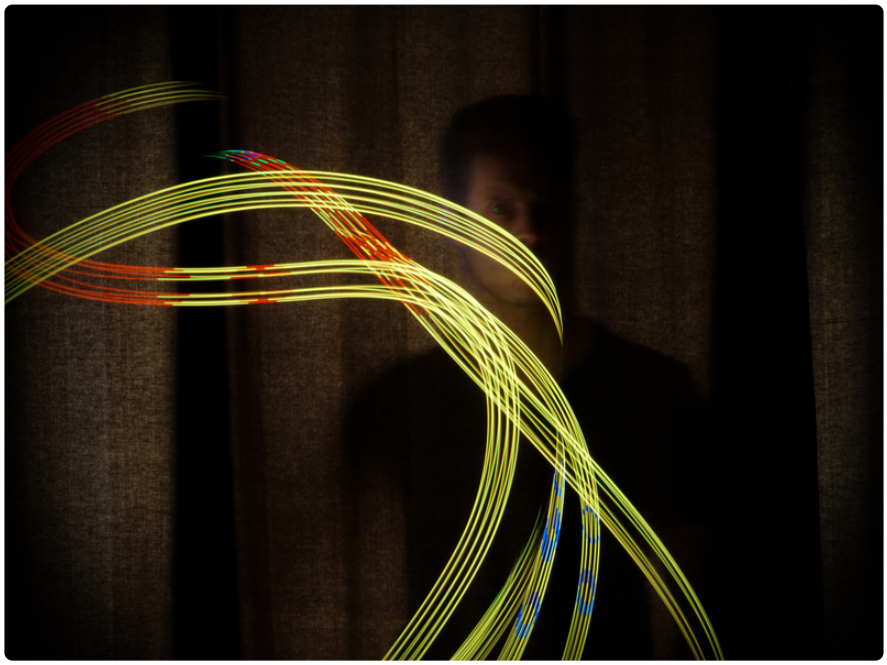




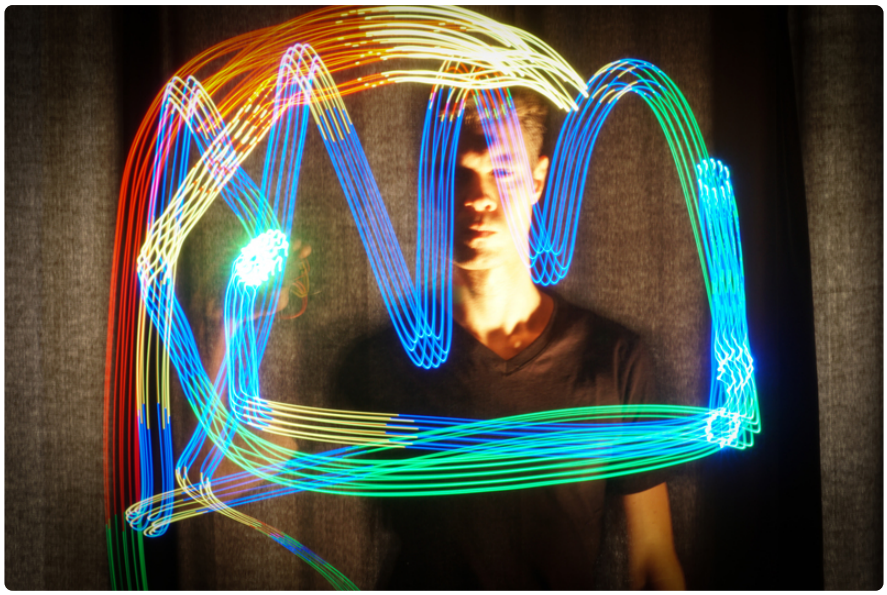
More can also be more.



Here I illuminated my face briefly as well by pointing the Light Paintbrush at myself.



An external flash (or bright light turned on and off briefly) can help add your subject to the photograph without blurring.



Enjoy creating art with your Circuit Playground Light Paintbrush!