



Bluefruit Luminary Lanterns with Capacitive Touch

Created by Erin St Blaine



<https://learn.adafruit.com/bluefruit-luminary-lanterns-with-capacitive-touch>

Last updated on 2024-06-03 02:58:04 PM EDT

Table of Contents

Overview	3
<ul style="list-style-type: none">• Parts• Materials & Tools	
Wiring Diagram	6
Software	7
<ul style="list-style-type: none">• Install Libraries	
Electronics Assembly	14
Design the Luminary	18
<ul style="list-style-type: none">• Materials• Design	
Cutting & Assembly	22
Use It	28
<ul style="list-style-type: none">• Adafruit Bluefruit App	

Overview

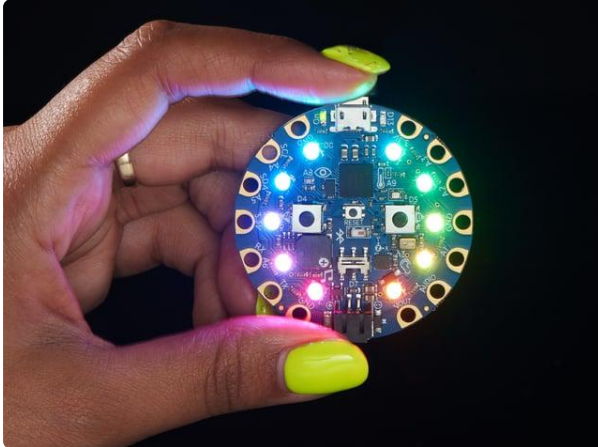
Design beautiful customizable luminary lanterns for your home or garden. Cut them out on a vinyl cutter or laser cutter, and fill them with light using a Circuit Playground Bluefruit Express and Adafruit's NeoPixel rings. Add custom color animations and control the lanterns with Adafruit's BlueFruit app.

We've also added capacitive touch control, so you can change color modes just by touching the lantern. Circuit Playground Express has all this functionality already built-in, so it's easier than ever to create gorgeous controllable lights.

If you're making just one lantern, this project doesn't require any soldering at all! The Circuit Playground's onboard lights will do the work. If you want to make multiple synchronized lanterns, it's easy to solder NeoPixel rings to the Circuit Playground and string together as many lanterns as you'd like.



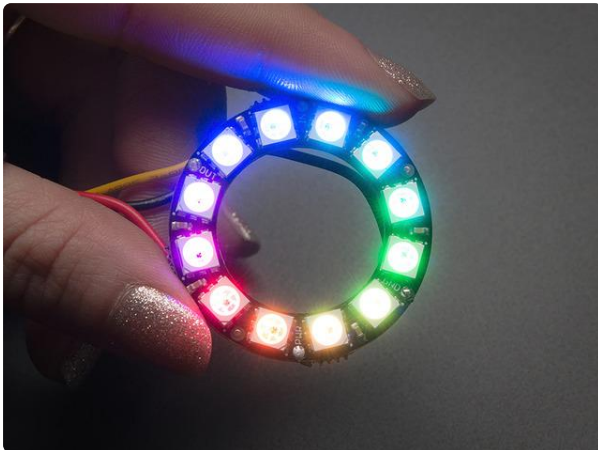
Parts



[Circuit Playground Bluefruit - Bluetooth Low Energy](https://www.adafruit.com/product/4333)

Circuit Playground Bluefruit is our third board in the Circuit Playground series, another step towards a perfect introduction to electronics and programming. We've...

<https://www.adafruit.com/product/4333>



[NeoPixel Ring - 12 x 5050 RGB LED with Integrated Drivers](https://www.adafruit.com/product/1643)

Round and round and round they go! 12 ultra bright smart LED NeoPixels are arranged in a circle with 1.5" (37mm) outer diameter. The rings are 'chainable' - connect the...

<https://www.adafruit.com/product/1643>



[5V 2A Switching Power Supply w/ USB-A Connector](https://www.adafruit.com/product/1994)

Our 5V 2A USB power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's power-hungry! This adapter was...

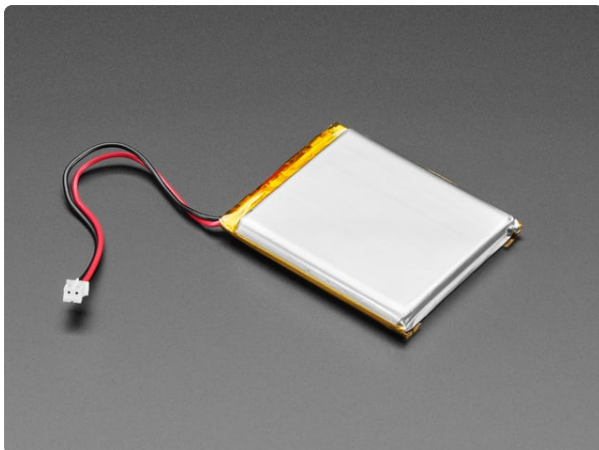
<https://www.adafruit.com/product/1994>



[5V 2A \(2000mA\) switching power supply - UL Listed](https://www.adafruit.com/product/276)

This is an FCC/CE certified and UL listed power supply. Need a lot of 5V power? This switching supply gives a clean regulated 5V output at up to 2000mA. 110 or 240 input, so it works...

<https://www.adafruit.com/product/276>



[Lithium Ion Polymer Battery - 3.7v 2500mAh](https://www.adafruit.com/product/328)

Lithium-ion polymer (also known as 'lipo' or 'lipoly') batteries are thin, light, and powerful. The output ranges from 4.2V when completely charged to 3.7V. This...

<https://www.adafruit.com/product/328>

[1 x 4 Conductor Silicone Wire](https://www.adafruit.com/product/3892)

Silicone Cover Stranded-Core Ribbon Cable - 4 Wires 1 Meter Long

<https://www.adafruit.com/product/3892>

[1 x Copper Foil Tape](https://www.adafruit.com/product/1127)

Conductive Adhesive - 25mm x 15 meter roll

<https://www.adafruit.com/product/1127>

[1 x 2 M USB Cable](https://www.adafruit.com/product/4148)

A to micro B, purple braided covering

<https://www.adafruit.com/product/4148>

[1 x DC Female Adapter](https://www.adafruit.com/product/3892)

Female DC Power adapter - 2.1mm jack to screw terminal block

[Slice%20Craft%20Knife%20with%20Ceramic%20Blade](https://www.adafruit.com/product/3892)

[1 x Craft Knife](https://www.adafruit.com/product/4306)

Slice Craft Knife with Ceramic Blade

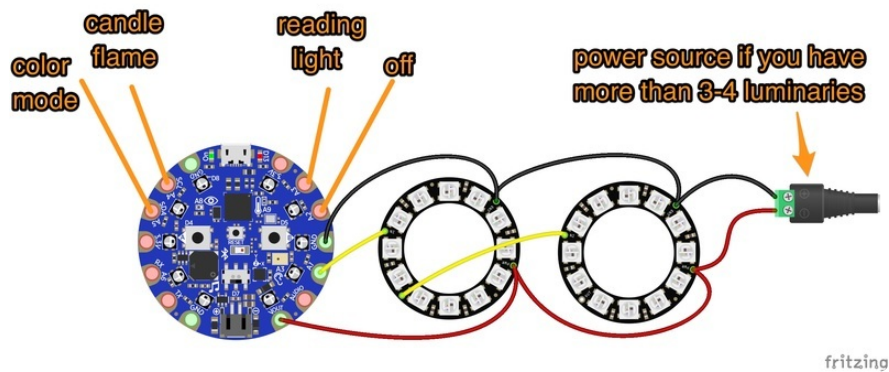
<https://www.adafruit.com/product/4306>

Materials & Tools

- Vinyl cutter or laser cutter (or a knife and lots of patience)
- Soldering Iron & accessories

- Pretty paper or poster board
- Clear plastic poster board for diffusion
- Glue (I used hot glue)
- Utility Knife and/or putty knife

Wiring Diagram



For One Luminary

If you have just one luminary, the wiring is very simple. Copper tape strands will attach to pins **A2**, **A3**, **A4** and **A5** for the controls on the outside of the luminary. For power, plug a [battery](http://adafru.it/328) or USB cable into your Circuit Playground Bluefruit and you're good to go! We will use the Circuit Playground's onboard NeoPixels, so no other wiring is required.

For 2-4 Luminaries

We can still power through the USB port or battery port on the Circuit Playground Bluefruit. Wiring to the NeoPixel rings is as follows:

- **VOUT --> 5V**
- **GND --> G**
- **A1 --> Data In**

Also connect the **Data Out** pin on the first NeoPixel ring to the **Data In** pin on the next one in the series, and so on. The copper tape connections are the same as above.

For 5+ Luminaries

If you have more than 4 luminaries, or if you want to make them all really bright, it's best to power the LEDs first so you aren't pulling so much current through the Circuit Playground Bluefruit. If you find your lights are "browning out" or your Circuit Playground hangs or crashes if you make the lights too bright, this is the solution.

Wiring for the NeoPixel rings is the same as above. For the power wires, solder an extra red and black wire to the **5V** and **G** pins on the last NeoPixel ring and connect to a [screw terminal \(http://adafru.it/369\)](http://adafru.it/369). Then plug into a 5v power supply ([you'll want at least 2A, this one works great \(http://adafru.it/276\)](http://adafru.it/276)) to power the luminaires.

The power flows both ways through the rings, so if you connect power at the end or in the middle of the chain of rings, the Circuit Playground Bluefruit will still power up and run just fine.

Software

This code uses the Adafruit Bluefruit app's Control Pad and Color Picker features. The Color Picker will send a solid color to the luminary, and the Control Pad will allow you to choose between four different modes. You can also speed up and slow down the animations to get just the look you want using the arrow keys.

For instant gratification, we've also made the four Control Pad modes accessible via capacitive touch "buttons" made from copper tape and stuck to the outside of the luminary. This way you don't have to look around for your phone just to turn out the lights.

Modes

1. Off
2. Reading Light (bright solid warm yellow)
3. Candle Flame
4. Rainbow Swirl

Your Circuit Playground Bluefruit should ship with the CircuitPython software already installed. At the time of writing this software is fairly new and changing rapidly, so it's still a good idea to update to the latest version of CircuitPython before you start. Here's what we'll do:

1. Install / Update CircuitPython

2. Install the necessary CircuitPython libraries
3. Copy and customize the Python code
4. Save the code to your board

Ready to start? Here we go!

Install CircuitPython

This guide tells you all you need to know about CircuitPython:

Install CircuitPython on your CPB

<https://adafru.it/GA4>

You probably want to follow the instructions to update CircuitPython to the latest version - this project requires version 5.0.0-beta.0 or higher.

Install Libraries

Now we need to install a few libraries onto our board. The libraries contain pre-written code blocks that we can reference to make our code simpler to write. Here's a guide that tells you all you'll ever want to know about installing libraries:

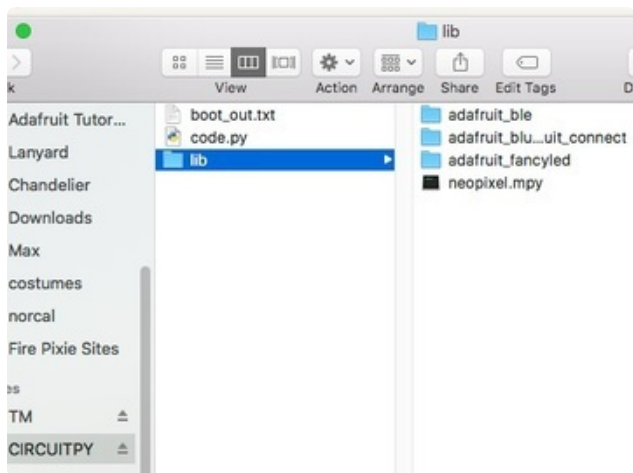
<https://learn.adafruit.com/welcome-to-circuitpython/circuitpython-libraries> (<https://adafru.it/ABU>)

I'll just hit the highlights again to get you up and running.

Download Circuit Python Library Bundle

<https://adafru.it/ENC>

Find the bundle that matches the version of CircuitPython you're running. At the time of writing, we're running 5.x so download the 5.x bundle for the Circuit Playground.



Now go to your **CIRCUITPY** drive and create a new folder called **lib**. Unzip the Library bundle and find:

adafruit_ble
adafruit_bluefruit_connect
adafruit_fancyled
neopixel.mpy

Drag these folders/files into your brand new **lib** folder.

Upload the Code

The last thing we need to add is a file called **code.py** on the **CIRCUITPY** drive. This is where the board will look for actual instructions on what to do. Copy the code below into a text or code editor -- we recommend the [Mu editor which can be downloaded here \(https://adafru.it/Be6\)](https://adafru.it/Be6).

```
# SPDX-FileCopyrightText: 2019 Phillip Burgess for Adafruit Industries
# SPDX-FileCopyrightText: 2019 Dan Halbert for Adafruit Industries
# SPDX-FileCopyrightText: 2019 Erin St Blaine for Adafruit Industries
#
# SPDX-License-Identifier: MIT

""" FancyLED Palette and Color Picker Control with BlueFruit App
    Code by Phil Burgess, Dan Halbert & Erin St Blaine for Adafruit Industries
"""
import board
import neopixel
import touchio
import adafruit_fancyled.adafruit_fancyled as fancy
from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.button_packet import ButtonPacket
from adafruit_bluefruit_connect.color_packet import ColorPacket

from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

NUM_LEDS = 24                # change to reflect your total number of ring LEDs
RING_PIN = board.A1         # change to reflect your wiring
CPX_PIN = board.D8          # CPX Neopixels live on pin D8

touch_A2 = touchio.TouchIn(board.A2)
touch_A3 = touchio.TouchIn(board.A3)
touch_A4 = touchio.TouchIn(board.A4)
touch_A5 = touchio.TouchIn(board.A5)
touch_A6 = touchio.TouchIn(board.A6)
touch_TX = touchio.TouchIn(board.TX)

# Palettes can have any number of elements in various formats
# check https://learn.adafruit.com/fancyled-library-for-circuitpython/colors
# for more info
```

```

# Declare a 6-element RGB rainbow palette
PALETTE_RAINBOW = [fancy.CRGB(1.0, 0.0, 0.0), # Red
                   fancy.CRGB(0.5, 0.3, 0.0), # Orange
                   fancy.CRGB(0.5, 0.5, 0.0), # Yellow
                   fancy.CRGB(0.3, 0.7, 0.0), # Yellow Green
                   fancy.CRGB(0.0, 1.0, 0.0), # Green
                   fancy.CRGB(0.0, 0.7, 0.3), # Teal
                   fancy.CRGB(0.0, 0.5, 0.5), # Cyan
                   fancy.CRGB(0.0, 0.3, 0.7), # Blue
                   fancy.CRGB(0.0, 0.0, 1.0), # Blue
                   fancy.CRGB(0.5, 0.0, 0.5), # Magenta
                   fancy.CRGB(0.7, 0.0, 0.3)] # Purple

# Reading Lamp mode - Warm Yellow
PALETTE_BRIGHT = [fancy.CRGB(255, 183, 55)]

# Black Only palette for "off" mode
PALETTE_DARK = [fancy.CRGB(0, 0, 0)]

# Declare a FIRE palette
PALETTE_FIRE = [fancy.CRGB(160, 30, 0), # Reds and Yellows
                fancy.CRGB(27, 65, 0),
                fancy.CRGB(0, 0, 0),
                fancy.CRGB(224, 122, 0),
                fancy.CRGB(0, 0, 0),
                fancy.CRGB(250, 80, 0),
                fancy.CRGB(0, 0, 0),
                fancy.CRGB(0, 0, 0),
                fancy.CRGB(200, 40, 0)]

# Declare a NeoPixel object on NEOPIXEL_PIN with NUM_LEDS pixels,
# no auto-write.
# Set brightness to max because we'll be using FancyLED's brightness control.
ring = neopixel.NeoPixel(RING_PIN, NUM_LEDS, brightness=1.0, auto_write=False)
cpx = neopixel.NeoPixel(CPX_PIN, NUM_LEDS, brightness=1.0, auto_write=False)

offset = 0 # Positional offset into color palette to get it to 'spin'
offset_increment = 6
OFFSET_MAX = 1000000

# Setup BLE
ble = BLERadio()
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

def set_palette(palette):
    for i in range(NUM_LEDS):
        # Load each pixel's color from the palette using an offset, run it
        # through the gamma function, pack RGB value and assign to pixel.
        color = fancy.palette_lookup(palette, (offset + i) / NUM_LEDS)
        color = fancy.gamma_adjust(color, brightness=1.0)
        ring[i] = color.pack()
    ring.show()

    for i in range(NUM_LEDS):
        # Load each pixel's color from the palette using an offset, run it
        # through the gamma function, pack RGB value and assign to pixel.
        color = fancy.palette_lookup(palette, (offset + i) / NUM_LEDS)
        color = fancy.gamma_adjust(color, brightness=1.0)
        cpx[i] = color.pack()
    cpx.show()

# set initial palette to run on startup
palette_choice = PALETTE_FIRE

# True if cycling a palette
cycling = True

```

```

# Are we already advertising?
advertising = False

while True:

    if cycling:
        set_palette(palette_choice)
        offset = (offset + offset_increment) % OFFSET_MAX

    if not ble.connected and not advertising:
        ble.start_advertising(advertisement)
        advertising = True

# Are we connected via Bluetooth now?
if ble.connected:
    # Once we're connected, we're not advertising any more.
    advertising = False
    # Have we started to receive a packet?
    if uart.in_waiting:
        packet = Packet.from_stream(uart)
        if isinstance(packet, ColorPacket):
            cycling = False
            # Set all the pixels to one color and stay there.
            ring.fill(packet.color)
            cpx.fill(packet.color)
            ring.show()
            cpx.show()
        elif isinstance(packet, ButtonPacket):
            cycling = True
            if packet.pressed:
                if packet.button == ButtonPacket.BUTTON_1:
                    palette_choice = PALETTE_DARK
                elif packet.button == ButtonPacket.BUTTON_2:
                    palette_choice = PALETTE_BRIGHT
                elif packet.button == ButtonPacket.BUTTON_3:
                    palette_choice = PALETTE_FIRE
                    offset_increment = 6
                elif packet.button == ButtonPacket.BUTTON_4:
                    palette_choice = PALETTE_RAINBOW
                    offset_increment = 1

            # change the speed of the animation by incrementing offset
            elif packet.button == ButtonPacket.UP:
                offset_increment += 1
            elif packet.button == ButtonPacket.DOWN:
                offset_increment -= 1

# Whether or not we're connected via Bluetooth,
# we also want to handle touch inputs.
if touch_A2.value:
    cycling = True
    palette_choice = PALETTE_DARK
elif touch_A3.value:
    cycling = True
    palette_choice = PALETTE_BRIGHT
elif touch_A4.value:
    cycling = True
    palette_choice = PALETTE_FIRE
    offset_increment = 6
elif touch_A5.value:
    cycling = True
    palette_choice = PALETTE_RAINBOW
    offset_increment = 1
# Also check for touch speed control
# if touch_A6.value:
#     offset_increment += 1
# if touch_TX.value:
#     offset_increment -= 1

```

Once you've got the code in your editor, look near the top and find this line:

```
NUM_LEDS = 24           # change to reflect your total number of ring LEDs
RING_PIN = board.A1     # change to reflect your wiring
CPX_PIN = board.D8      # CPX Neopixels live on pin D8
```

NUM_LEDS refers to the total number of NeoPixels you have just in the rings -- NOT including the pixels on the face of the Circuit Playground Bluefruit. Since I have two 12-pixel rings attached, **NUM_LEDS = 24** in my case.

Change this number to reflect your total number of LEDs.

You can also play with this number to change the feel of your animations. As long as the number is greater than your actual number of pixels (so, not less than 24 in this case) you can make the animations more "spread out" and more slow-moving by telling the code you have more pixels than you actually have. Try making this number 34 or 44 and see what kind of results you get.

Customizing Palettes

I've added four different color palettes for the animations accessed from the Control Pad: a "dark" palette (for turning the lights off), a "light" palette for the reading light mode, a "fire" and a "rainbow" palette. You can customize these fairly easily in the code. The power of the FancyLED library allows you so much control when it comes to choosing custom colors and animating them smoothly.

Find the palette definitions in the code:

```
# Palettes can have any number of elements in various formats
# check https://learn.adafruit.com/fancyled-library-for-circuitpython/colors for
more info
# Declare a 6-element RGB rainbow palette
PALETTE_RAINBOW = [fancy.CRGB(1.0, 0.0, 0.0), # Red
                   fancy.CRGB(0.5, 0.3, 0.0), # Orange
                   fancy.CRGB(0.5, 0.5, 0.0), # Yellow
                   fancy.CRGB(0.3, 0.7, 0.0), # Yellow Green
                   fancy.CRGB(0.0, 1.0, 0.0), # Green
                   fancy.CRGB(0.0, 0.7, 0.3), # Teal
                   fancy.CRGB(0.0, 0.5, 0.5), # Cyan
                   fancy.CRGB(0.0, 0.3, 0.7), # Blue
                   fancy.CRGB(0.0, 0.0, 1.0), # Blue
                   fancy.CRGB(0.5, 0.0, 0.5), # Magenta
                   fancy.CRGB(0.7, 0.0, 0.3)] # Purple

# Reading Lamp mode - Warm Yellow
PALETTE_BRIGHT = [fancy.CRGB(255, 183, 55)]

# Black Only palette for "off" mode
PALETTE_DARK = [fancy.CRGB(0, 0, 0)]

# Declare a FIRE palette
PALETTE_FIRE = [fancy.CRGB(160, 30, 0), # Reds and Yellows
```

```
fancy.CRGB(27, 65, 0),  
fancy.CRGB(0, 0, 0),  
fancy.CRGB(224, 122, 0),  
fancy.CRGB(0, 0, 0),  
fancy.CRGB(250, 80, 0),  
fancy.CRGB(0, 0, 0),  
fancy.CRGB(0, 0, 0),  
fancy.CRGB(200, 40, 0)]
```

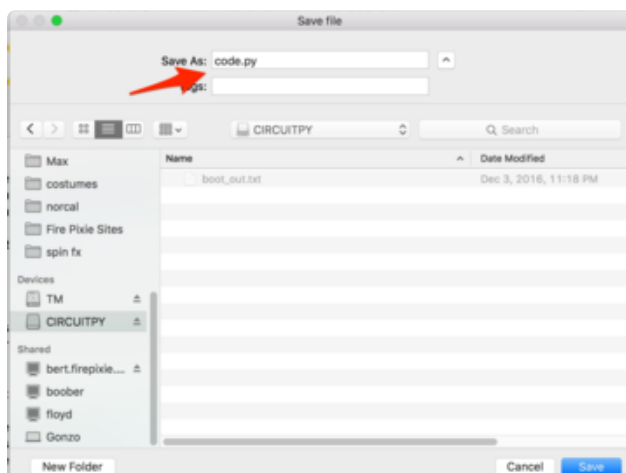
You can use CRGB values or CHSV values to choose colors, or use them both at the same time. There are also multiple ways to declare values and a lot of control over how spread out the gradients can be.

This is explained in detail in the [FancyLED guide \(https://adafru.it/Fra\)](https://adafru.it/Fra) so take a look to find out all you need to know about creating your own custom color palettes.

Offset Increment

Once you've customized your palettes, you can also customize the speed of the animation. Look for `offset_increment` in the code. A higher number will make the palettes swirl faster. Since I want the candle flame mode to look flickery, I've set `offset_increment` to `6` for that mode. I want the rainbow mode to be slow and smooth, so I've set the increment to `1`.

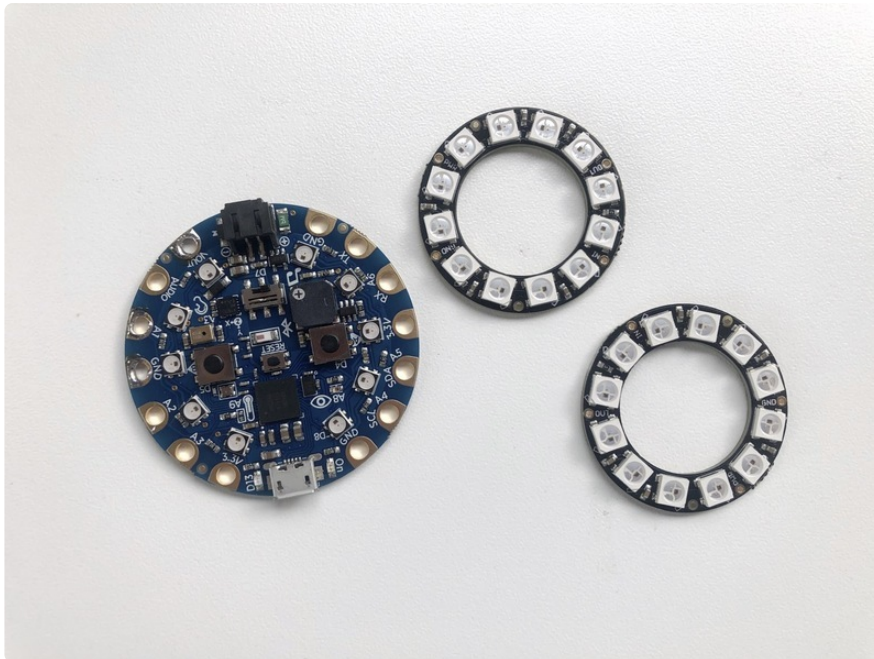
You can play with these values to customize the speed of your animations. Just note that you need to change this in TWO places in the code. Lines 124-134 change modes while using Bluetooth, and lines 143-156 change modes via the capacitive touch buttons.



Save the code on your **CIRCUITPY** drive, called **code.py** and the candle flame animation should start. You can test and make changes as many times as you'd like. Clicking "save" will push the changes to your board so you can see the results.

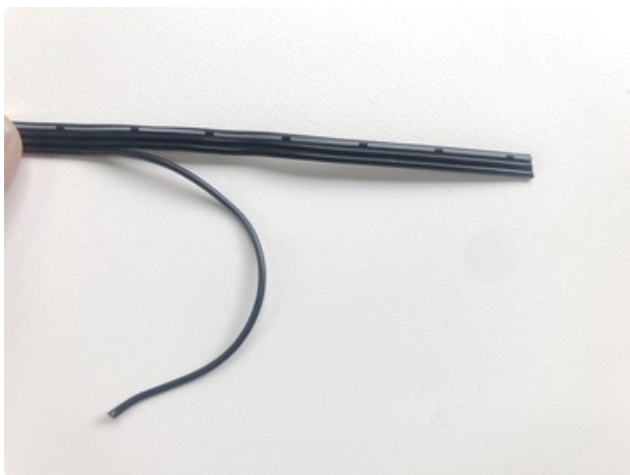
That's it! You're ready to start controlling your lights with Bluetooth.

Electronics Assembly



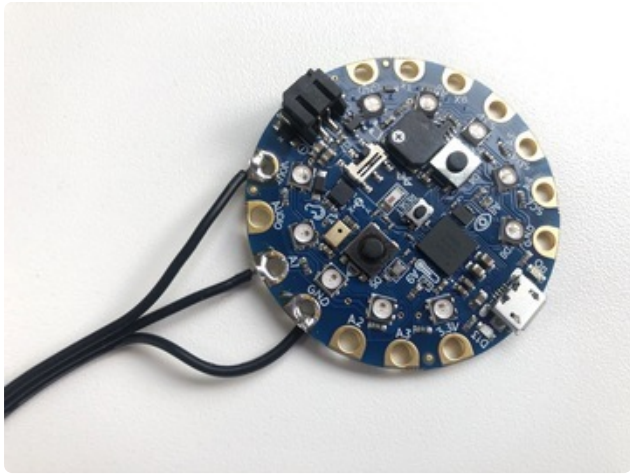
If you're just making one luminary, you can skip most of this page -- the first luminary will use the onboard lights on the Circuit Playground. I'll show wiring up two additional rings for additional lanterns.

If you want to make more than 5 luminaries, take another look at the [Wiring Diagram \(https://adafru.it/H6e\)](https://adafru.it/H6e) page of this guide. You'll want to power your LEDs a bit differently than I am doing with my 3 lanterns.

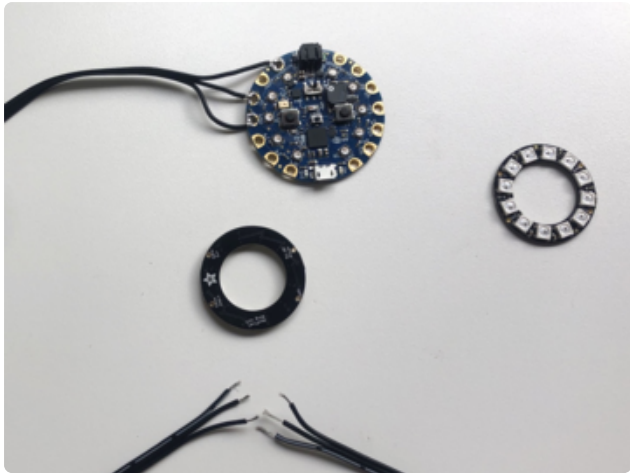


Open up your ribbon cable wire and find the striped wire. We only need three wires, and this cable comes with four, so we can strip off the wire on the **opposite** side from the striped wire and save it for another project. (Don't strip off the striped wire -- we'll need that one!)

Decide how far apart you'd like your luminaries spaced. Cut appropriate lengths of ribbon cable to stretch comfortably between luminaries with a bit of slack. Strip the fourth wire from these lengths as well.



Strip a little bit off each of your 3 wires and separate them for about an inch or so. Solder your striped wire to **VOUT**, the middle wire to **A1** and the third wire to **G**.



The holes on the rings are very tiny, and if you're daisy-chaining rings you'll need a wire going in and a wire going out of the 5V and G holes. The easiest way to achieve this is to strip a little off two separate wires and twist them tightly together before inserting into the holes in the rings.



Twist the end of the striped wire coming from the Circuit Playground together with the striped wire heading to the second NeoPixel ring. Insert the wires into the 5V DC Power hole from the **front** of the ring. Flip the ring over and solder to the hole on the **back** of the ring.

This is much easier to manage than trying to solder on the front of the ring, where it's easy to damage the LEDs.

Repeat with the **G** connection, using the 3rd wire in the ribbon cable.



The middle wire is your data wire. It should be coming from **A1** on your Circuit Playground. Solder it into the **Data Input** pin.



Then, solder a wire from the **Data Output** pin and connect it to the **Data Input** pin on the next ring in the series, and so forth.

Solder the **Power** and **G** wires to the next ring and so on, until all your rings are connected.



Now is a great time to power up your board and make sure everything is working. Plug your board in via USB port or battery (or screw terminal if you've got more than 5 lanterns). Touch pads **A2**, **A3**, **A4** and **A5** and watch the lights change colors. If all the lights aren't coming on, this is a great time to double check your wiring and make sure you've got everything hooked up correctly.

Design the Luminary

I'm using a Cricut brand vinyl cutting machine to make my luminary. You can use any brand of cutter you'd like, and laser cutters will work just as well. The trick is in getting the right materials for optimum diffusion and beauty.



Materials

I went to my local craft store and found the poster board section. My [Cricut \(https://adafru.it/Hdh\)](https://adafru.it/Hdh) will cut poster board up to 2mm thick, so most of the thin flexible poster board at the craft store will work perfectly for this project.

My luminaries have two layers: a translucent diffusion layer, which is simple and structural, and the fancy outer layer made of pretty sparkly or colorful paper or poster board.

The Michael's craft store by my house sells a [clear plastic poster board \(https://adafru.it/Hdi\)](https://adafru.it/Hdi) that is a wonderful diffusion material. It's flexible but fairly stiff and it diffuses light really well. It's also pretty cheap -- I paid two dollars for a huge poster-sized sheet of this stuff, which was enough for all 3 of my luminaries.

For the outside layer, I had the most success with light, sparkly poster board. It's tempting to use the pretty holographic or glittery sticky-back vinyl that's sold in rolls in the vinyl cutter section. However, with these intricate cuts and large surface areas, this stuff turns into nightmare soup when you have to weed and peel and stick it. It

looks really nice, but the non-sticky poster board was much easier to work with, and sticks just as well with a little spray glue.

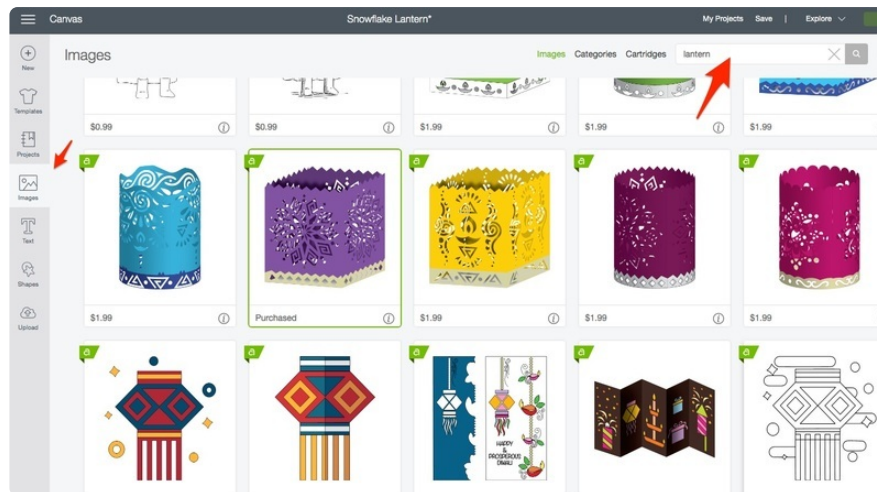
Poster board is also insanely cheap compared to branded rolls of vinyl, which can run between \$15-\$20 per roll, often with less actual surface area than a \$0.99 poster board.



Design

There are lots of ways to design your luminary. The quickest and easiest way is to find a pre-existing design that you like and purchase it, and then customize to your heart's content. Many really high quality designs are available for around a dollar or two. In the Cricut Design Space software, click "Images" from within the workspace and do a search on "luminary" or "lantern" to see all the designs that are available.

It's not hard to customize the designs, and it's often worth a dollar or two to know you've got a tried-and-true design complete with tabs and score lines and everything.



Cricut won't let me publish and share my project, so here's a video of how I customized this design to house the Circuit Playground.

Video Transcript

I'm in the Cricut Design space, and I'm going to show you how to customize your design for use with a Circuit Playground Express.

It's easiest to start with a premade lantern design. You can find lots of designs for just a dollar or two under the Images tab. I did a search on the word "lantern" and found one I liked.

Insert your design into the workspace. You won't need to pay for it until you're ready to cut, so don't be afraid to experiment and try out different base designs.

The first thing we'll do is ungroup the layers so we can start messing with them.

I want the base of the lantern and the bottom to be cut out of clear poster board. I only need one base so I'm going to delete this other square. The Crickit will cut like colors out of the same material so I'll make the bottom the same color as the base.

This lantern is just one layer thick and I want two layers for better light diffusion. To make the diffusion layer I'll copy and paste the walls of the lantern to make a duplicate. Now I'll erase all the designs to make it plain.

Grab a square from the shapes menu. Click the lock button to unlock the aspect ratio and drag it so it covers all the flower designs. Select both layers — I'm holding down "shift" to select more than one layer — and choose Weld.

Now we have a nice blank but perfectly sized diffusion layer. Let's make it the same color as the base so all three pieces will be cut from the same material.

Next, I want to add a custom snowflake design instead of the flower design on the sides of the lantern. I'll use the same trick of welding a box to the shape, only this time I'll leave the diamonds along the bottom. Now I've got a nice blank slate for my design.

I made my snowflake using [Paul Kaplan's fabulous online snowflake generator \(https://adafru.it/Hdj\)](https://adafru.it/Hdj). Click and drag the circles to create your dream snowflake, and then you can easily import it into the Crickit Design space.

One thing to be careful of: this snowflake generator will allow you to cross the lines and make really intricate snowflakes with negative space. These look fabulous, but will be much harder to work with on the vinyl cutter. Each disconnected piece will need to be handled separately, which will make your life miserable later on. Simple is your friend, at least at first.

Once you're happy with your snowflake, save it as an .svg so you can import it into the Crickit space.

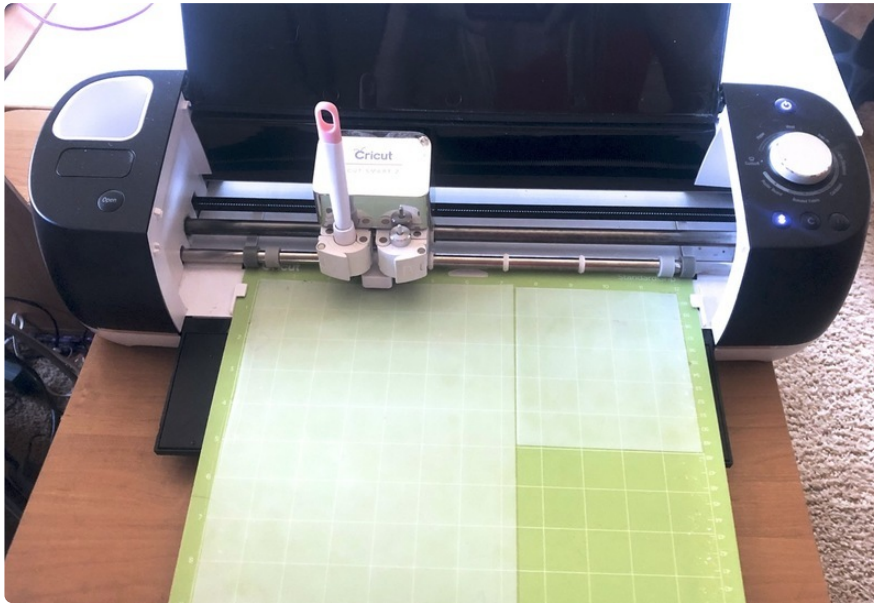
Upload the file and import it into your workspace. Duplicate it with copy/paste, and use the align and distribute tools to get your snowflakes lined up perfectly. Once you're happy with alignment, select all your snowflakes, and choose weld. Then select your background and choose slice. Delete the extra layers.

You can add more shapes or decorations right within the design space. Go nuts until your luminary is perfect. I'm going to keep it simple for this video, the tools are pretty self explanatory. Remember also to leave space for your capacitive touch buttons if you're adding them.

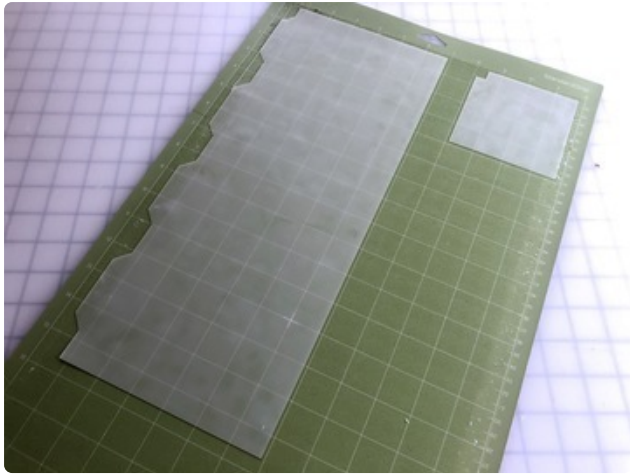
If you want to resize your lantern, be sure to select all the pieces and resize them together at the same time. That way they'll still fit together at the larger size.

The Crickit arranges the pieces onto your mats by color, so you can cut all the like materials at the same time. I've got my diffusion layer and base on one mat, and my pretty design on the other. Set your machine to "poster board" or "poster board plus" and get cutting.

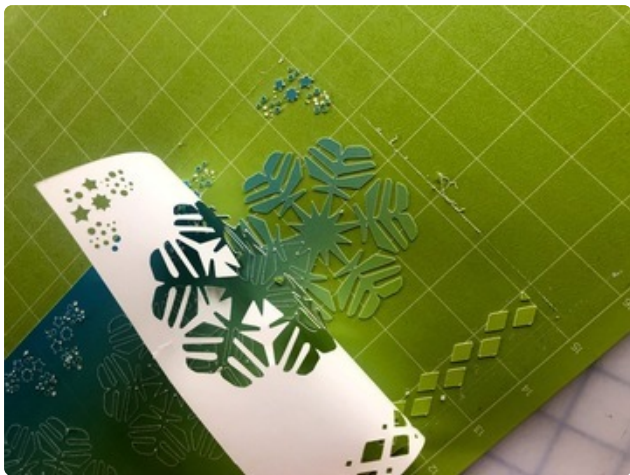
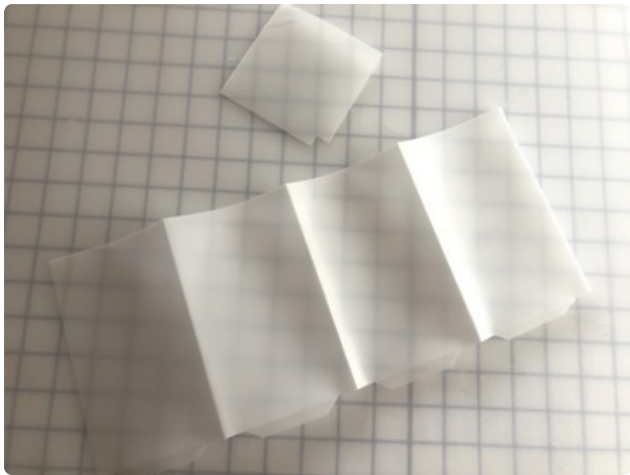
Cutting & Assembly



Once you're happy with your design, cut it out on your vinyl cutter or laser cutter. You could also do this by hand with a sharp knife and a lot of patience. Personally, I find the vinyl cutter to be the easiest option. They're fairly inexpensive and user friendly and I'm often surprised at the many uses I find for mine.



The Cricut has a scoring pen accessory that will score lines into your material so you know where to fold. I have one. It doesn't really seem to do anything, so don't waste your money. Once your diffusion layer is cut, you'll want to figure out the fold lines and make some creases.



The fancy design layer comes next. This is the wickedly tricky bit. You'll need to peel your poster board from the mat without tearing off any of the little fiddly snowflake bits. Be patient. Take a lot of deep breaths, and have some knife blades or putty knives or other loosening tools available.

If something gets torn or left behind, don't worry -- small mistakes will be fixable in the next step.



Next we'll affix the poster board to the diffusion layer. I'm using a mild spray glue. Spray the back of the poster board and stick it to the plastic diffusion layer, NOT the other way around. That way the glue won't get all over the cutouts and look bumpy and weird.

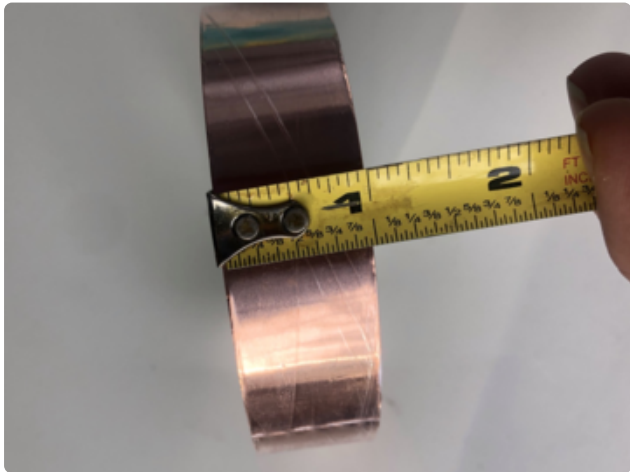
If you tore off any little bits, carefully spray a little glue on them and nestle them gently into place.



Stick the two pieces together, making sure your creases line up, and press down firmly.

Next we'll get the capacitive touch switches in place.

Copper Tape Cutting



I'm using 1" wide copper tape, so each of my switch icons need to be narrower than that. I've designed icons for "off," "reading light," "candle flame" and "color" modes. Here are my image files. Use them if you like, or use them as a template to design your own icons.

[icons.zip](#)

<https://adafru.it/H6B>

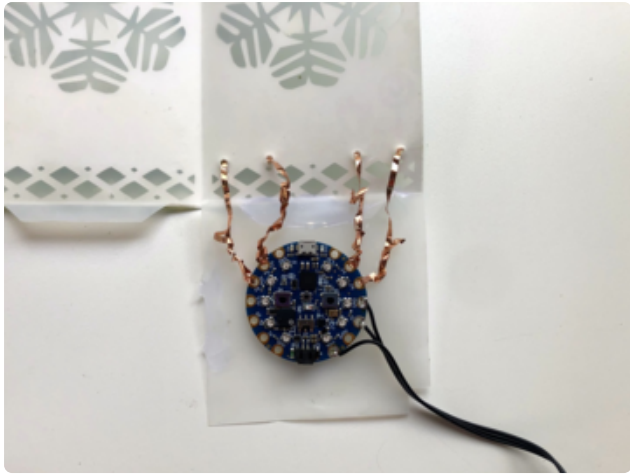


Stick four pieces of tape side by side on your cutting mat. You'll want to align each icon within the inch-measurement on the mat so they each get cut out of a complete piece of tape.

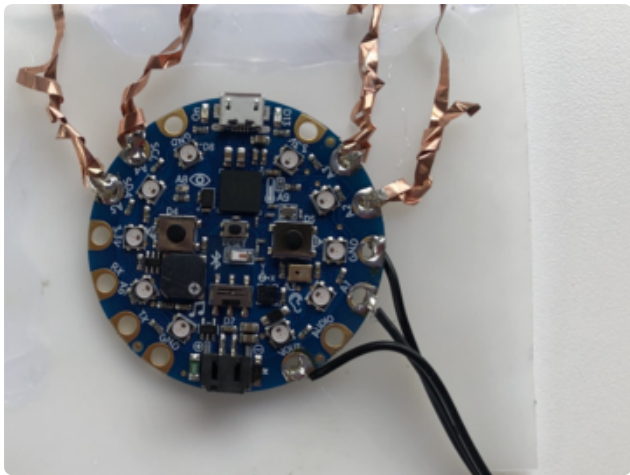
I chose Vinyl for the setting and they came out just perfectly. Weed the extra tape from around the edges and inside the cutouts with a utility knife.



Use an awl or knife to poke a few small holes in the side of your luminary. Feed the tails of the tape buttons through the holes, and stick the copper icons to the front of your lantern.



The tails will reach through the lantern and connect to your Circuit Playground Bluefruit's pads **A2**, **A3**, **A4** and **A5**. Make sure not to "cross the streams" as the tape isn't shielded and if the tails touch each other, things will get screwy.



It's helpful to add a blob of solder on the pad where the tape goes through the Circuit Playground Bluefruit to improve your connection.

Assemble the rest of your luminary. I found it easiest to start by hot-gluing the Circuit Playground to the base layer (make sure there's room for your USB cable or battery cable to get in), then attach the base to the lantern's tabs. Once the bottom is all settled, then glue the sides.



I also cut the corner from the luminary's bottom piece to make room for the wires to feed through.

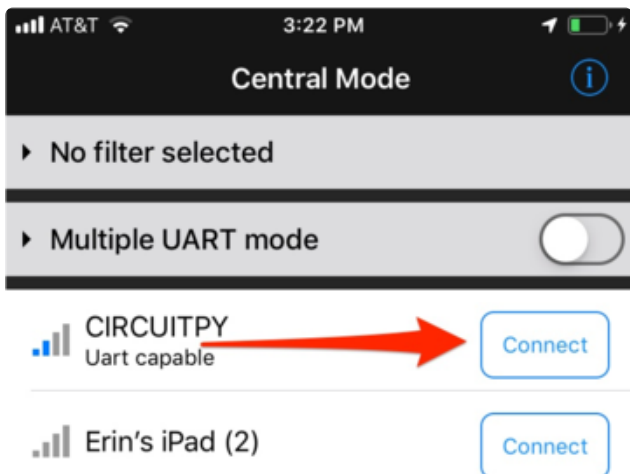
Power it up with a USB cable or battery and test it out! Do your capacitive touch buttons work? If they're twitchy, clean them with 99% alcohol and make sure the copper tape inside the lanterns is not crossed or touching.

For the remaining lanterns, assemble the same way and use a bit of hot glue to secure the NeoPixel rings in place.

Use It

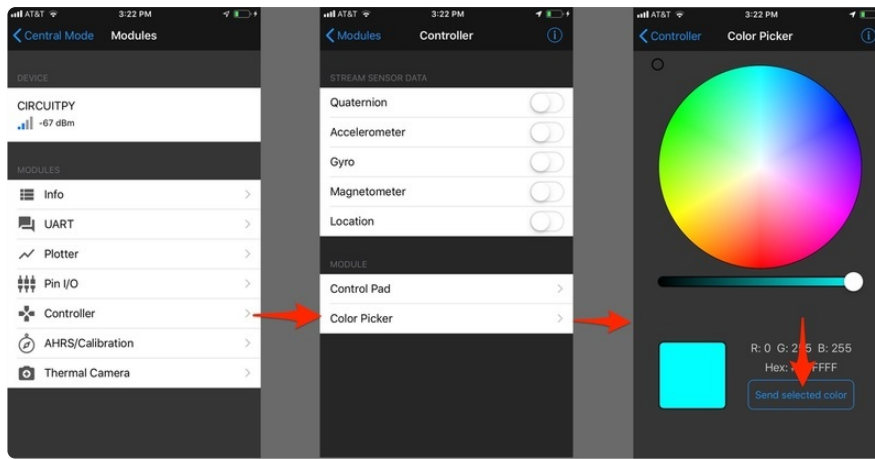
Adafruit Bluefruit App

Go to the app store (iOS or Android) and find and install the Adafruit Bluefruit app on your smartphone.

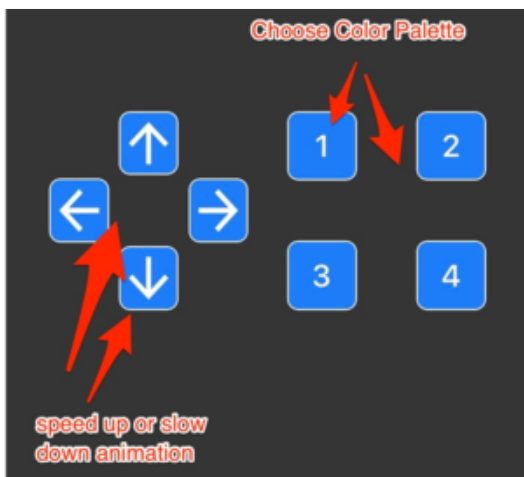


Launch the app. Make sure you're in "Central Mode" and that your Circuit Playground Bluefruit is powered on. You should see the **CIRCUITPY** drive appear on the main screen. If it's not there, try pulling down to refresh or restarting the app.

If you still don't see it, try re-uploading your code, making sure you have all the libraries installed. It won't work without all the libraries! CircuitPython still lets you save the file without throwing an error, so double check you've got everything you need.



Click Connect, then Controller > Color Picker. Choose a color and press the "Send selected color" button. Your luminaries will update with the new color. Like magic!



Now go back to the Control Pad. You'll see four buttons and four arrows. The number buttons will select between the four color palettes you made, and the up and down arrows will speed up or slow down the animation of the gradient.

I made my capacitive touch buttons match up with the buttons in the Bluefruit app, so my four switches work the same as buttons 1-4. You don't need to do it this way! Dig into the code a bit and you'll find it's fairly simple to add or change palettes or functionality in the code. Create your own magic luminous color modes and UI to suit your home and sensibilities.

