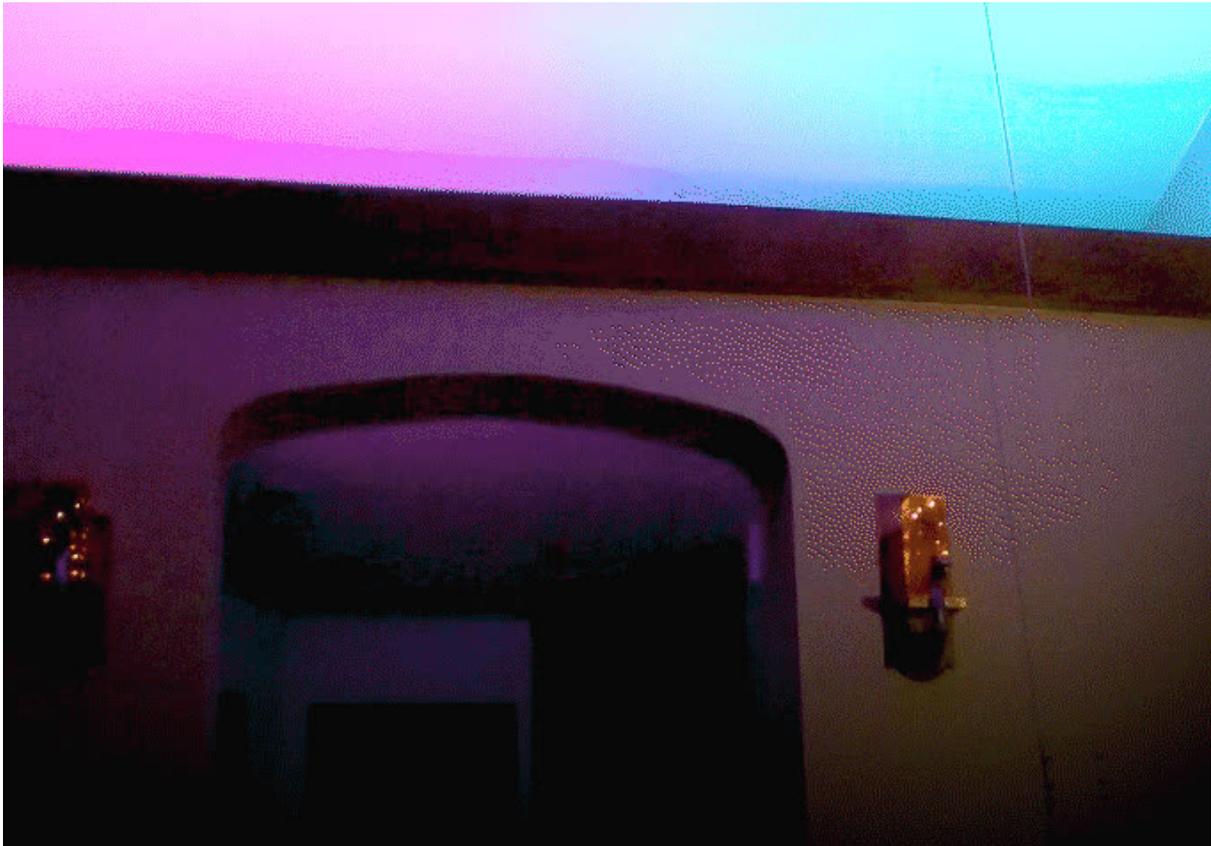




Easy No-Soldering Bluetooth Controlled Room Lights

Created by Erin St Blaine



<https://learn.adafruit.com/easy-no-solder-bluetooth-controlled-room-lights>

Last updated on 2024-06-03 03:13:01 PM EDT

Table of Contents

Overview	3
• Parts	
Wiring Diagram	6
Code with CircuitPython	6
• How To Upload the Code	
Customizing Your Code	11
BlueFruit App Controls	13
Electronics Assembly	16
• Troubleshooting	
Installation	21

Overview

Install a strip of NeoPixel lights in your bedroom or living space. Hook them up to a Circuit Playground Bluefruit and control the lights with your phone or tablet. You can add as many different animation modes as you like - it's really easy to do with Adafruit's [CircuitPython LED Animations Library](https://adafru.it/LZF) (<https://adafru.it/LZF>).

Choose from millions of colors with our graphical color picker. And the library includes around a dozen pre-written animations that you can customize to your heart's content including Rainbow, Pulse, Sparkle Pulse, Color Chase, Comet, and more. Within each animation you can set custom colors, speeds, directions and brightness. You can even layer the animations over one another to create your own custom animated lighting scenes.

Cycle through modes and fine-tune your brightness levels using Adafruit's free BlueFruit app, which works seamlessly with our CircuitPython sample code.



Difficulty Level

Assembly:

This is a beginner level project! There's no soldering required - the whole project can be assembled with a screwdriver and a pair of wire strippers!

Code:

For setup, you'll need to download and drag the latest version of the free firmware to your Circuit Playground Bluefruit, then copy and paste a couple of library files from our downloadable bundle.

You can then copy and paste our code directly onto the board, or install the free Mu code editor if you want to customize your animations. The code for this project is fairly simple and easy to understand and experiment with. This is a great place to start if you're new to CircuitPython.

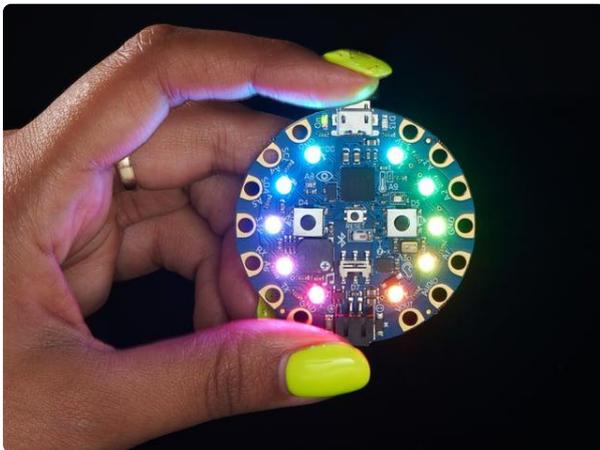
Cost:

The cost will depend on how many lights you want to install. For my room, a 4 meter strip of 60/m lights is enough to light two rooms when placed near the ceiling along the room divider. The total cost for my project came in at around \$150.

Comparatively, the Philips Hue line charges \$239 for a light strip that's about half as long, and that doesn't include the control system -- and it's not customizable.

With our solution you'll get to dig into the nuts and bolts a bit, and you'll end up with a one of a kind Home Automation lighting system you can be proud of.

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>



[Adafruit NeoPixel Digital RGB LED Strip - White 60 LED](https://www.adafruit.com/product/1138)

You thought it couldn't get better than our world-famous 32-LED-per-meter Digital LED strip but we will prove you wrong! You wanted...

<https://www.adafruit.com/product/1138>



[Bolt-On Kit for Circuit Playground, micro:bit, Flora or Gemma](https://www.adafruit.com/product/4103)

You have a Circuit Playground Express, and want to connect some wires to it for adding LEDs or sensors or speakers? You can use our...

<https://www.adafruit.com/product/4103>



[5V 4A \(4000mA\) switching power supply - UL Listed](https://www.adafruit.com/product/1466)

Need a lot of 5V power? This switching supply gives a clean regulated 5V output at up to 4 Amps (4000mA). 110 or 240 input, so it works in any country. The plugs are "US...

<https://www.adafruit.com/product/1466>

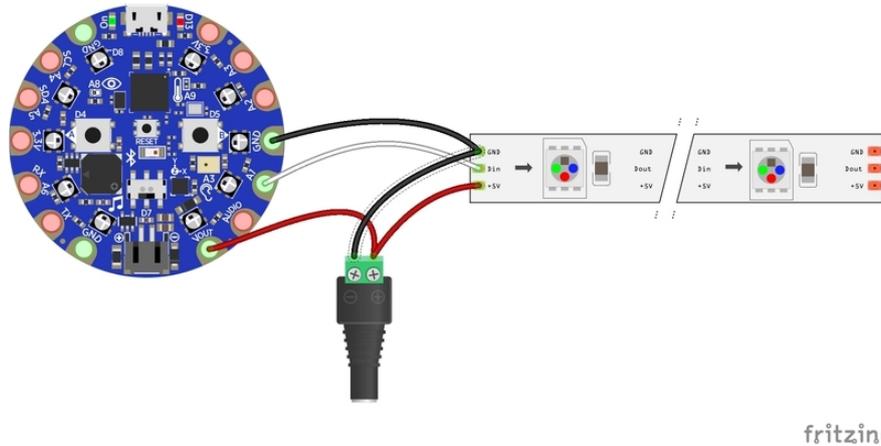


[Female DC Power adapter - 2.1mm jack to screw terminal block](https://www.adafruit.com/product/368)

If you need to connect a DC power wall wart to a board that doesn't have a DC jack - this adapter will come in very handy! There is a 2.1mm DC jack on one end, and a screw terminal...

<https://www.adafruit.com/product/368>

Wiring Diagram



We'll use the preattached wires from the LED strip to attach to the power supply. The red wire goes to + and the black to -.

We'll add an additional wire from the + side of the power supply to **VOUT** on the Circuit Playground, and attach the black wire from the NeoPixel strip **GND** to Circuit Playground **GND**. Finally, we'll attach the white wire from **DIN** on the NeoPixel strip to **A1** on the Circuit Playground.

If your setup has more than around 250 lights, or if you want to turn the brightness ALL the way up with your lights on full white, you will need to add a second power screw terminal to the other end of your strip as well (red to + and black to -) and connect power at both ends to avoid strip brown-out.

Code with CircuitPython

CircuitPython is a fast growing programming platform based on Python, that's easy to learn and customize. This project makes use of the [Circuit Python LED Animations \(https://adafru.it/LZF\)](https://adafru.it/LZF) code by [Kattni Rembor \(https://adafru.it/L-a\)](https://adafru.it/L-a) to quickly and easily add gorgeous animations to your light strand. There are more pre-made animations available than I'm using for this guide, so go check it out if you want more modes on your light strand.

How To Upload the Code

Download the Latest Version of
CircuitPython for Circuit Playground
Bluefruit

<https://adafru.it/FNK>

Step 1: Update CircuitPython

1. Download the latest version of the Circuit Python operating system using the button above.
2. Plug your Circuit Playground into your computer via its USB port and double-click the "reset" button. All the lights will turn green and your Circuit Playground will appear as a drive on your computer called **CPLAYBTBOOT**.
3. Drag the code you just download onto this drive to install CircuitPython -- like putting files on a USB stick.

Note: If you plug in the board and you see a drive called **CIRCUITPY** appear, press the reset button again (double-click) to get to **CPLAYBTBOOT**.

[Download CircuitPython Library Bundle](#)

<https://adafru.it/ENC>

Step 2: Install the Required Libraries

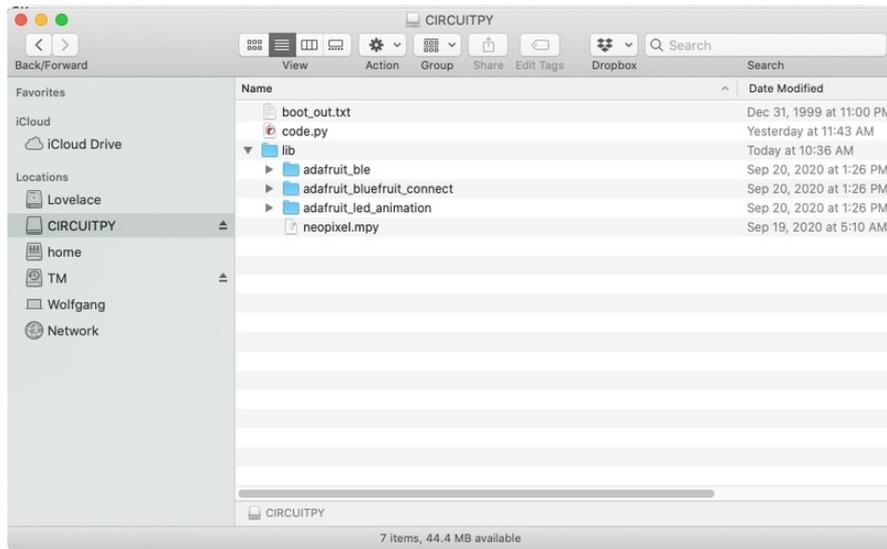
1. Click the button above to download the latest CircuitPython Library Bundle Release.
2. Open the file you just downloaded and look in the **lib** folder. Find these files and copy them into the **lib** folder on your Circuit Playground's **CIRCUITPY** drive.

- **adafruit_ble** (directory)
- **adafruit_bluefruit_connect** (directory)
- **adafruit_led_animation** (directory)
- **neopixel.mpy**

Step 3: Upload the Code

Copy the code from the code window below and save it as **code.py** at the root of your **CIRCUITPY** drive.

Your **CIRCUITPY** drive should look like this when you're done:



```

# SPDX-FileCopyrightText: 2020 Kattni Rembor for Adafruit Industries
# SPDX-FileCopyrightText: 2020 Erin St Blaine for Adafruit Industries
#
# SPDX-License-Identifier: MIT

"""
Bluetooth Controlled Room Lights using a Circuit Playground Bluetooth
    Scroll between 7 modes and control brightness with your smartphone via Bluetooth
    Full tutorial: https://learn.adafruit.com/easy-no-solder-bluetooth-controlled-room-lights/overview
Code by Kattni Rembor & Erin St Blaine for Adafruit Industries
Adafruit invests time and resources to bring you this code! Please support our shop!
"""

# pylint: disable=attribute-defined-outside-init
# pylint: disable=too-few-public-methods

import board
import neopixel
from adafruit_led_animation.animation.solid import Solid
from adafruit_led_animation.animation.comet import Comet
from adafruit_led_animation.animation.rainbow import Rainbow
from adafruit_led_animation.animation.rainbowcomet import RainbowComet
from adafruit_led_animation.animation.sparkle import Sparkle
from adafruit_led_animation.animation.sparklepulse import SparklePulse
from adafruit_led_animation.sequence import AnimationSequence
from adafruit_led_animation.group import AnimationGroup
from adafruit_led_animation.animation import Animation
from adafruit_led_animation.sequence import AnimateOnce
from adafruit_led_animation.color import (
    AMBER,
    ORANGE,
    WHITE,
    RED,
    BLACK,
    colorwheel,
)

)

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

from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.button_packet import ButtonPacket
from adafruit_bluefruit_connect.color_packet import ColorPacket

```

```

NUM_LEDS = 240                # change to reflect your LED strip
NEOPIXEL_PIN = board.A1      # change to reflect your wiring

# Declare a NeoPixel object on NEOPIXEL_PIN with NUM_LEDS pixels,
# no auto-write.
# Set brightness to max, we'll control it later in the code
pixels = neopixel.NeoPixel(NEOPIXEL_PIN, NUM_LEDS, brightness=1.0,
                           auto_write=False,
                           #pixel_order=(1,0,2,3) #uncomment if using RGBW NeoPixels
                           )

ble = BLERadio()
uart_service = UARTService()
advertisement = ProvideServicesAdvertisement(uart_service)

class RainbowFade(Animation):
    ''' fades the entire strip through the whole spectrum '''
    _color_index = 150 # choose start color (0-255)
    def __init__(self, pixel_object, speed, name): # define animation
        super().__init__(pixel_object, speed=speed, color=WHITE, name=name)

    def draw(self): # draw the animation
        ''' fades the entire strip through the whole spectrum '''
        self.color = colorwheel(self._color_index + 1)
        self._color_index = (self._color_index + 1) % 256
        self.fill(self.color)

# ANIMATION DEFINITIONS --
#   create as many animations as you'd like and define their attributes here.
#   They can be a single line or a group of animations - the groups will play
#   at the same time, overlaid on top of each other.

readingLight = Solid(pixels, color=0xFF7D13) #warm white color HEX code
brightWhite = Solid(pixels, color=(150, 150, 150))
rainbow = Rainbow(pixels, speed=0.1, period=10, step=0.5)
rainbowfade = RainbowFade(pixels, speed=0.4, name="rainbowfade")
powerup = RainbowComet(pixels, speed=0, tail_length=50, bounce=False)
off = Solid(pixels, color=BLACK)

#startup animation will play just once
startup = AnimateOnce(powerup)

#starrynight and fire are animation groups with layered effects.
starrynight = AnimationGroup(
    SparklePulse(pixels, speed=0.01, color=(0, 0, 150), period=1),
    Comet(pixels, speed=0, tail_length=8, color=(150, 150, 150), bounce=False),)

fire = AnimationGroup(
    Comet(pixels, speed=0, tail_length=1, color=BLACK),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=AMBER),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=RED),
    Sparkle(pixels, speed=0.05, num_sparkles=20, color=ORANGE),
    Sparkle(pixels, speed=0.05, num_sparkles=5, color=0xFF7D13),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=BLACK),
)

# Here is the animation playlist where you set the order of modes

animations = AnimationSequence(
    readingLight,
    fire,
    rainbow,
    starrynight,
    rainbowfade,
    brightWhite,
    auto_clear=True,
)

```

```

MODE = 0

while True:
    if MODE == 0: # If currently off...
        startup.animate()
        while startup.animate():
            pass
        MODE = 1
    # Advertise when not connected

    elif MODE >= 1: # If not OFF MODE...
        ble.start_advertising(advertisement)
        while not ble.connected:
            if MODE == 2:
                pass
            elif MODE == 1:
                animations.animate()
        # Now we're connected

    while ble.connected:
        if uart_service.in_waiting:
            packet = Packet.from_stream(uart_service)
            # Color Picker Functionality
            if isinstance(packet, ColorPacket):
                MODE = 2
                # Set all the pixels to one color and stay there.
                pixels.fill(packet.color)
                pixels.show()
            # Control Pad Functionality
            elif isinstance(packet, ButtonPacket):
                if packet.pressed:
                    if packet.button == ButtonPacket.BUTTON_1:
                        MODE = 1
                        animations.activate(1)
                    elif packet.button == ButtonPacket.BUTTON_2:
                        MODE = 1
                        animations.activate(2)
                    elif packet.button == ButtonPacket.BUTTON_3:
                        MODE = 1
                        animations.activate(3)
                    elif packet.button == ButtonPacket.BUTTON_4:
                        MODE = 1
                        animations.activate(4)
                    # change the mode with right arrow
                    elif packet.button == ButtonPacket.RIGHT:
                        MODE = 1
                        animations.next()
                    elif packet.button == ButtonPacket.LEFT:
                        MODE = 4
                        off.animate()
                    #change the brightness with up and down arrows
                    elif packet.button == ButtonPacket.UP:
                        pixels.brightness = pixels.brightness + 0.1
                        pixels.show()
                        if pixels.brightness > 1:
                            pixels.brightness = 1
                    elif packet.button == ButtonPacket.DOWN:
                        pixels.brightness = pixels.brightness - 0.1
                        pixels.show()
                        if pixels.brightness < 0.1:
                            pixels.brightness = 0.1

                if MODE == 1:
                    animations.animate()
                if MODE == 4:
                    animations.freeze()

```

Customizing Your Code

The sample code comes with seven different animations:

1. **Reading Light:** a warm yellowish white light
2. **Fire** - a flickering torch light effect
3. **Rainbow** - a spread-out animated, shifting rainbow
4. **Rainbow Fade** - The entire strip fades very slowly through the spectrum
5. **Starry Night** - a pulsing dark blue with a white comet
6. **Bright White:** Lights on white at 50% brightness (you can make this brighter but watch your power consumption)
7. **Off**

You can add more animations or reorder them however you'd like. Check out the [CircuitPython LED Animations Guide \(https://adafru.it/LZF\)](https://adafru.it/LZF) to see what other animations this library offers. There are several different Rainbow animations, Pulse, Chase, and some Sparkle variations that are a lot of fun to play with.

Adding a New Animation

You'll need to change the code in three or four places:

1. Import the animation at the very top. This defines which animation you're using (i.e. Rainbow, Pulse, Sparkle, etc).
2. Create a definition for each mode under "Animation Setup". Here's where you decide what kind of Pulse or Sparkle you want to make (color, speed, etc).
3. Add your mode to the Animations playlist. This determines which animation plays first, second, etc.
4. For your top four favorite modes, tell the mode to trigger with one of the four numbered buttons in the BlueFruit app so you can jump straight to it.

1. Import the Animation

Near the top of the code you'll see a few lines importing various animations from the LED Animations Library. Each animation only needs to be imported once. If you want to use one of the animations we aren't using in the sample code ([see them all here \(https://adafru.it/LZF\)](https://adafru.it/LZF)), you'll need to import the one you want in this section.

```
from adafruit_led_animation.animation.solid import Solid
from adafruit_led_animation.animation.comet import Comet
from adafruit_led_animation.animation.rainbow import Rainbow
```

2. Create a Definition for each Animation Mode

This is where you decide what solid colors you'd like, or how spread out your rainbows should be, or how quickly they move. You can define as many animations as you want, including multiples of each type. It's fine to have a slow, spread-out rainbow and also a fast and tight rainbow. Just call them by different names.

You can declare colors in a variety of ways including RGB tuples, hex codes, and HSV, or import from a selection of predefined colors up near the top of the code in the import section.

You can also group animations together so they play at the same time. Look at the "fire" and "starry night" animations for examples of this.

One more thing to note is the "startup" animation. This is sort of a meta-animation, which calls the "powerup" animation I defined a little earlier, and runs it just once, using the `AnimateOnce` feature.

```
readingLight = Solid(pixels, color=0xFF7D13) #warm white color HEX code
brightWhite = Solid(pixels, color=(150, 150, 150))
rainbow = Rainbow(pixels, speed=0.1, period=10, step=0.5)
rainbowfade = RainbowFade(pixels, speed=0.4, name="rainbowfade")
powerup = RainbowComet(pixels, speed=0, tail_length=50, bounce=False)
off = Solid(pixels, color=BLACK)

#startup animation will play just once
startup = AnimateOnce(powerup)

#starrynight and fire are animation groups with layered effects.
starrynight = AnimationGroup(
    SparklePulse(pixels, speed=0.01, color=(0, 0, 150), period=1),
    Comet(pixels, speed=0, tail_length=8, color=(150, 150, 150), bounce=False),)

fire = AnimationGroup(
    Comet(pixels, speed=0, tail_length=1, color=BLACK),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=AMBER),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=RED),
    Sparkle(pixels, speed=0.05, num_sparkles=20, color=ORANGE),
    Sparkle(pixels, speed=0.05, num_sparkles=5, color=0xFF7D13),
    Sparkle(pixels, speed=0.05, num_sparkles=10, color=BLACK),
)
```

3. Animations Playlist

Under the definitions you'll see an Animations Playlist section. The right arrow in the Bluetooth app will scroll through all the animations in this list, in the order you list them. You can define animations right in the playlist if you want, or just call the animations you defined earlier.

```
# Here is the animation playlist where you set the order of modes

animations = AnimationSequence(
    readingLight,
    fire,
    rainbow,
    starrynight,
    rainbowfade,
    brightWhite,
    auto_clear=True,
)
```

4. Customize the Buttons

Finally, you can pre-program the numbered 1-4 buttons in the app to play specific animations - so you can always jump to your favorites. Just change the number of the animation within the playlist. Remember that the first animation (readingLight in our case) is numbered 0, and so these 4 buttons will play fire, rainbow, starry night, and rainbow fade since I've assigned them to 1, 2, 3, and 4.

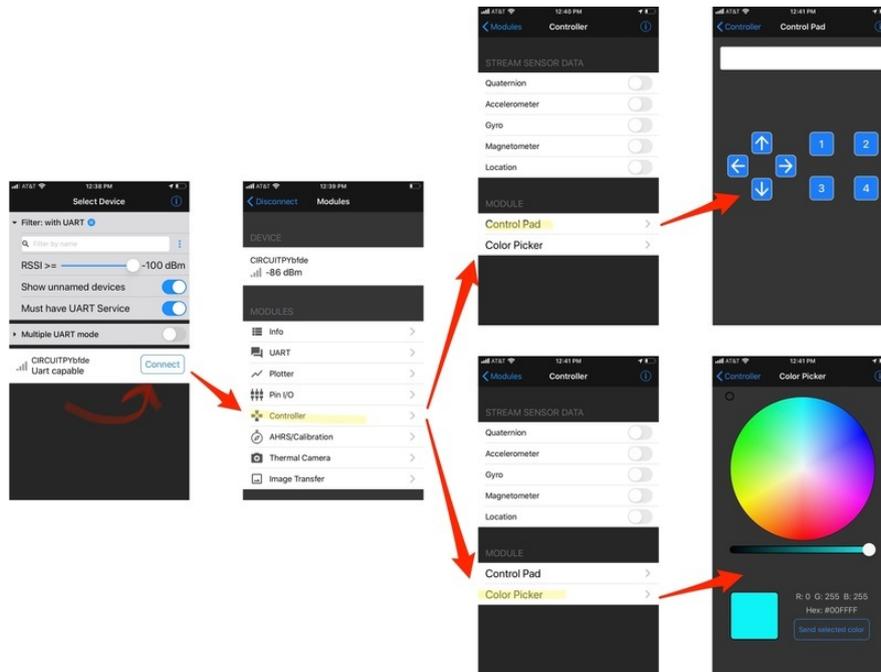
```
elif isinstance(packet, ButtonPacket):
    if packet.pressed:
        if packet.button == ButtonPacket.BUTTON_1:
            MODE = 1
            animations.activate(1)
        elif packet.button == ButtonPacket.BUTTON_2:
            MODE = 1
            animations.activate(2)
        elif packet.button == ButtonPacket.BUTTON_3:
            MODE = 1
            animations.activate(3)
        elif packet.button == ButtonPacket.BUTTON_4:
            MODE = 1
            animations.activate(4)
        # change the mode with right arrow
        elif packet.button == ButtonPacket.RIGHT:
            MODE = 1
            animations.next()
        elif packet.button == ButtonPacket.LEFT:
            MODE = 4
            off.animate()
```

The right arrow scrolls through all the modes, stepping forward. There's no easy way to scroll backwards through the modes, so I've made the left arrow into the "off" button -- always nice to have within one tap.

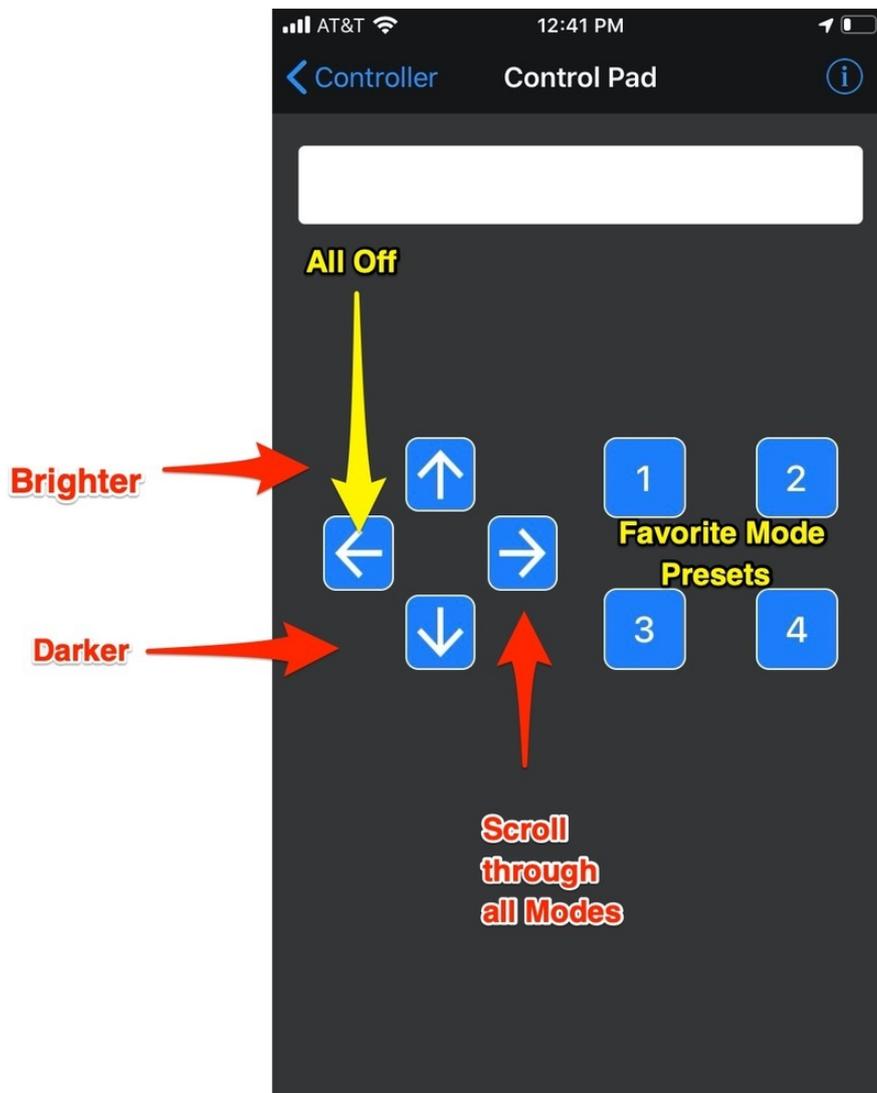
BlueFruit App Controls

Go to the Apple or Android store and find the **Adafruit BlueFruit App**. Download and install it on your phone or tablet.

If you're having trouble, head over to the [Bluefruit LE Connect Guide \(https://adafru.it/DNc\)](https://adafru.it/DNc) for more detailed instructions.



Open the app and select your board. Click Connect, then select Controller. Choose Control Pad to access your pre-programmed buttons or Color Picker to choose and send a solid color to your lights.



The up and down arrows will adjust the brightness of the entire strip. The right arrow scrolls through all your modes, and the left arrow turns the strip off.

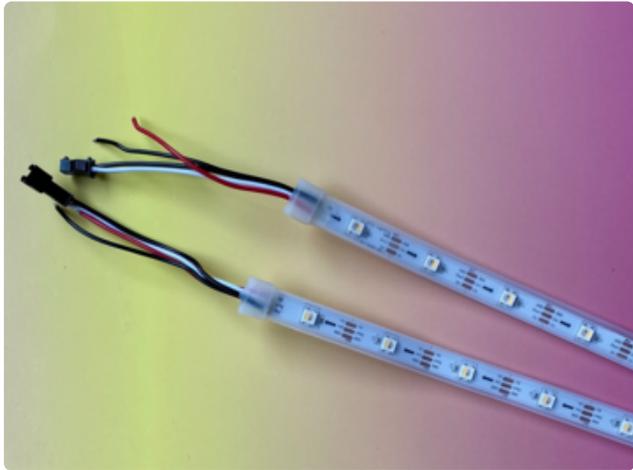
The 1-4 buttons activate your favorite modes. Our sample code works as follows:

1. Reading Light
2. Fire
3. Rainbow
4. Rainbow Fade

These buttons can be programmed to shortcut to your favorite modes - see the "Customizing Your Code" page to set up your favorites.

Electronics Assembly

Up To 250 NeoPixels



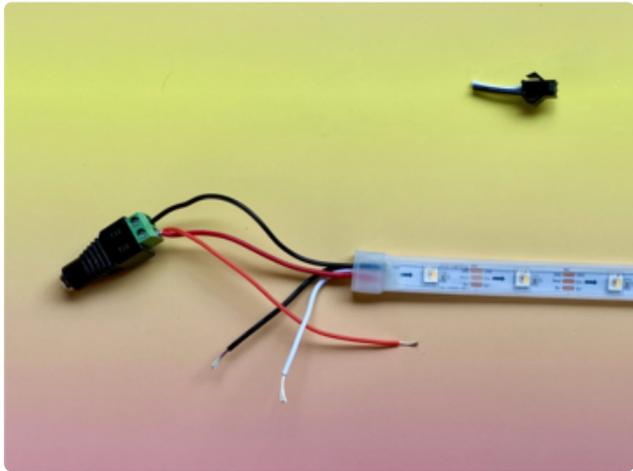
Your NeoPixel strip is directional: it has an IN end and an OUT end.

You can connect power to either or both ends, but you must connect data (the white wire) to the IN end for the pixels to work properly.

Take a look at your strip. There are arrows pointing from IN to OUT. Find the end with the arrows pointing away (down the strip). You'll make your connections to this end.



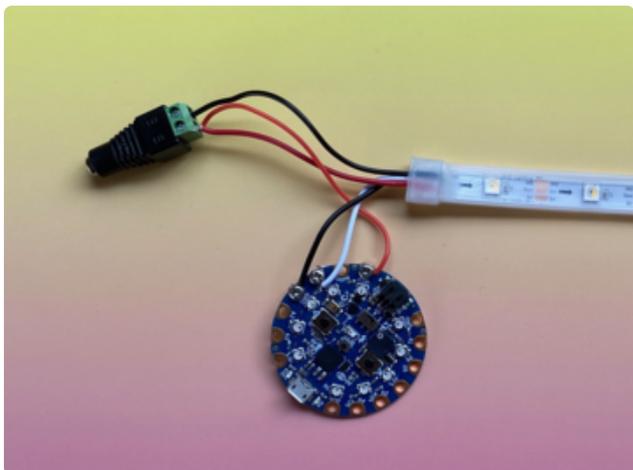
Cut a piece of red wire about 2-3 inches long. Use your wire strippers to strip about 1/4" of shielding from both ends.



Strip a little more shielding from the red and black wires coming from your NeoPixel strip's IN end. Twist the red wire you just cut together with the red wire coming from the strip.

Open the ports on the screw terminal all the way with your screwdriver. Push both red wires into the + side and tighten the screw. Push one black wire into the - side and tighten.

Tug on the wires to make sure you have a very secure connection.



Connect the remaining red, white, and black wires to the Circuit Playground as shown:

Red to **VOUT**

White to **A1**

Black to **GND**

You can screw them into place with your bolt-on kit, or solder them with your soldering iron if you prefer. Either way, be sure you have a nice tight metal-on-metal connection with no stray wires promiscuously touching neighboring pads.

More Than 250 NeoPixels

After around 250 pixels the 4A power supply attached to one end of the strip will become insufficient. The copper pads that pass power and data along have a small amount of resistance, and that resistance adds up after a while. What this means is that each subsequent pixel gets a little less power until the ones at the very end are dim. This is called "brown-out."



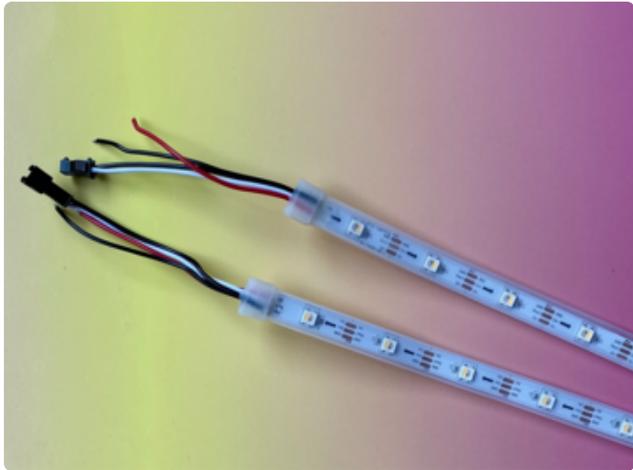
In this photo I have my power connected to just one end of a 4 meter strip of lights, with the lights on full brightness. You can see how the outside of the spool is an almost bluish white, and the inside of the reel gets increasingly brown in color. This is how you know you need more power.

Why This Happens

There are actually 3 LEDs inside each NeoPixel: a green, red, and blue LED. If you turn your strip to show bright white, the NeoPixel strip "builds" this color by turning on all 3 LEDs as bright as they will go. This means that full-bright white uses three times the power that full-bright red or blue or green uses.

My single power connection works fine if I make all the lights blue. But for a bedroom light, I want the ability to turn up the lights to a bright white, in case I need to vacuum, or wake somebody up, or something. Adding power at the other end of the strip is a great way to do this.

For even larger installations, you can connect power to the middle of the strip as well. This is called "injecting" power. It's a good idea to inject power every 250 lights or so if you want the option of a steady bright white light.



Adafruit helpfully attaches wires to both end of the NeoPixel strips, so it's easy to add a second screw terminal and a power splitter to the other end, in the same fashion as above.

You will also want to upgrade to a beefier 10A power supply if you have a whole lot of lights. Get a splitter and an extension cable or two as well if the two ends of your strip aren't within easy reach of each other.

Do NOT connect the two different ends to two different outlets using two different power supplies. This will cause all kinds of problems and can damage your board.



[5V 10A switching power supply](https://www.adafruit.com/product/658)

This is a beefy switching supply, for when you need a lot of power! It can supply 5V DC up to 10 Amps, running from 110V or 220V power (the plug it comes with is for US/Canada/Japan...

<https://www.adafruit.com/product/658>



[2.1mm female/male barrel jack extension cable](https://www.adafruit.com/product/327)

This product is a 3-in-1 cable. By itself, it can act as an extension cord for any 5.5/2.1mm DC power cable. The heavy 24 AWG wires can handle up to 5A. You can also cut...

<https://www.adafruit.com/product/327>



2-Way 2.1mm DC Barrel Jack Splitter Squid

Double up your power adapters with this wonderfully simple 2-way power jack splitter. All you have to do is connect a standard 5.5mm/2.1mm sized plug into one end, and behold, two...

<https://www.adafruit.com/product/1351>

Troubleshooting

Q: My lights are flickering on and off with some of the animations. How do I fix this?

A. You may have a less-than-optimal data wire connection. Try readjusting your bolt and nut so there's more metal-on-metal connection, or upgrade to a soldered connection which will pass the data much more cleanly.

If that doesn't fix it, try using a different screw terminal. Sometimes the internal workings don't line up as well as they should and it's not that uncommon to get a "bad" one.

Q: My lights come on for a minute then click off again and my board resets. What's going on?

A. You may have insufficient power. Try using a larger 10A power supply and connecting to both ends of the strip. Or try turning the brightness down in the code. One more thing to try is to temporarily "shorten" your LED strip in the code (set `NUM_LEDS` to 10 or 20 instead of your full count) to see if that fixes things.

Q: I'm getting really fast flickering with inconsistent color.

A: Your ground connection is probably loose. These screw terminals can be really fiddly and sometimes don't give you a great connection. Try reconnecting the wires, making sure you've got enough shielding stripped off before shoving the end into the port.

Q: My lights aren't lighting up at all.

A: Are there lights on the face of your Circuit Playground? If the board is getting power but the lights aren't coming on, it's either a problem with the wire connections coming from the strip, or a problem with the code. Check to be sure you've connected to the correct pads (**A1**, **VOUT**, and **G**). Double check to be sure you have all the correct library files installed, and try re-uploading your code.

If the board is not getting power either, it may be a problem with your power supply. Check your red (power) wire connection at the screw terminal.

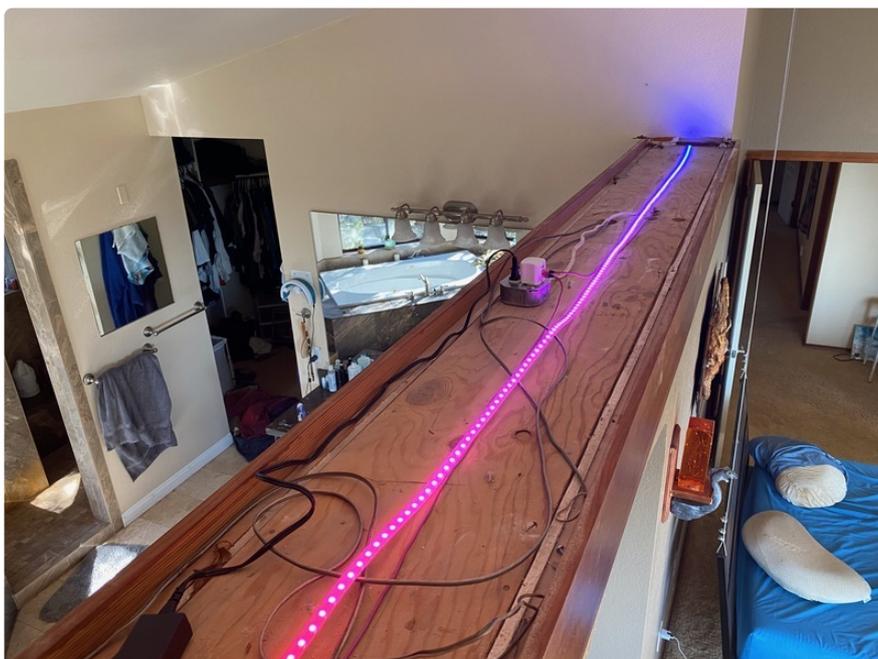
Q: I've got another problem I need help with. Where can I get support?

A: [We've got lots more troubleshooting ideas on this page \(https://adafru.it/Den\)](https://adafru.it/Den). Adafruit also has a Discord Community where there are experts available to help you with your build. [Come join our community! \(https://adafru.it/NzB\)](https://adafru.it/NzB)

Installation

Your installation setup will depend on your room. It's easiest to place it on top of a wall or in an alcove so it points up at the ceiling, above eye level, to shine the most light around the room. Indirect light looks the best, so coming up with a creative solution to point the light away from you, at a white or light-colored wall, will give you the best results.

We put ours above a doorway arch between rooms, and now we have beautiful colored light modes in both our bedroom and master bath.



If you're installing someplace that's hard to reach, pull power from a switched outlet in your room. This way you can still control on/off with a normal light switch, and if your Circuit Playground ever needs to be rebooted (which will happen from time to time) you can reboot it just by turning the light switch off and on.

We installed a [fancy light switch \(https://adafru.it/NC8\)](https://adafru.it/NC8) that has a USB port included on the switch plate. Then we took an old outdated iPhone and mounted it on the wall, plugged in to power with the USB port. This old iPhone is now our dedicated room controller - we can leave the iPhone "on" all the time and it becomes our fancy light switch and Bluetooth music controller. DIY Home Automation FTW!



We also tried putting these lights in an RV and found plenty of easy-to-reach ledges where indirect lighting worked like a dream! Trick out your camper, and impress your neighbors at the rally with your own custom lighting modes.