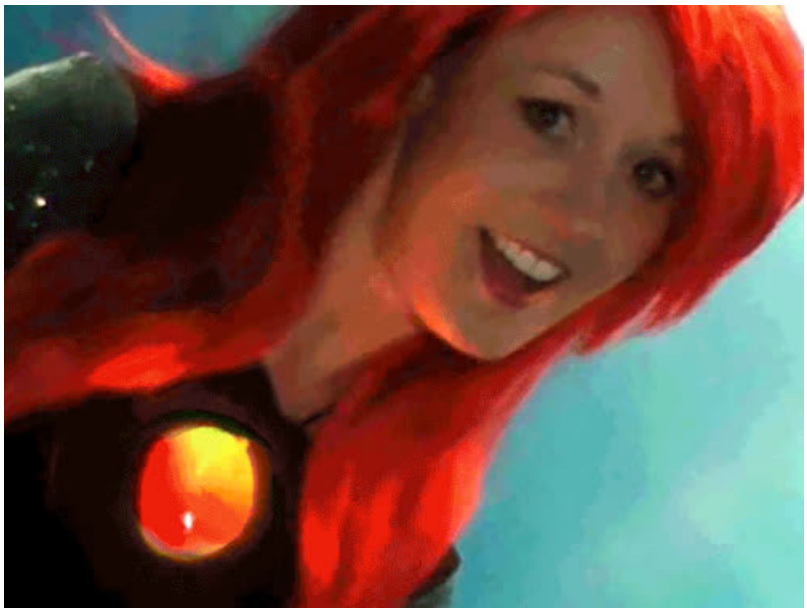
adafruit learning system

Circuit Playground Seashell Pendant

Created by Erin St Blaine



Last updated on 2018-08-22 04:00:39 PM UTC

Guide Contents

Introduction

Make a gorgeous light up necklace using Circuit Playground and a real polished seashell. The onboard push button makes it easy to choose between color palettes, to quickly match your outfit or your mood.

There's also a button for brightness control: Turn it up to light up your sarong style at an afternoon beach barbecue, or turn it down to set off your evening gown in a romantic restaurant.

We've even included a sound reactive mode using Circuit Playground's onboard microphone. When you talk or sing, your necklace will glow along with your voice just like Ariel the Little Mermaid.

A polished seashell makes the perfect diffusion material, but this build will work with a huge variety of materials.

This is a great beginner project with just a little (optional) soldering. This project is perfect for kids, teenagers who need prom accessories, or for those of us who simply refuse to grow up.

If you're just getting started, get Lady Ada's Electronics Toolkit! (https://adafru.it/fE3)

If you just need a soldering iron, try this one (https://adafru.it/ide).

Materials

1 x Circuit Playground
Circuit Playground Classic
ADD TO CART

1 x On/Off Switch
SPDT Slide Switch
ADD TO CART

1 x Battery Cable
JST Extension Cable
ADD TO CART

1 x Battery
LiPoly 500 mAh Battery
ADD TO CART

1 x Battery Charger
USB LiPoly Battery Charger
ADD TO CART

Additional Parts

- 3d Printed Circuit Playground case
- Seashell or other diffusion material
- Jewelry wire
- Necklace cord

Code

Before You Start

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

You'll only need to do all this once, so be a little patient if it seems like a lot!

FastLED Library

You will also need to install the **FastLED** library in Arduino (`Sketch > Include Library > Manage Libraries...`)

One other note: if you're using **FastLED** with Circuit Playground, be sure to `#include` the Circuit Playground library FIRST and the **FastLED** library second, or you may run into problems.

Upload Code

Once you've got everything installed and your computer can talk to the Circuit Playground, it's time to upload the code.

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

Copy and paste this code into a new Arduino window and click "upload".

```
#include <Adafruit_CircuitPlayground.h>
#include <FastLED.h>

// tell FastLEED all about the Circuit Playground's layout

#define CP_PIN      17   //circuit playground's neopixels live on pin 17
#define NUM_LEDS      10   // number of neopixels on the circuit playground
#define COLOR_ORDER GRB

// The right button will control the neopxiels' brightness.  Use these
// varables to control how many brightness levels you'd like.

uint8_t brightness = 30;  // initial brightness level
uint8_t minbrightness = 10;  // minimum brightness level
uint8_t maxbrightness = 180;  //maximum brightness level -- can be set as high as 255
uint8_t brightnessint = 40;  //amount of brightness added for each button press.

int STEPS = 10;  //makes the rainbow colors more or less spread out
int NUM_MODES = 5;  // change this number if you add or subtract modes

CRGB leds[NUM_LEDS];      // set up an LED array

CRGBPalette16 currentPalette;
TBlendType    currentBlending;

int ledMode = 0;       //Initial mode
bool leftButtonPressed;
bool rightButtonPressed;
```

```
// SOUND REACTIVE SETUP --------------
// based on Adafruit's VU Meter code: https://learn.adafruit.com/led-ampli-tie/overview?view=all#the-code

#define MIC_PIN         A4                                      // Analog port for microphone
#define DC_OFFSET  0                                            // DC offset in mic signal - if unusure, le
                                                                // I calculated this value by serialprintln
#define NOISE     200                                           // Noise/hum/interference in mic signal and
#define SAMPLES   60                                            // Length of buffer for dynamic level adjus
#define TOP (NUM_LEDS + 2)                                      // Allow dot to go slightly off scale
#define PEAK_FALL 10                                            // Rate of peak falling dot

byte
  peak      = 0,                                                // Used for falling dot
  dotCount  = 0,                                                // Frame counter for delaying dot-falling s
  volCount  = 0;                                                // Frame counter for storing past volume da
int
  vol[SAMPLES],                                                 // Collection of prior volume samples
  lvl       = 10,                                               // Current audio level, change this number
  minLvlAvg = 0,                                                // For dynamic adjustment of graph low & hi
  maxLvlAvg = 512;


// SETUP ---------------------------------------------------
void setup() {
  Serial.begin(57600);
  CircuitPlayground.begin();
  FastLED.addLeds<WS2812B, CP_PIN, COLOR_ORDER>(leds, 10).setCorrection( TypicalLEDStrip );
  currentBlending = LINEARBLEND;
  set_max_power_in_volts_and_milliamps(5, 500);               // FastLED 2.1 Power management set at 5V,


}

void loop()  {

  leftButtonPressed = CircuitPlayground.leftButton();
  rightButtonPressed = CircuitPlayground.rightButton();

  if (leftButtonPressed) {  //left button cycles through modes
    clearpixels();
    ledMode=ledMode+1;
    delay(300);
    if (ledMode > NUM_MODES){
    ledMode=0;
     }
  }
   if (rightButtonPressed) {   // brightness control button
     brightness = brightness + brightnessint;
     delay(300);
     if (brightness > maxbrightness) {
      brightness=minbrightness;
     }

   }

 switch (ledMode) {
       case 0: currentPalette = RainbowColors_p; rainbow(); break;
       case 1: currentPalette = OceanColors_p; rainbow(); break;
       case 2: currentPalette = LavaColors_p; rainbow(); break;
```

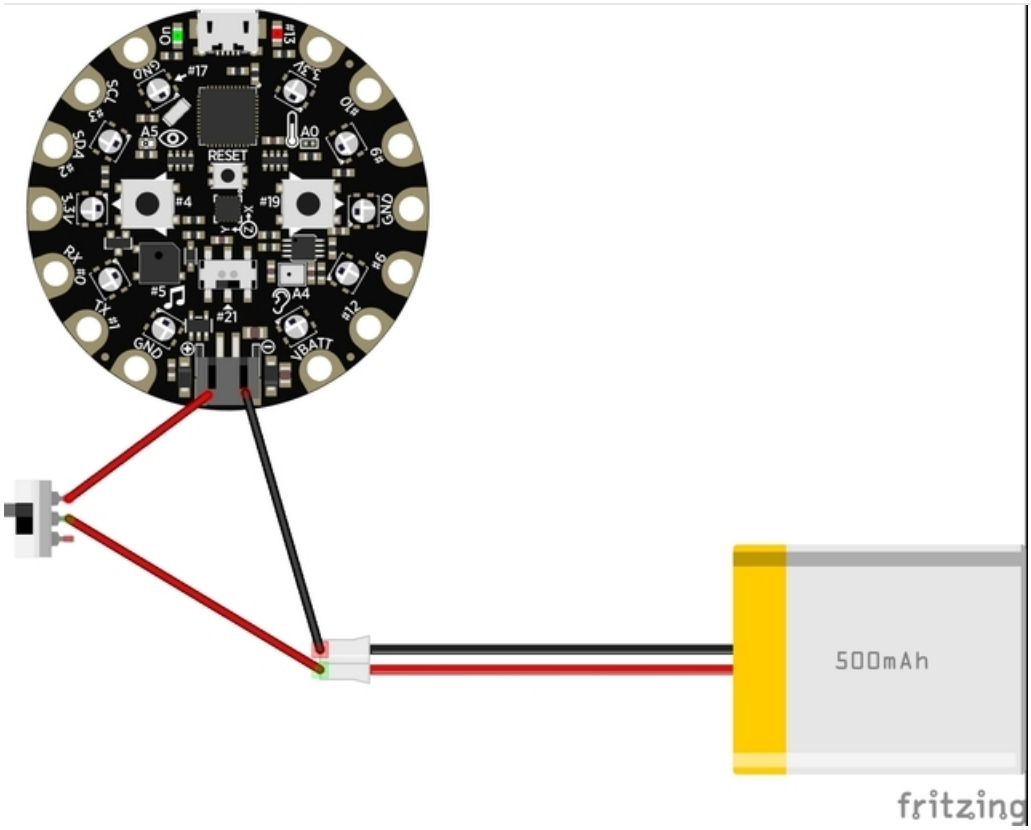
```
       case 3: currentPalette = ForestColors_p; rainbow(); break;
       case 4: currentPalette = PartyColors_p; rainbow(); break;
       case 5: soundreactive(); break;
       case 99: clearpixels(); break;

}
}
void clearpixels()
{
  CircuitPlayground.clearPixels();
   for( int i = 0; i < NUM_LEDS; i++) {
   leds[i]= CRGB::Black;
    }
   FastLED.show();
}

void rainbow()
{

  static uint8_t startIndex = 0;
  startIndex = startIndex + 1; /* motion speed */

  FillLEDsFromPaletteColors( startIndex);

  FastLED.show();
  FastLED.delay(20);}

//this bit is in every palette mode, needs to be in there just once
void FillLEDsFromPaletteColors( uint8_t colorIndex)
{
  for( int i = 0; i < NUM_LEDS; i++) {
    leds[i] = ColorFromPalette( currentPalette, colorIndex, brightness, currentBlending);
    colorIndex += STEPS;
  }
}


void soundreactive() {

  uint8_t  i;
  uint16_t minLvl, maxLvl;
  int      n, height;

  n = analogRead(MIC_PIN);                                  // Raw reading from mic
  n = abs(n - 512 - DC_OFFSET);                             // Center on zero

  n = (n <= NOISE) ? 0 : (n - NOISE);                       // Remove noise/hum
  lvl = ((lvl * 7) + n) >> 3;                               // "Dampened" reading (else looks twitchy)

  // Calculate bar height based on dynamic min/max levels (fixed point):
  height = TOP * (lvl - minLvlAvg) / (long)(maxLvlAvg - minLvlAvg);

  if (height < 0L)       height = 0;                        // Clip output
  else if (height > TOP) height = TOP;
  if (height > peak)     peak   = height;                   // Keep 'peak' dot at top


  // Color pixels based on rainbow gradient
  for (i=0; i<NUM_LEDS; i++) {
    if (i >= height)   leds[i].setRGB( 0, 0,0);
```

```
    else leds[i] = CHSV(map(i,0,NUM_LEDS-1,20,70), 255, 255);  //constrained to yellow color, change CHSV
  }

  // Draw peak dot  -- circuit playground
  if (peak > 0 && peak <= NUM_LEDS-1) leds[peak] = CHSV(map(peak,0,NUM_LEDS-1,20,70), 255, 255);

// Every few frames, make the peak pixel drop by 1:

    if (++dotCount >= PEAK_FALL) {                            // fall rate
      if(peak > 0) peak--;
      dotCount = 0;
    }

  vol[volCount] = n;                                          // Save sample for dynamic leveling
  if (++volCount >= SAMPLES) volCount = 0;                    // Advance/rollover sample counter

  // Get volume range of prior frames
  minLvl = maxLvl = vol[0];
  for (i=1; i<SAMPLES; i++) {
    if (vol[i] < minLvl)      minLvl = vol[i];
    else if (vol[i] > maxLvl) maxLvl = vol[i];
  }
  // minLvl and maxLvl indicate the volume range over prior frames, used
  // for vertically scaling the output graph (so it looks interesting
  // regardless of volume level).  If they're too close together though
  // (e.g. at very low volume levels) the graph becomes super coarse
  // and 'jumpy'...so keep some minimum distance between them (this
  // also lets the graph go to zero when no sound is playing):
  if((maxLvl - minLvl) < TOP) maxLvl = minLvl + TOP;
  minLvlAvg = (minLvlAvg * 63 + minLvl) >> 6;                 // Dampen min/max levels
  maxLvlAvg = (maxLvlAvg * 63 + maxLvl) >> 6;                 // (fake rolling average)


show_at_max_brightness_for_power();                          // Power managed FastLED display
  Serial.println(LEDS.getFPS());
}
```

If all goes well, the lights on the Circuit Playground will come on. Press the right side button to cycle through brightness modes. Press the left side button to toggle between color modes and the sound reactive mode.

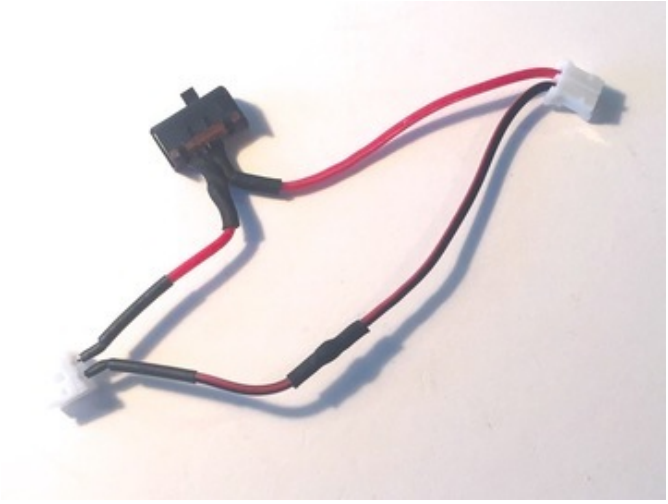
Add your own modes or customize the modes that are already there. If you want to add your own color palettes, check out this Neopixel Parasol guide (https://adafru.it/wEY) for ideas and instructions on how to do that with FastLED.

Wiring



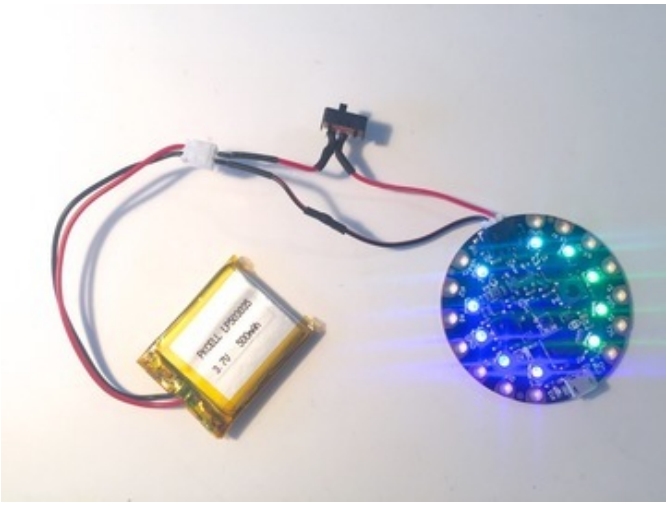
The Circuit Playground already comes with everything we need except an on/off switch. If you're making this project with kids and don't want to do any soldering, you can skip this part and just plug the battery directly into the circuit playground. You'll turn the necklace on and off by unplugging the battery.

I've found that having a switch makes the necklace much more user-friendly, and makes your battery last longer since you can turn it off when nobody is looking.



Cut both ends off your battery cable and trim off one of the outside legs on your on/off switch.

Solder the black wires of the battery cable back together (it's just shorter now). Solder one red wire to each of the two remaining legs on the on/off switch. You can solder either wire to either leg. The way the switch works is that when the button is centered between the two connected legs, it connects them and power flows. When it's centered above just one leg, power does not flow.



3d Printing

This is remixed from the Circuit Playground Yo-Yo case (https://adafru.it/wEO) by the Ruiz Brothers. Print the case at 100% infill in either ABS or PLA. If you don't have a 3d printer, you can use a service like Shapeways and they'll print one for you and send it over.

https://adafru.it/wEP

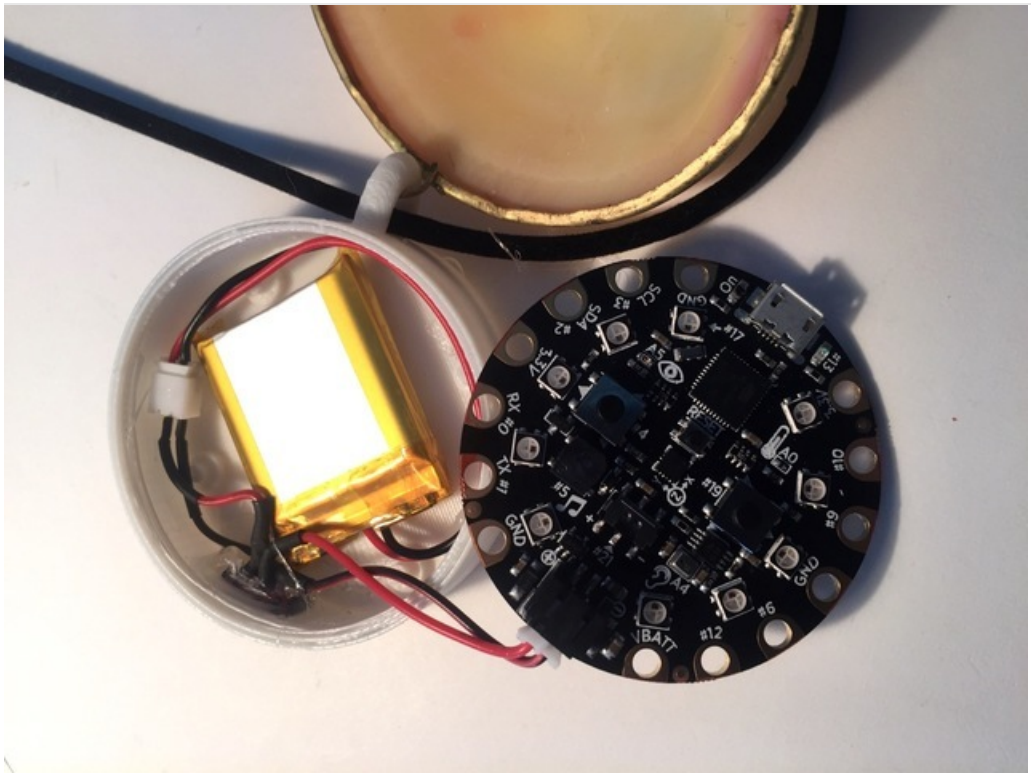
https://adafru.it/wEP

https://adafru.it/wEQ

https://adafru.it/wEQ

The case has a loop near the top where you can thread your necklace cord and a small slot at the bottom for your on/off switch.

Final Assembly



Glue the switch into the slot in the 3d printed case. Plug in the battery and the Circuit Playground and fit them inside the case as well. Screw the lid onto the case to hold them in place.

The final build is up to you. I used half of a polished Seashell Trinket Box (https://adafru.it/wEM). It already had a gold wire around the edge of the shell and already had a loop for attaching to the 3d printed case.

If your shell doesn't already have an attachment point, there are a lot of ways to attach the pendant front. Drilling holes in the 3d printed case and the shell and running jewelry wire through as a hinge works great.

Here are a few photos of some of the materials I tried. Post photos of yours in the forum!