Ursula's Seashell Necklace

Created by Erin St Blaine

https://learn.adafruit.com/ursulas-seashell-necklace

Last updated on 2021-11-15 06:25:52 PM EST

Table of Contents

Overview

Look at this shell. Isn't it neat?
Wouldn't it just make your cosplay complete?
Wouldn't they think you're the girl,
The girl that has... everything?

Wear this necklace and it will respond to your beautiful Mermaid voice whenever you talk or sing. Use it to find your Prince, or to steal someone else's Prince if that's your bent. (Hey, it almost worked for Ursula. Ultimate Power Awaits.)

Electronics

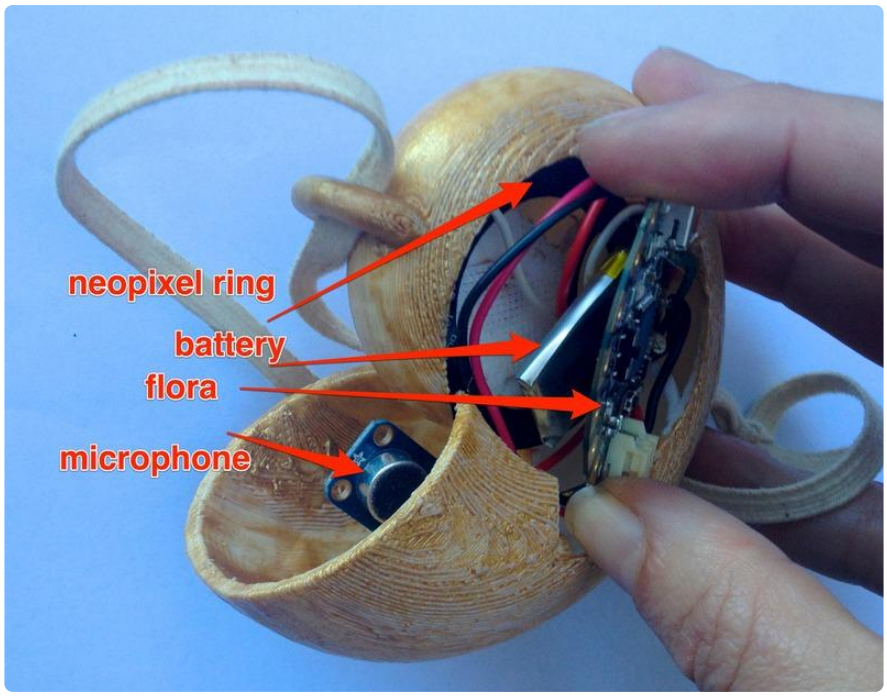
- Flora (https://adafru.it/dVl)
- 16 pixel Neopixel Ring (https://adafru.it/dyU)
- 150mAh LiPoly battery (https://adafru.it/dYY)
- Electret Microphone breakout (https://adafru.it/eQw)
- Micro Lipo charger for the battery (http://adafru.it/1304)

Tools

- 3D Printer
- Soldering iron & accessories
- Teensy weensy Philips head screwdriver

Crafty Stuff

- Translucent gold paint (I used Pebeo Vitrail Gold (https://adafru.it/eQx))
- Elastic or necklace cord



If you like this guide, also check out this Seashell Necklace tutorial (https://adafru.it/xbc) for a new twist on this project using our Circuit Playground board.

3D Printing

Print the seashell in white ABS. Get as much of the raft and supports as possible out from the inside -- you don't want that stuff blocking your lights.

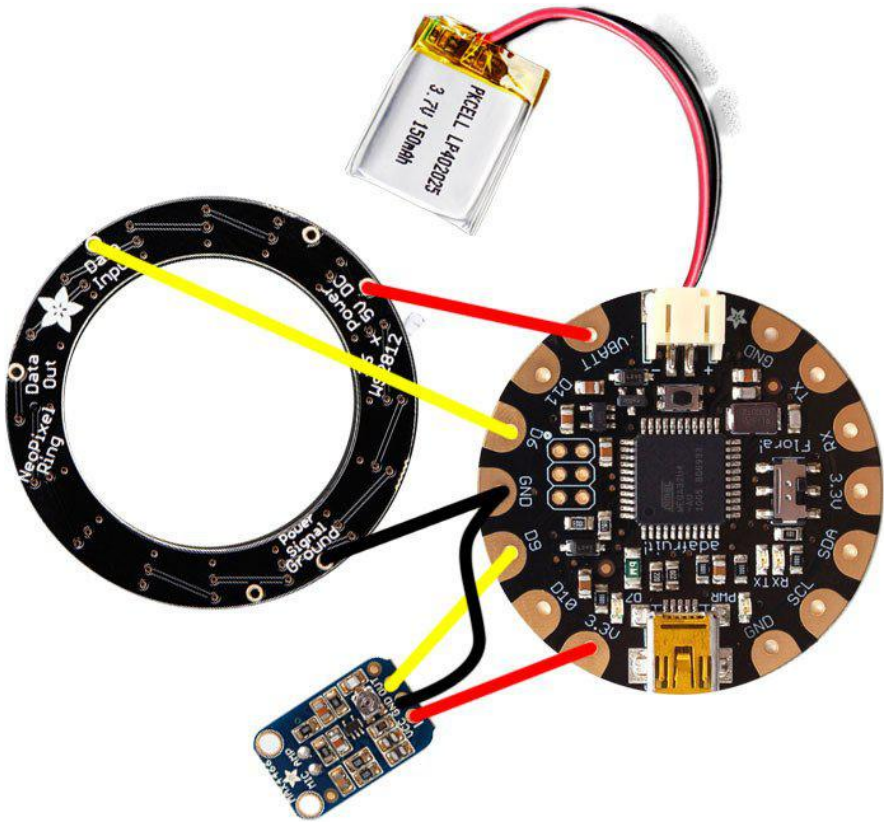
Once it's done, paint it with gold or yellow translucent paint. I used two coats of Pebeo Vitriol glass paint (https://adafru.it/eQx) which gave it an almost carved-wood look, but any not-too-opaque paint should work.

Download ursula_seashell.stl

https://adafru.it/eUa

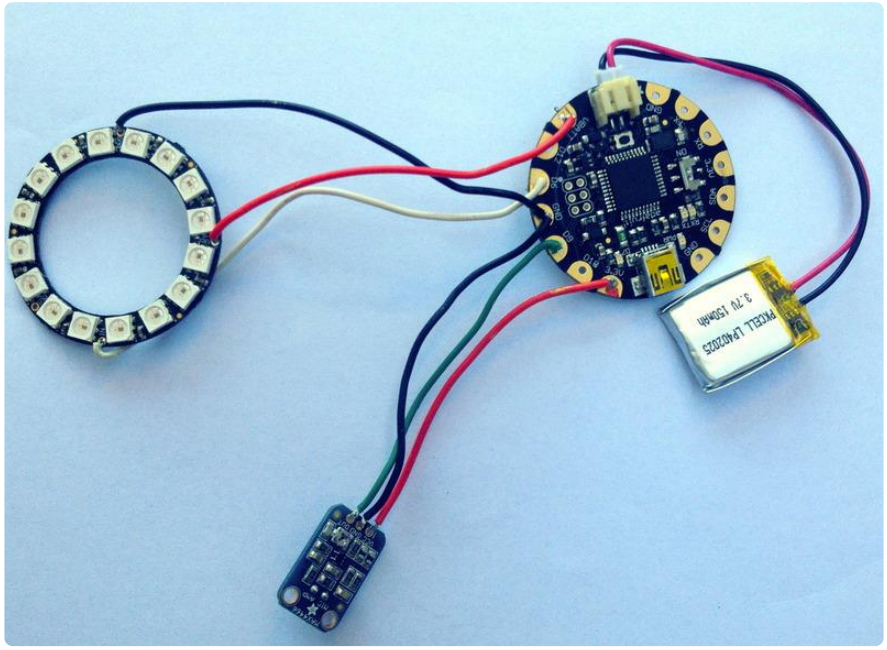
If you don't have a 3D printer, you can get one printed at Shapeways (https://adafru.it/eUb). The internet is so great.

Circuit Diagram



- Flora VBAT --> Neopixel 5V
- Flora D6 --> Neopixel Data Input
- Flora GND --> Neopixel Ground
- Flora GND --> Microphone GND
- Flora D9 --> Microphone Out
- Flora 3.3V --> Microphone VCC

Assembly



Cut all your wires to about 3" -- this will ensure you've got enough wiggle room to get everything situated in the shell but not have excess wire hanging around.

Stranded (flexible) wire will work best since you may need to flex the components around a bit to get them situated in the seashell.

On the neopixel ring, it's easiest to stick the wire in from the front and solder on the back to ensure you don't melt the LEDs or break anything.

Notice that there are two wires going to Flora's GND pad, so it will work best to twist them together at the ends and solder them in at the same time.

The Code

Software Setup

If this is your first time using Flora, take a look at Getting Started with Flora (https://adafru.it/dwi) to get a guided tour.

Once you've got Flora up and running with Arduino (https://adafru.it/dgt), you'll need to install the FastLED library in Arduino (`Sketch > Include Library > Manage Libraries...`)

Libraries? Why? What's a Library?

In a nutshell, Arduino libraries have a lot of pre-written functions that make your neopixels easy to command. You can do fancy stuff without being a code guru. Yay Libraries!

FastLED is a fast, efficient, easy-to-use Arduino library for programming addressable LED strips and pixels. It has a lot of features to get your animations up and running fast -- and it has a lot of code samples available if you're just learning to code.

All about Arduino Libraries (https://adafru.it/dNR) will tell you everything you ever wanted to know about libraries, including more detailed installation instructions.

Once your curiosity is satiated and your library is installed, copy and paste the code into your Arduino window.

Go to your Tools menu and select Flora from the list of boards. Plug your Flora into your computer via the onboard USB port. Press the "reset" button on your Flora and wait for the blinky red light, then click the upload button in Arduino.

```
/* soundbracelet for FastLED 2.1 or greater

Converted by: Andrew Tuline

Date: Oct, 2014

this code is based Neopixel code by John Burroughs:

https://www.youtube.com/watch?v=JjX8X5D8RW0&amp;feature=youtu.be
https://plus.google.com/105445034001275025240/posts/jK2fxRx79kj
http://www.slickstreamer.info/2014/07/led-bracelet-vu-meter-3dprinting.html

That was based on the Adafruit LED Ampli-tie project at:

https://learn.adafruit.com/led-ampli-tie/overview

This was written for a Sparkfun INMP401 MEMS microphone/pre-amp. In this case, it's
plugged into A5 of the Arduino.

You may need to only use battery power as noise from the USB can affect the signals.

Plug Vcc of the microphone into 3.3V of Arduino. Connect 3.3V of Arduino to aref
pin, and gnd to gnd.

FastLED is available at https://github.com/FastLED/FastLED

*/


#include "FastLED.h"                                          // FastLED library.
Preferably the latest copy of FastLED 2.1.

// Fixed definitions cannot change on the fly.
#define LED_DT 6                                             // Serial data pin for
WS2812B or WS2801
```

```
#define COLOR_ORDER GRB                                         // Are they RGB, GRB
or what??
#define LED_TYPE WS2812B                                         // What kind of
strip are you using?
#define NUM_LEDS 16                                             // Number of LED's

// Initialize changeable global variables.
uint8_t max_bright = 255;                                       // Overall brightness
definition. It can be changed on the fly.

struct CRGB leds[NUM_LEDS];                                     // Initialize our LED
array.


#define MIC_PIN 9                                               // Analog port for
microphone

#define DC_OFFSET  0                                           // DC offset in mic
signal - if unusure, leave 0
                                                                // I calculated this
value by serialprintln lots of mic values
#define NOISE     100                                           // Noise/hum/
interference in mic signal and increased value until it went quiet
#define SAMPLES   60                                            // Length of buffer
for dynamic level adjustment
#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 speed
  volCount  = 0;                                                // Frame counter for
storing past volume data
int
  vol[SAMPLES],                                                 // Collection of
prior volume samples
  lvl       = 10,                                               // Current "dampened"
audio level
  minLvlAvg = 0,                                                // For dynamic
adjustment of graph low &amp;amp; high
  maxLvlAvg = 512;


void setup() {

  // This is only needed on 5V Arduinos (Uno, Leonardo, etc.).
  // Connect 3.3V to mic AND TO AREF ON ARDUINO and enable this
  // line.  Audio samples are 'cleaner' at 3.3V.
  // COMMENT OUT THIS LINE FOR 3.3V ARDUINOS (FLORA, ETC.):
  //analogReference(EXTERNAL);

  delay(1000);                                                  // Power-up safety
delay or something like that.
  Serial.begin(57600);
  LEDS.addLeds&amp;lt;LED_TYPE, LED_DT, COLOR_ORDER&amp;gt;(leds, NUM_LEDS);
  FastLED.setBrightness(max_bright);
  set_max_power_in_volts_and_milliamps(5, 500);                 // FastLED 2.1 Power
management set at 5V, 500mA
} // setup()



void loop() {
  seashell();
  show_at_max_brightness_for_power();                           // Power managed
```

```
FastLED display
  Serial.println(LEDS.getFPS());
} // loop()



void seashell() {

  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 &lt;= NOISE) ? 0 : (n - NOISE);                       // Remove noise/hum
  lvl = ((lvl * 7) + n) &gt;&gt; 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 &lt; 0L)       height = 0;                      // Clip output
  else if (height &gt; TOP) height = TOP;
  if (height &gt; peak)     peak   = height;                 // Keep 'peak' dot
at top


  // Color pixels based on rainbow gradient
  for (i=0; i&lt;NUM_LEDS; i++) {
    if (i &gt;= height)   leds[i].setRGB( 0, 0,0);
    else leds[i] = CHSV(map(i,0,NUM_LEDS-1,50,70), 255, 255);  //constrained to
yellow color, change CHSV values for rainbow
  }

  // Draw peak dot
  if (peak &gt; 0 &amp;&amp; peak &lt;= NUM_LEDS-1) leds[peak] = CHSV(map(peak,
0,NUM_LEDS-1,50,70), 255, 255);

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

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

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

  // Get volume range of prior frames
  minLvl = maxLvl = vol[0];
  for (i=1; i&lt;SAMPLES; i++) {
    if (vol[i] &lt; minLvl)      minLvl = vol[i];
    else if (vol[i] &gt; 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) &lt; TOP) maxLvl = minLvl + TOP;
  minLvlAvg = (minLvlAvg * 63 + minLvl) &gt;&gt; 6;              // Dampen min/
max levels
  maxLvlAvg = (maxLvlAvg * 63 + maxLvl) &gt;&gt; 6;              // (fake
rolling average)
```

```
} // fastbracelet()
```

If you encounter trouble...

Any time you hit a roadblock with a NeoPixel project, we'll usually ask that you start with the "strandtest" example from our own Adafruit_NeoPixel library. This helps us narrow down whether it's a hardware or software issue. The library is installed similarly to FastLED or any other (`Sketch > Include Library > Manage Libraries...`)

You'll find the strandtest example under File→Sketchbook→Libraries→Adafruit_NeoPixel→strandtest

If strandtest fails to run, this suggests a hardware issue...for example, connecting to the wrong input on the LED ring, or the wrong Flora pin.

If you're new to Arduino programming and LEDs, we usually suggest starting with the Adafruit_NeoPixel library...it's pretty basic, the strip declaration is more conventional, and we can stay on top of keeping it compatible with our own products and the most mainstream Arduino boards.

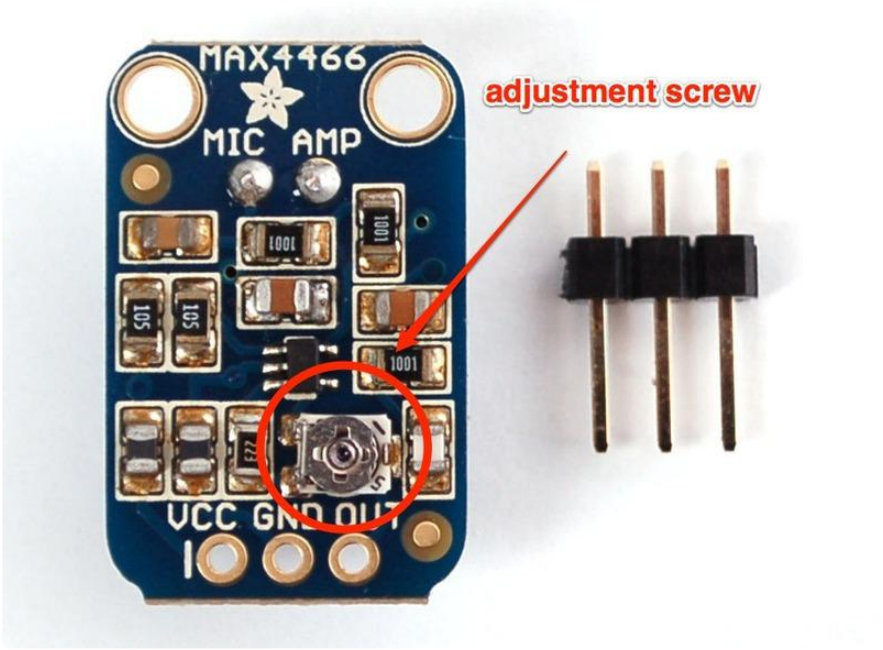
As FastLED is a more "bleeding edge" third-party library, we can't always guarantee compatibility across versions or with specific boards. You can find help through their community on Google+ (https://adafru.it/ebn). This is potent stuff, written by people with a deep appreciation for LED art, and we wanted to showcase it.

Calibration

Microphone Adjustment

You want the necklace to respond to your voice, but not go crazy at every little ambient noise. Some of this adjustment will happen in the code, but the nice thing about this mic breakout is that it has a teensy little adjustment screw. By configuring the code to respond to a medium gain, you'll have wiggle room to adjust on-the-fly for different environments. You'll be able to turn it down for noisy convention halls or up for quiet storytelling sessions.

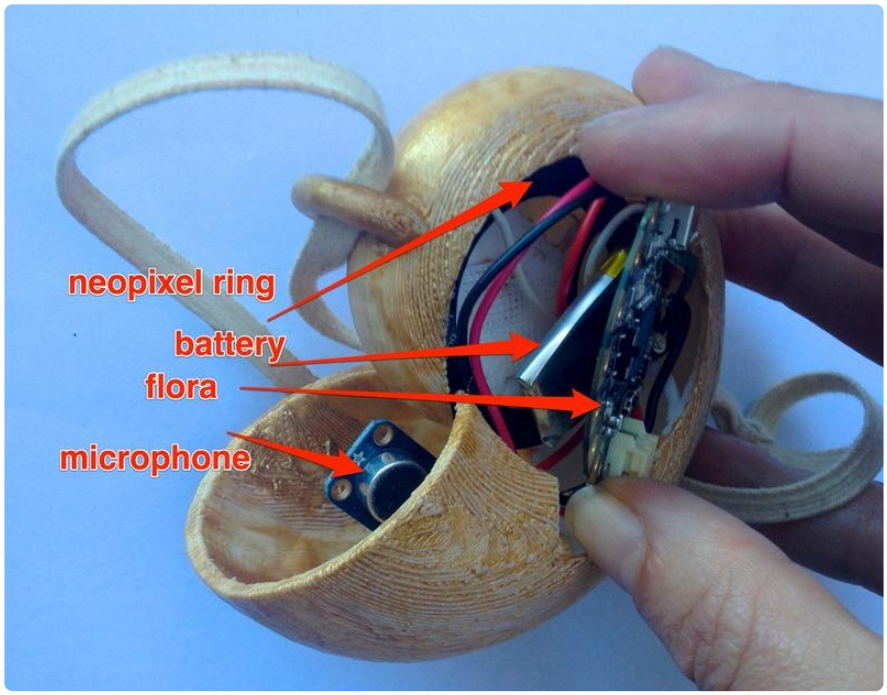
Take your teeny weeny screwdriver and twist the gain adjustment screw until it's roughly in the middle of its turning radius.



Now, make some noise. See how loud you need to be for your LEDs to respond.

Look at the code and find the value for NOISE. To make the mic more sensitive, make this number smaller. To make it less sensitive, make this number bigger.

Finishing



Put it all together!

First, place the neopixel ring down. There's a star on the back of the ring -- that marks the first LED to light, so put this at the top.

Slip the microphone through into the "bell" part of the shell so it's facing up toward your face when you have the necklace on.

Coil the battery wire around and set it on top of the neopixel ring.

Pop the Flora into place on top of it all and thread your elastic or necklace cord through the ring at the top of the shell. It will all stay in place nicely (you don't need to glue anything) and you can carefully pop the battery cable out for charging.

Turn your necklace on and off with Flora's onboard on/off switch.

When you need to recharge, open the necklace back up. Disconnect the battery and charge using a Micro Lipo Charger. (http://adafru.it/1304)

You may notice that your calibration / sensitivity changes when the mic is inside the seashell. Adjust it with the screw or in the code until you feel the magic.