# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Overview</strong></td>
<td>3</td>
</tr>
<tr>
<td>• Parts Needed</td>
<td>4</td>
</tr>
<tr>
<td><strong>Wiring Diagram</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>Software</strong></td>
<td>7</td>
</tr>
<tr>
<td>• Upload the Code</td>
<td>8</td>
</tr>
<tr>
<td><strong>Electronics Assembly</strong></td>
<td>11</td>
</tr>
<tr>
<td>• Waterproofing</td>
<td>13</td>
</tr>
<tr>
<td><strong>Install the Lights</strong></td>
<td>17</td>
</tr>
<tr>
<td>• Capacitive Touch Control</td>
<td>20</td>
</tr>
</tbody>
</table>
Overview

One Fish, Two Fish
Red Fish, Blue Fish
- Dr. Seuss

Decorate your aquarium with NeoPixels. Add a high density strip of lights above the water for a beautiful overall colored glow, with subtle color shifts and changes. Cycle between color modes to suit your mood, the time of day, or to signal feeding time.

For an extra special bit of underwater magic, attach a NeoPixel Strand and submerse the lights under the water. Diffuse the lights with seashells or bits of colored glass to complete the effect.

This DIY solution will cost you a lot less than the prepackaged aquarium LED light kits available at the pet store, and will give you a lot more control over brightness, color modes and customization. But it's the addition of the submersible strand that really makes this project special, and will make your aquarium stand out from the crowd.

My aquarium features Tetra fish that have been genetically bred so that their scales fluoresce and glow under blue lights. It's fun to play around with the NeoPixel's hue and saturation to find the color spectrum that makes their colors really shine.
Parts Needed

Microcontroller

We're using the adorable QT Py microcontroller. It's inexpensive and easy to use, and small enough to fit into a project with limited space.

![Adafruit QT Py - SAMD21 Dev Board with STEMMA QT](https://www.adafruit.com/product/4600)

What a cutie pie! Or is it... a QT Py? This diminutive dev board comes with our favorite lil chip, the SAMD21 (as made famous in our GEMMA M0 and Trinket M0 boards). This time it...

Lights

NeoPixel LED dots are rugged and robust, and submersible underwater so long as the ends and connectors are sealed up. They're easy to hide under seashells or glass beads and are really what make this project special.

![Adafruit NeoPixel LED Dots Strand - 20 LEDs at 2" Pitch](https://www.adafruit.com/product/3630)

Attaching NeoPixel strips to your costume can be a struggle as the flexible PCBs can crack when bent too much. So how to add little dots of color? Use these stranded NeoPixel dots!

This project also features indirect lighting from above-water lights to illuminate the entire tank and make the water appear to glow. These 144/meter lights will offer the smoothest animation and color shifting because of their deliciously high density. If you're on a budget, the 30/m ([https://adafru.it/OZA](https://adafru.it/OZA)) or 60/m ([https://adafru.it/IFs](https://adafru.it/IFs)) NeoPixels can work just about as well.
Adafruit NeoPixel Digital RGBW LED Strip - Black PCB 144 LED/m
What is better than smart RGB LEDs?
Smart RGB+White LEDs! These NeoPixels now have 4 LEDs in them (red, green, blue and white) for excellent lighting effects. These LED...
https://www.adafruit.com/product/2848

Additional Bits & Pieces

This build uses JST connectors between the light strips and the QT Py microcontroller, to make reprogramming your lights easier. It can be tricky to move an aquarium once it's set up, so making the QT Py easily removable is a good idea in case you ever want to update the code and give your fish something new to look at.

This also makes the submersible light strand easy to swap out in case your waterproofing isn't quite up to long term submersion.

1 x 2-pin Connector
2-pin JST SM Plug + Receptacle Cable Set
https://www.adafruit.com/product/2880

1 x 3-Pin Connector
3-pin JST SM Plug + Receptacle Cable Set
https://www.adafruit.com/product/1663

1 x Extension Wire Ribbon Cable
Silicone Cover Stranded-Core Ribbon Cable - 4 Wires 1 Meter Long - 26AWG Black
https://www.adafruit.com/product/3892

1 x Clear Heat Shrink
Food-Grade Heat Shrink - 3/8" diameter 12" long
https://www.adafruit.com/product/1020

The code has 6 different color modes you can cycle through to suit your mood, as well as a software on/off brightness toggle. These features are triggered by touching a length of copper tape connected to the QT Py's onboard capacitive touch pads. Adafruit carries copper tape with a conductive adhesive on the back, making it ideal for use as a stylish and steampunk-style capacitive touch switch.
1 x **Copper Tape**  
Copper Foil Tape with Conductive Adhesive - 25mm x 15 meter roll  
https://www.adafruit.com/product/1127

Whenever working with electronics and water, it's a good idea to have a quick way to turn your whole system off in case of splashes or shorts. This on/off switch plugs inline with your USB cable for a no-fuss off switch.

Also grab a USB C cable for programming and powering the QT Py.

1 x **On/Off Switch**  
USB Cable with Switch  
https://www.adafruit.com/product/1620

1 x **USB C Cable**  
USB C Cable - 1 Meter Long  
https://www.adafruit.com/product/4474

### Tools & Accessories

This list may vary depending on your build environment, but these are all useful things to have ready.

- Solder-seal wire connectors (https://adafruit.it/OWf)  
- 3/4" cable staples (https://adafruit.it/OZB) and a hammer for installation  
- Hot glue gun  
- Heat gun  
- Soldering iron & accessories  
- Aquarium gravel - fluorescent colors are available! Shop around.  
- Colored glass, sea shells, and plants or plastic plants made for aquariums

### Wiring Diagram
The QT Py is connected to the NeoPixel strip, which is daisy-chained to the submersible NeoPixel strand.

- QT Py 5v to NeoPixel +5v
- QT Py G to NeoPixel G
- QT Py A1 to NeoPixel DIN

Solder a long wire to QT Py A2 and A3 to use as capacitive touch controllers for mode switching and on/off.

This diagram shows all the connections in the simplest way. For the actual build, we'll be adding connectors and extension wires between the QT Py and the strips, to make updating the code easier later on.

Software

It's a great idea to get your software all set up and loaded onto your board right away, to make testing your connections easier later on.

To get the code running you'll need:

1. Arduino IDE (1.8 or newer)
2. Adafruit Board support for QT Py
3. Arduino libraries: FastLED, Adafruit_FreeTouch, Adafruit_NeoPixel
1. Arduino IDE

If you’re not using a recent version of the Arduino IDE, this would be a good time to upgrade (https://adafru.it/fvm). If this is your first time using Arduino, head over to this guide to get it installed (https://adafru.it/jDQ). It's free and fairly simple to get set up.

2. Board Support

You'll need to tell the Arduino IDE which board you're using. This takes just a few minutes of setup, and you'll only need to do it once.

Here is a step-by-step tutorial for setting up QT Py (https://adafru.it/OZD)

3. Libraries

All three libraries can be installed using the Arduino Library Manager — use Sketch → Include Library → Manage Libraries... and search for all or part of the library's name, then click “Install.”

Look for:

- FastLED
- Adafruit_FreeTouch
- Adafruit_NeoPixel

Adafruit_NeoPixel isn’t absolutely required for this project, but it's handy to have installed in case you have problems with FastLED. Troubleshooting the basics is a little easier with Adafruit_NeoPixel.

Upload the Code

Get the code by clicking "Download Project Zip" in the code listing below. Unzip the code into your Arduino project folder.

Plug your microcontroller into your computer with a USB cable. In the Arduino IDE, go to File -> Open to open the code file. Go to Tools -> Boards and select the name of the board (Adafruit QT Py). Then go to Tools -> Port and select the board there too. (If it's not showing up there, be sure your microcontroller is plugged into your computer via USB).
// Code by Erin St. Blaine for Adafruit Industries
// For QT Py board

#include "Adafruit_FreeTouch.h"
#include "FastLED.h"

#define CAPTOUCH_PIN A2  //A2 capacitive touch pin
#define CAPTOUCH_PIN2 A3 //A3 capacitive touch pin
#define NEOPIXEL_PIN A1 //A1 neopixel pin
#define NUM_LEDS    164 //how many pixels total

#define LED_TYPE    WS2812
#define COLOR_ORDER GRB

CRGB leds[NUM_LEDS]; //LED array

// These variables will affect the way the gradient animation looks. Feel free to
// mess with them.
int SPEEDO = 10;
int STEPS = 50;
int HUE = 0;
int SATURATION = 255;
int COLORCHANGE = 50;
int BRIGHTNESS = 200;
int onBright = 200;

// Calibrating your capacitive touch sensitivity: Change this variable to something
// between your capacitive touch serial readouts for on and off
int touch = 500;
long oldState = 0;
long offState = 0;

// set up capacitive touch button using the FreeTouch library
Adafruit_FreeTouch qt_1 = Adafruit_FreeTouch(CAPTOUCH_PIN, OVERSAMPLE_4,
RESISTOR_50K, FREQ_MODE_NONE);
Adafruit_FreeTouch qt_2 = Adafruit_FreeTouch(CAPTOUCH_PIN2, OVERSAMPLE_4,
RESISTOR_50K, FREQ_MODE_NONE);

TBlendType currentBlending;
CRGBPalette16 currentPalette;

void setup() {
    Serial.begin(115200);
    if (!qt_1.begin())
        Serial.println("Failed to begin qt on pin A2");
    if (!qt_2.begin())
        Serial.println("Failed to begin qt on pin A3");
    FastLED.addLeds<WS2812, NEOPIXEL_PIN, COLOR_ORDER>(leds, NUM_LEDS); // Set up
    neopixels with FastLED
    FastLED.setBrightness(BRIGHTNESS); // set global brightness
    FastLED.setMaxPowerInVoltsAndMilliamps(3,350); //Constrain FastLED's power usage
}

void loop() {
    Serial.print(qt_1.measure());
    Serial.print(' ');
    checkpress(); //check to see if the button's been pressed
    delay(20);
}

void checkpress() {
    // Get current button state.
`long` newState = qt_1.measure();
`long` switchState = qt_2.measure();
Serial.println(qt_1.measure());
Serial.println(qt_2.measure());
if (newState > touch && oldState < touch) {
    // Short delay to debounce button.
    delay(500);
    // Check if button is still low after debounce.
    `long` newState = qt_1.measure();
    `long` switchState = qt_2.measure();
}
if (switchState > touch && offState < touch) {
    // Short delay to debounce button.
    delay(500);
    // Check if button is still low after debounce.
    `long` newState = qt_1.measure();
}

if (newState > touch) {
    // Set the last button state to the old state.
    oldState = newState;
    offState = switchState;
}

if (switchState > touch) {
    if (BRIGHTNESS == onBright) {
        dark();
        switchState = 0;
        BRIGHTNESS = 0;
    } else {
        BRIGHTNESS = onBright;
    }
}

if (HUE > 255) {
    HUE = 0;
} else {
    Gradient();
}

void Gradient() {
    SetupGradientPalette();
    static `uint8_t` startIndex = 0;
    startIndex = startIndex - 1; // motion speed
    FillLEDsFromPaletteColors(startIndex);
    FastLED.show();
    FastLED.delay(SPEEDO);

    // Set the last button state to the old state.
    oldState = newState;
    offState = switchState;
}

// GRADIENT ------------------------------------------------------------------
void SetupGradientPalette()
{
    CRGB light = CHSV( HUE + 25, SATURATION - 20, BRIGHTNESS);
    CRGB lightmed = CHSV( HUE + 15, SATURATION - 10, BRIGHTNESS - 50);
    CRGB medium = CHSV( HUE + 10, SATURATION - 15, BRIGHTNESS);
    CRGB dark = CHSV( HUE, SATURATION, BRIGHTNESS);
    CRGB black = CHSV( HUE, SATURATION, 0);

    currentPalette = CRGBPalette16(
        black, light, light, light, lightmed, lightmed, lightmed, medium,
        medium, medium, medium, dark, dark, dark, dark, black);
}

void FillLEDsFromPaletteColors( uint8_t colorIndex)
{
    uint8_t brightness = BRIGHTNESS;

    for( int i = 0; i < NUM_LEDS; i++) {
        leds[i] = ColorFromPalette( currentPalette, colorIndex, brightness,
                                      currentBlending);
        colorIndex += STEPS;
    }
}

void dark()
{
    for( int i = 0; i < NUM_LEDS; i++) {
        leds[i] = CRGB::Black;
        FastLED.show();
        delay(20);
    }
}

Electronics Assembly

Solder your 3-pin female connector to the QT Py as shown:

- Right wire to 5v
- Middle wire to A1
- Left wire to G

This will attach to the NeoPixel strip.
Solder your 2-pin female connector to pins A2 and A3. These pins will connect to the capacitive touch wires we're using as a mode controller and an on/off switch.

Find the "in" end of your NeoPixel strip - it's the end with the little arrows pointing away. Our microcontroller needs to be connected to this end for the pixels to work.

Solder the male side of your 3-pin connector to the red, black, and white wires. It's helpful to plug the two halves of the connectors together before you solder so you can be sure which wire is connecting to which pin.

Be sure the red wire connects through both halves of the connector to 5v, the white wire to A1, and the black wire to G.

If you've already uploaded your code, plug in your QT Py with a USB cable and test to be sure your lights come on.

Solder a long wire to each pin of your male 2-pin connector. These wires will need to be long enough to reach your copper tape "switch" pads so give yourself a little extra wire. You can trim them down later.
Plug the 2-pin connector in to the QT Py. Strip some shielding from the far end of the wire and pinch it with your fingers to make the light strip change color modes (A2) or turn on and off (A3).

**Waterproofing**

Next we'll add the submersible NeoPixel strand. These lights are really sturdily made and can happily be used underwater, but you will need to seal both ends if you want to submerse the entire strand.

First we'll seal the "out" end completely, then we'll add a waterproof extension wire to the "in" end, so we can run the extension up the back of the tank to connect with the above-water NeoPixel strip.
Find the "out" end of your NeoPixel dot strand. On my strand it was the end with the female connector, but always double check the arrows on the back of the strip in case the strip you have has the connectors assembled differently.

Cut this connector off. Cut a piece of clear heat shrink tubing and slide it over the exposed wires.

Fill the end of the heat shrink with hot glue. While the glue is wet and molten, grab your heat gun and shrink the heat shrink, being sure the end of the strands are fully enclosed. This will encase the end of the strand in plastic and completely waterproof it.
Take the female connector you cut off (from the out end of the dot strand). Splice it onto the out end of your LED strip. Connect the red wire to the red-striped wire on the connector, the white wire to the middle wire, and the black wire to the remaining wire.

This connector will be above the waterline, so you don't need to worry about getting the connections watertight.

Plug in your NeoPixel dot strand and test to be sure your lights come on.

If you want the strand of lights to be visible running down the back wall of your aquarium, you can stop here and start putting your aquarium together. Just keep the connector above the waterline and drop the rest of the strand into the water.

I want my entire light strand to be submersed, without any lights along the back wall. To achieve this, there's one more step: adding a waterproofed extension wire to the in end of the NeoPixel strand, long enough to reach the bottom of the tank so I can bury the entire light strand under the gravel.

I'm using [solder seal connectors](https://adafruit.it/OWf) with silicone stranded wire to make the underwater connections waterproof. A Solder seal connector is a piece of heat shrink that contains some low-melt solder and a couple of glue seals on either
side. A heat gun will melt the solder and the glue, creating a submersible solder connection in just one step.

These work great for underwater connections or for in-place repairs on your project, but they are a bit more likely to cause a cold solder joint than a good connection made with a soldering iron. If your project needs to flex or move a lot or needs to pass a lot of current, stick with traditional soldering and heat shrink with glue. But for small projects that don't need to flex, these can save you a lot of time.

And of course, if you don't have any of these on-hand, you can also use the same hot-glue-and-heat-shrink method detailed above over all three solder joints.

For the wire, I'm using 4-stranded 26awg silicone ribbon cable (https://adafru.it/IHe). Silicone stranded wire has a completely waterproof silicone shielding, which will never absorb water or get waterlogged. Choice of wire is important here - cheaper plastic-shielded wire really won't work as well for an underwater connection as the water will eventually seep through the plastic.
Find the striped wire - it's on one edge of your 4-wire strand. Grab the wire on the opposite edge of your ribbon cable and remove it completely, so you're left with 3 wires, one of which is striped.

Separate the ends for a few inches and slip on your solder seal connectors.

Cut the connector off the in end of your NeoPixel dot strand. Twist the ribbon cable onto the ends securely: striped wire to red striped wire, middle wire to middle wire, and remaining wire to remaining wire.

Slide the solder seal connectors over your twisted wires so the gray ring in the middle is positioned right over the exposed wires. Heat the connectors for 20-30 seconds until the metal liquefies and the glue rings on either side have melted completely.

Finally, attach the connector you just cut off to the other end of the extension wire. I used solder seal connectors here too, but this connection will be above the water so waterproofing is not essential. I just like to use the solder seal connectors! It's satisfying to watch them work.

Install the Lights

My aquarium stand has a wooden lid that opens and closes on a hinge. I installed the light strip and the QT Py inside the lid. The lights are powered directly through the QT
Py's USB port. I used a short USB cable and daisy-chained my USB power switch inline with the cable and then onward to the power outlet.

My 1m of lights was exactly the right length to go 3/4 around my 20 gallon tank - along both sides and the front edge. I left the back edge alone, since that's where the filter is and where the wires come up for the heater.

Tape them in place and turn them on to be sure you're happy with placement and alignment.
I used wire staples to secure my LED strip to the inside of my aquarium lid. You could also use clear packing tape over the strip, secured with staples or more tape along the edges to keep water from seeping in.

Avoid using silicone glue or other chemical-heavy solutions. Fish are very sensitive to their environment, and introducing smelly glue into their environment could be bad. To you it's smelly, but to them, it's toxic.

Plug your NeoPixel dot strand into the end of your NeoPixel strip and run the extension cord down the back corner of your aquarium. Bury the whole strip underneath the gravel so the wire is hidden.

Use glass beads or seashells to diffuse the light pixels on the bottom. Diffusion can be achieved with a variety of aquarium props or plants, and it's really important - your fish are sensitive to light and probably don't want to live in a 24 hour disco, so give them a break and soften the lights.
Capacitive Touch Control

Run the two capacitive touch wires out the back of the aquarium and connect them to your control pads. Capacitive touch control pads can be made of anything conductive. I used a length of copper tape lining the edge of my aquarium. It is a lovely decorative accent, but also functions as a mode-changing switch.

If you're using copper tape, strip a lot of shielding from the end of your wire - at least 3/4 inch to get a really good connection. Stick the bare wire under the tape and press it down really well. The entire strip of tape becomes your conductive switch - touch anywhere on the tape strip to switch color modes.
The wire attached to A2 will cycle through color modes, and the wire attached to A3 will toggle the lights on and off.

In order to have two different functions along your copper tape strip, you'll need to leave a small gap in the tape. I also found that attaching too much tape can make the switch really sensitive, which makes it go a little haywire. It works best to just use a small section of tape as your switch area. With my setup, touching along the left side cycles modes and touching along the right side turns the lights off or on.

Another option is to use nylon fabric squares. You can cut them into any shape you like and stick them to the wire the same way as you would to the copper tape.