



# Twinkling Neopixel Parasol

Created by Erin St Blaine



<https://learn.adafruit.com/twinkling-led-parasol>

Last updated on 2024-06-03 01:53:21 PM EDT

# Table of Contents

<a href="#">Introduction</a>	<a href="#">3</a>
<a href="#">Code</a>	<a href="#">4</a>
<a href="#">Wiring Diagram</a>	<a href="#">16</a>
<a href="#">LED Assembly</a>	<a href="#">17</a>
<a href="#">Parasol Assembly</a>	<a href="#">23</a>
<a href="#">Creating Color Palettes</a>	<a href="#">27</a>

---

# Introduction

This parasol will shield you from the fierce rays of the sun, and protect you from the dark, fearsome things of the night.

Neopixels surround the parasol's edge, giving a beautiful, customizable twinkling glow to its bearer. Learn to easily choose color palettes to go with your outfit for the evening, or stack up hundreds of different palettes for a light show that lasts all night without repeating.



## Materials Needed:

- Teensy 3.1/3.2
- 40 to 60 Neopixels (individual breakout)
- On/off switch
- 2000mAh Lithium Polymer battery
- 30awg Silicone stranded wire in various colors

## Non-Adafruit Ingredients

- Parasol or umbrella (I used "wedding parasol" as a search term to find delicate lacey ones)
- A bit of matching netting or fabric for the battery pocket
- Lace trim

## Tools

- Soldering iron & accessories
- Hot glue gun
- Heat gun
- Needle & thread



---

## Code



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

To get the code running on the Teensy you'll need:

1. Arduino IDE (1.6.5 or newer preferred)
2. Teensyduino Installer
3. FastLED Library

It's also helpful to install the Adafruit NeoPixel library, for testing purposes or in case FastLED is not working. (Sketch→Include Library→Manage Libraries...).

## 1. Arduino IDE

If you're not using a recent version of the Arduino IDE (1.6.5 or newer), this would be a [good time to upgrade](https://adafru.it/fvm) (<https://adafru.it/fvm>).

## 2. Teensyduino Installer

Once you have that software installed and working, download and run the [Teensyduino installer](https://adafru.it/iwF) (<https://adafru.it/iwF>), which adds support for the full line of Teensy microcontroller boards in the Arduino IDE.

In the Arduino IDE, from the **Tools** menu, select **Board→Teensy 3.2** and **CPU Speed→72 MHz (Optimized)**. Confirm that you can compile and upload the classic “blink” sketch to the Teensy board. Be sure you can blink the light before continuing.

## 3. FastLED Library

Use the Library Manager in the Arduino IDE to install this (Sketch→Include Library→Manage Libraries...), or if you're using an older version of the Arduino IDE, it can be [downloaded](https://adafru.it/ezj) (<https://adafru.it/ezj>) and [installed manually](https://adafru.it/dNR) (<https://adafru.it/dNR>).

Once you've got everything installed, restart Arduino and select Teensy 3.1/3.2 from the Tools > Boards dropdown menu and upload the code.

This code was written by Mark Kriegsman. There are a few simple variables you can change: brightness of the LEDs and how quickly the lights shift between palettes. I'll get more into how to add and configure your own color palettes later on.

```
#include "FastLED.h"

// ColorWavesWithPalettes
```



```

// Animated shifting color waves, with several cross-fading color palettes.
// by Mark Kriegsman, August 2015
//
// Color palettes courtesy of cpt-city and its contributors:
//   http://soliton.vm.bytemark.co.uk/pub/cpt-city/
//
// Color palettes converted for FastLED using "PaletteKnife" v1:
//   http://fastled.io/tools/paletteknife/
//

//#if FASTLED_VERSION < 3001000
//error "Requires FastLED 3.1 or later; check github for latest code."
//endif

#define DATA_PIN    12
// #define CLK_PIN    4
#define LED_TYPE     WS2812
#define COLOR_ORDER   GRB
#define NUM_LEDS     56           // Change this to reflect the number of LEDs you have
#define BRIGHTNESS   255         // Set Brightness here

CRGB leds[NUM_LEDS];

// ten seconds per color palette makes a good demo
// 20-120 is better for deployment
#define SECONDS_PER_PALETTE 10

void setup() {
  delay(3000); // 3 second delay for recovery

  // tell FastLED about the LED strip configuration
  FastLED.addLeds<LED_TYPE,DATA_PIN,COLOR_ORDER>(leds, NUM_LEDS)
    .setCorrection(TypicalLEDStrip) // cpt-city palettes have different color
    balance
    .setDither(BRIGHTNESS < 255);

  // set master brightness control
  FastLED.setBrightness(BRIGHTNESS);
}

// Forward declarations of an array of cpt-city gradient palettes, and
// a count of how many there are. The actual color palette definitions
// are at the bottom of this file.
extern const TProgmemRGBGradientPalettePtr gGradientPalettes[];
extern const uint8_t gGradientPaletteCount;

// Current palette number from the 'playlist' of color palettes
uint8_t gCurrentPaletteNumber = 0;

CRGBPalette16 gCurrentPalette( CRGB::Black);
CRGBPalette16 gTargetPalette( gGradientPalettes[0] );

void loop()
{
  EVERY_N_SECONDS( SECONDS_PER_PALETTE ) {
    gCurrentPaletteNumber = addmod8( gCurrentPaletteNumber, 1,
    gGradientPaletteCount);
    gTargetPalette = gGradientPalettes[ gCurrentPaletteNumber ];
  }

  EVERY_N_MILLISECONDS(40) {
    nblendPaletteTowardPalette( gCurrentPalette, gTargetPalette, 16);
  }

  colorwaves( leds, NUM_LEDS, gCurrentPalette);

  FastLED.show();
}

```

```

FastLED.delay(20);
}

// This function draws color waves with an ever-changing,
// widely-varying set of parameters, using a color palette.
void colorwaves( CRGB* ledarray, uint16_t numleds, CRGBPalette16& palette)
{
    static uint16_t sPseudotime = 0;
    static uint16_t sLastMillis = 0;
    static uint16_t sHue16 = 0;

    uint8_t sat8 = beatsin88( 87, 220, 250);
    uint8_t brightdepth = beatsin88( 341, 96, 224);
    uint16_t brightnessthetainc16 = beatsin88( 203, (25 * 256), (40 * 256));
    uint8_t msmultiplier = beatsin88(147, 23, 60);

    uint16_t hue16 = sHue16; //gHue * 256;
    uint16_t hueinc16 = beatsin88(113, 300, 1500);

    uint16_t ms = millis();
    uint16_t deltams = ms - sLastMillis ;
    sLastMillis = ms;
    sPseudotime += deltams * msmultiplier;
    sHue16 += deltams * beatsin88( 400, 5, 9);
    uint16_t brightnesstheta16 = sPseudotime;

    for( uint16_t i = 0 ; i < numleds; i++) {
        hue16 += hueinc16;
        uint8_t hue8 = hue16 / 256;
        uint16_t h16_128 = hue16 >> 7;
        if( h16_128 & 0x100) {
            hue8 = 255 - (h16_128 >> 1);
        } else {
            hue8 = h16_128 >> 1;
        }

        brightnesstheta16 += brightnessthetainc16;
        uint16_t b16 = sin16( brightnesstheta16 ) + 32768;

        uint16_t bri16 = (uint32_t)((uint32_t)b16 * (uint32_t)b16) / 65536;
        uint8_t bri8 = (uint32_t)((uint32_t)bri16 * brightdepth) / 65536;
        bri8 += (255 - brightdepth);

        uint8_t index = hue8;
        //index = triwave8( index);
        index = scale8( index, 240);

        CRGB newcolor = ColorFromPalette( palette, index, bri8);

        uint16_t pixelnumber = i;
        pixelnumber = (numleds-1) - pixelnumber;

        nbblend( ledarray[pixelnumber], newcolor, 128);
    }
}

// Alternate rendering function just scrolls the current palette
// across the defined LED strip.
void palettetest( CRGB* ledarray, uint16_t numleds, const CRGBPalette16&
gCurrentPalette)
{
    static uint8_t startindex = 0;
    startindex--;
    fill_palette( ledarray, numleds, startindex, (256 / NUM_LEDS) + 1,
gCurrentPalette, 255, LINEARBLEND);
}

```

```

// Gradient Color Palette definitions for 33 different cpt-city color palettes.
// 956 bytes of PROGMEM for all of the palettes together,
// +618 bytes of PROGMEM for gradient palette code (AVR).
// 1,494 bytes total for all 34 color palettes and associated code.

// Gradient palette "ib_jul01_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ing/xmas/tn/ib_jul01.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 16 bytes of program space.

DEFINE_GRADIENT_PALETTE( ib_jul01_gp ) {
    0, 194, 1, 1,
    94, 1, 29, 18,
    132, 57, 131, 28,
    255, 113, 1, 1};

// Gradient palette "es_vintage_57_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/vintage/tn/
// es_vintage_57.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_vintage_57_gp ) {
    0, 2, 1, 1,
    53, 18, 1, 0,
    104, 69, 29, 1,
    153, 167, 135, 10,
    255, 46, 56, 4};

// Gradient palette "es_vintage_01_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/vintage/tn/
// es_vintage_01.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 32 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_vintage_01_gp ) {
    0, 4, 1, 1,
    51, 16, 0, 1,
    76, 97, 104, 3,
    101, 255, 131, 19,
    127, 67, 9, 4,
    153, 16, 0, 1,
    229, 4, 1, 1,
    255, 4, 1, 1};

// Gradient palette "es_rivendell_15_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/rivendell/tn/
// es_rivendell_15.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_rivendell_15_gp ) {
    0, 1, 14, 5,
    101, 16, 36, 14,
    165, 56, 68, 30,
    242, 150, 156, 99,
    255, 150, 156, 99};

// Gradient palette "rgi_15_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ds/rgi/tn/rgi_15.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 36 bytes of program space.

DEFINE_GRADIENT_PALETTE( rgi_15_gp ) {
    0, 4, 1, 31,
    31, 55, 1, 16,

```



```

    63, 197,  3,  7,
    95,  59,  2, 17,
   127,  6,  2, 34,
   159, 39,  6, 33,
   191, 112, 13, 32,
   223, 56,  9, 35,
   255, 22,  6, 38};

// Gradient palette "retro2_16_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ma/retro2/tn/
retro2_16.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 8 bytes of program space.

DEFINE_GRADIENT_PALETTE( retro2_16_gp ) {
    0, 188,135,  1,
   255, 46,  7,  1};

// Gradient palette "Analogous_1_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/red/tn/
Analogous_1.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( Analogous_1_gp ) {
    0,  3,  0,255,
   63, 23,  0,255,
  127, 67,  0,255,
  191, 142,  0, 45,
  255, 255,  0,  0};

// Gradient palette "es_pinksplash_08_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/pink_splash/tn/
es_pinksplash_08.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_pinksplash_08_gp ) {
    0, 126, 11,255,
   127, 197,  1, 22,
   175, 210,157,172,
   221, 157,  3,112,
   255, 157,  3,112};

// Gradient palette "es_pinksplash_07_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/pink_splash/tn/
es_pinksplash_07.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_pinksplash_07_gp ) {
    0, 229,  1,  1,
   61, 242,  4, 63,
  101, 255, 12,255,
  127, 249, 81,252,
  153, 255, 11,235,
  193, 244,  5, 68,
  255, 232,  1,  5};

// Gradient palette "Coral_reef_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/other/tn/
Coral_reef.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 24 bytes of program space.

DEFINE_GRADIENT_PALETTE( Coral_reef_gp ) {
    0,  40,199,197,
   50,  10,152,155,
   96,  1,111,120,

```

```

    96,  43,127,162,
    139, 10, 73,111,
    255,  1, 34, 71};

// Gradient palette "es_ocean_breeze_068_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/ocean_breeze/tn/
// es_ocean_breeze_068.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 24 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_ocean_breeze_068_gp ) {
    0, 100,156,153,
    51,  1, 99,137,
    101,  1, 68, 84,
    104, 35,142,168,
    178,  0, 63,117,
    255,  1, 10, 10};

// Gradient palette "es_ocean_breeze_036_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/ocean_breeze/tn/
// es_ocean_breeze_036.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 16 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_ocean_breeze_036_gp ) {
    0,  1,  6,  7,
    89,  1, 99,111,
    153, 144,209,255,
    255,  0, 73, 82};

// Gradient palette "departure_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/mjf/tn/departure.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 88 bytes of program space.

DEFINE_GRADIENT_PALETTE( departure_gp ) {
    0,  8,  3,  0,
    42, 23,  7,  0,
    63, 75, 38,  6,
    84, 169, 99, 38,
    106, 213,169,119,
    116, 255,255,255,
    138, 135,255,138,
    148, 22,255, 24,
    170,  0,255,  0,
    191,  0,136,  0,
    212,  0, 55,  0,
    255,  0, 55,  0};

// Gradient palette "es_landscape_64_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/landscape/tn/
// es_landscape_64.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 36 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_landscape_64_gp ) {
    0,  0,  0,  0,
    37,  2, 25,  1,
    76, 15,115,  5,
    127, 79,213,  1,
    128, 126,211, 47,
    130, 188,209,247,
    153, 144,182,205,
    204, 59,117,250,
    255,  1, 37,192};

// Gradient palette "es_landscape_33_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/landscape/tn/
// es_landscape_33.png.index.html

```

```

// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 24 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_landscape_33_gp ) {
    0, 1, 5, 0,
    19, 32, 23, 1,
    38, 161, 55, 1,
    63, 229, 144, 1,
    66, 39, 142, 74,
    255, 1, 4, 1};

// Gradient palette "rainbowsherbet_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ma/icecream/tn/
// rainbowsherbet.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( rainbowsherbet_gp ) {
    0, 255, 33, 4,
    43, 255, 68, 25,
    86, 255, 7, 25,
    127, 255, 82, 103,
    170, 255, 255, 242,
    209, 42, 255, 22,
    255, 87, 255, 65};

// Gradient palette "gr65_hult_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/hult/tn/gr65_hult.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 24 bytes of program space.

DEFINE_GRADIENT_PALETTE( gr65_hult_gp ) {
    0, 247, 176, 247,
    48, 255, 136, 255,
    89, 220, 29, 226,
    160, 7, 82, 178,
    216, 1, 124, 109,
    255, 1, 124, 109};

// Gradient palette "gr64_hult_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/hult/tn/gr64_hult.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 32 bytes of program space.

DEFINE_GRADIENT_PALETTE( gr64_hult_gp ) {
    0, 1, 124, 109,
    66, 1, 93, 79,
    104, 52, 65, 1,
    130, 115, 127, 1,
    150, 52, 65, 1,
    201, 1, 86, 72,
    239, 0, 55, 45,
    255, 0, 55, 45};

// Gradient palette "GMT_drywet_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/gmt/tn/GMT_drywet.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( GMT_drywet_gp ) {
    0, 47, 30, 2,
    42, 213, 147, 24,
    84, 103, 219, 52,
    127, 3, 219, 207,
    170, 1, 48, 214,
    212, 1, 1, 111,
    255, 1, 7, 33};

// Gradient palette "ib15_gp", originally from

```

```

// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ing/general/tn/ib15.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 24 bytes of program space.

DEFINE_GRADIENT_PALETTE( ib15_gp ) {
    0, 113, 91, 147,
    72, 157, 88, 78,
    89, 208, 85, 33,
    107, 255, 29, 11,
    141, 137, 31, 39,
    255, 59, 33, 89};

// Gradient palette "Fuschia_7_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/ds/fuschia/tn/
Fuschia-7.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( Fuschia_7_gp ) {
    0, 43, 3, 153,
    63, 100, 4, 103,
    127, 188, 5, 66,
    191, 161, 11, 115,
    255, 135, 20, 182};

// Gradient palette "es_emerald_dragon_08_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/emerald_dragon/tn/
es_emerald_dragon_08.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 16 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_emerald_dragon_08_gp ) {
    0, 97, 255, 1,
    101, 47, 133, 1,
    178, 13, 43, 1,
    255, 2, 10, 1};

// Gradient palette "lava_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/neota/elem/tn/lava.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 52 bytes of program space.

DEFINE_GRADIENT_PALETTE( lava_gp ) {
    0, 0, 0, 0,
    46, 18, 0, 0,
    96, 113, 0, 0,
    108, 142, 3, 1,
    119, 175, 17, 1,
    146, 213, 44, 2,
    174, 255, 82, 4,
    188, 255, 115, 4,
    202, 255, 156, 4,
    218, 255, 203, 4,
    234, 255, 255, 4,
    244, 255, 255, 71,
    255, 255, 255, 255};

// Gradient palette "fire_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/neota/elem/tn/fire.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( fire_gp ) {
    0, 1, 1, 0,
    76, 32, 5, 0,
    146, 192, 24, 0,
    197, 220, 105, 5,
    240, 252, 255, 31,
    250, 252, 255, 111,

```

```

255, 255,255,255});

// Gradient palette "Colorfull_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/atmospheric/tn/
Colorfull.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 44 bytes of program space.

DEFINE_GRADIENT_PALETTE( Colorfull_gp ) {
    0, 10, 85, 5,
    25, 29,109, 18,
    60, 59,138, 42,
    93, 83, 99, 52,
    106, 110, 66, 64,
    109, 123, 49, 65,
    113, 139, 35, 66,
    116, 192,117, 98,
    124, 255,255,137,
    168, 100,180,155,
    255, 22,121,174};

// Gradient palette "Magenta_Evening_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/atmospheric/tn/
Magenta_Evening.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( Magenta_Evening_gp ) {
    0, 71, 27, 39,
    31, 130, 11, 51,
    63, 213, 2, 64,
    70, 232, 1, 66,
    76, 252, 1, 69,
    108, 123, 2, 51,
    255, 46, 9, 35};

// Gradient palette "Pink_Purple_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/atmospheric/tn/
Pink_Purple.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 44 bytes of program space.

DEFINE_GRADIENT_PALETTE( Pink_Purple_gp ) {
    0, 19, 2, 39,
    25, 26, 4, 45,
    51, 33, 6, 52,
    76, 68, 62,125,
    102, 118,187,240,
    109, 163,215,247,
    114, 217,244,255,
    122, 159,149,221,
    149, 113, 78,188,
    183, 128, 57,155,
    255, 146, 40,123};

// Gradient palette "Sunset_Real_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/atmospheric/tn/
Sunset_Real.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( Sunset_Real_gp ) {
    0, 120, 0, 0,
    22, 179, 22, 0,
    51, 255,104, 0,
    85, 167, 22, 18,
    135, 100, 0,103,
    198, 16, 0,130,
    255, 0, 0,160};

```

```

// Gradient palette "es_autumn_19_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/es/autumn/tn/
es_autumn_19.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 52 bytes of program space.

DEFINE_GRADIENT_PALETTE( es_autumn_19_gp ) {
    0, 26, 1, 1,
    51, 67, 4, 1,
    84, 118, 14, 1,
    104, 137, 152, 52,
    112, 113, 65, 1,
    122, 133, 149, 59,
    124, 137, 152, 52,
    135, 113, 65, 1,
    142, 139, 154, 46,
    163, 113, 13, 1,
    204, 55, 3, 1,
    249, 17, 1, 1,
    255, 17, 1, 1};

// Gradient palette "Black_Blue_Magenta_White_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/basic/tn/
Black_Blue_Magenta_White.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( Black_Blue_Magenta_White_gp ) {
    0, 0, 0, 0,
    42, 0, 0, 45,
    84, 0, 0, 255,
    127, 42, 0, 255,
    170, 255, 0, 255,
    212, 255, 55, 255,
    255, 255, 255, 255};

// Gradient palette "Black_Magenta_Red_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/basic/tn/
Black_Magenta_Red.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( Black_Magenta_Red_gp ) {
    0, 0, 0, 0,
    63, 42, 0, 45,
    127, 255, 0, 255,
    191, 255, 0, 45,
    255, 255, 0, 0};

// Gradient palette "Black_Red_Magenta_Yellow_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/basic/tn/
Black_Red_Magenta_Yellow.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)
// Size: 28 bytes of program space.

DEFINE_GRADIENT_PALETTE( Black_Red_Magenta_Yellow_gp ) {
    0, 0, 0, 0,
    42, 42, 0, 0,
    84, 255, 0, 0,
    127, 255, 0, 45,
    170, 255, 0, 255,
    212, 255, 55, 45,
    255, 255, 255, 0};

// Gradient palette "Blue_Cyan_Yellow_gp", originally from
// http://soliton.vm.bytemark.co.uk/pub/cpt-city/nd/basic/tn/
Blue_Cyan_Yellow.png.index.html
// converted for FastLED with gammas (2.6, 2.2, 2.5)

```

```

// Size: 20 bytes of program space.

DEFINE_GRADIENT_PALETTE( Blue_Cyan_Yellow_gp ) {
    0,    0,  0,255,
    63,   0, 55,255,
    127,  0,255,255,
    191, 42,255, 45,
    255, 255,255,  0};

// Single array of defined cpt-city color palettes.
// This will let us programmatically choose one based on
// a number, rather than having to activate each explicitly
// by name every time.
// Since it is const, this array could also be moved
// into PROGMEM to save SRAM, but for simplicity of illustration
// we'll keep it in a regular SRAM array.
//
// This list of color palettes acts as a "playlist"; you can
// add or delete, or re-arrange as you wish.
const TProgmemRGBGradientPalettePtr gGradientPalettes[] = {
    Sunset_Real_gp,
    es_rivendell_15_gp,
    es_ocean_breeze_036_gp,
    rgi_15_gp,
    retro2_16_gp,
    Analogous_1_gp,
    es_pinksplash_08_gp,
    Coral_reef_gp,
    es_ocean_breeze_068_gp,
    es_pinksplash_07_gp,
    es_vintage_01_gp,
    departure_gp,
    es_landscape_64_gp,
    es_landscape_33_gp,
    rainbowherbet_gp,
    gr65_hult_gp,
    gr64_hult_gp,
    GMT_drywet_gp,
    ib_jul01_gp,
    es_vintage_57_gp,
    ib15_gp,
    Fuschia_7_gp,
    es_emerald_dragon_08_gp,
    lava_gp,
    fire_gp,
    Colorfull_gp,
    Magenta_Evening_gp,
    Pink_Purple_gp,
    es_autumn_19_gp,
    Black_Blue_Magenta_White_gp,
    Black_Magenta_Red_gp,
    Black_Red_Magenta_Yellow_gp,
    Blue_Cyan_Yellow_gp };

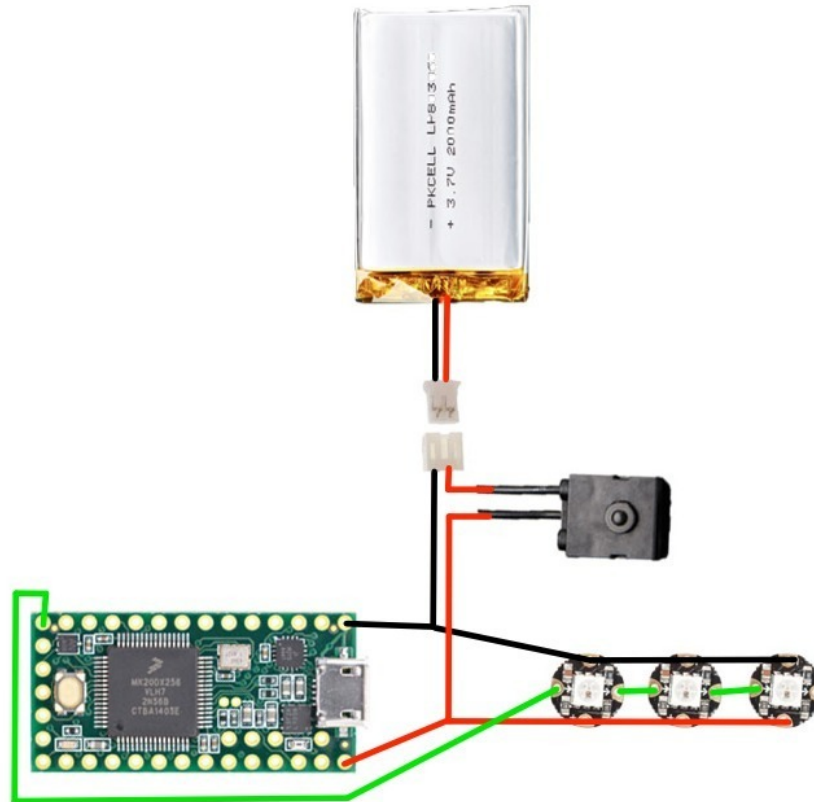
// Count of how many cpt-city gradients are defined:
const uint8_t gGradientPaletteCount =
    sizeof( gGradientPalettes ) / sizeof( TProgmemRGBGradientPalettePtr );

```



---

# Wiring Diagram



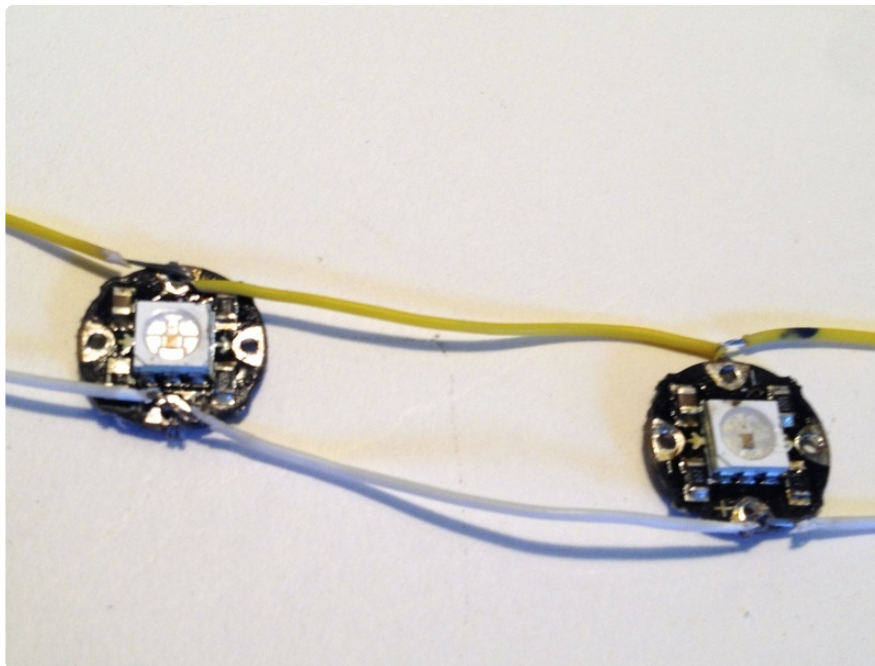
The wiring for this project is very simple. Power and ground split from the battery and go to the Teensy and the neopixels, and one data wire hooks the neopixels to Teensy pin 12.

This is a schematic layout of the parts, to clearly show all the connections, and not representative of the actual placement of all the parts or wires. We'll detail that in the following pages.



## LED Assembly





## Create your LED Strand

This parasol will need to open and close hundreds of times. A traditional LED strand won't stand up to this; LED strands are just not made to stretch and flex the way an umbrella does.

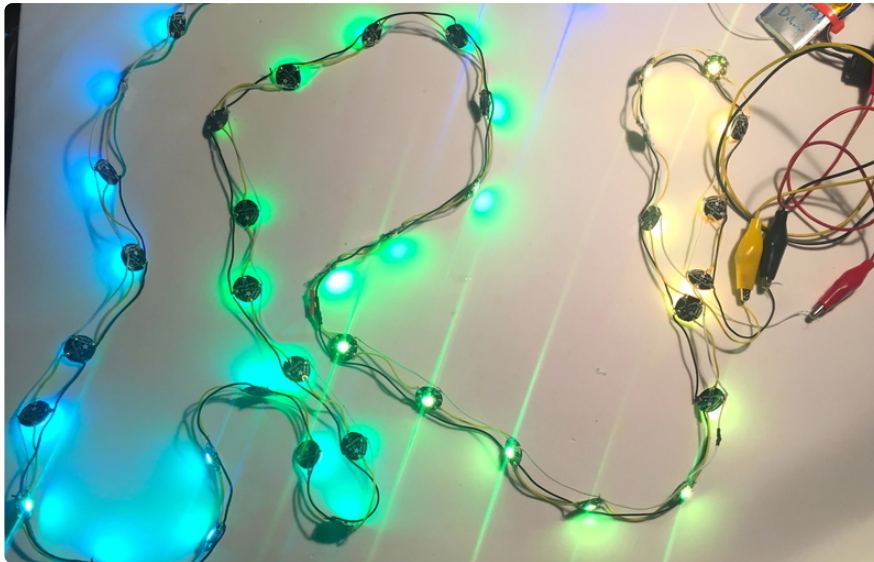
Using individual neopixels soldered together into a long strand works much better and gives you more control over the spacing between lights. I like to use flexible stranded silicone wire to string them together. This wire will keep your parasol working for years even with lots of use.

Here's a video showing my favorite method of quickly stringing neopixels together.

Measure around the circumference of your parasol and decide what spacing you'd like for your lights. Mark and strip your wires, then slide the neopixels on and solder them in place. Finally, connect all the data lines between the neopixels, as shown in the video, making sure you're following the directional arrows on the pixels.

Once you've got all the pixels strung together, hook them up to a microcontroller running the Neopixel Strandtest code to make sure everything is working.

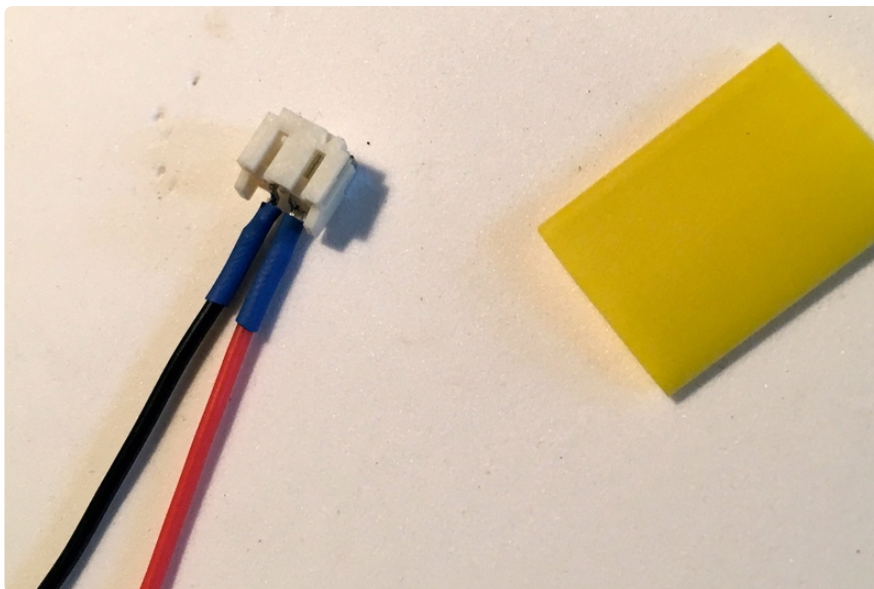




## Power Connector

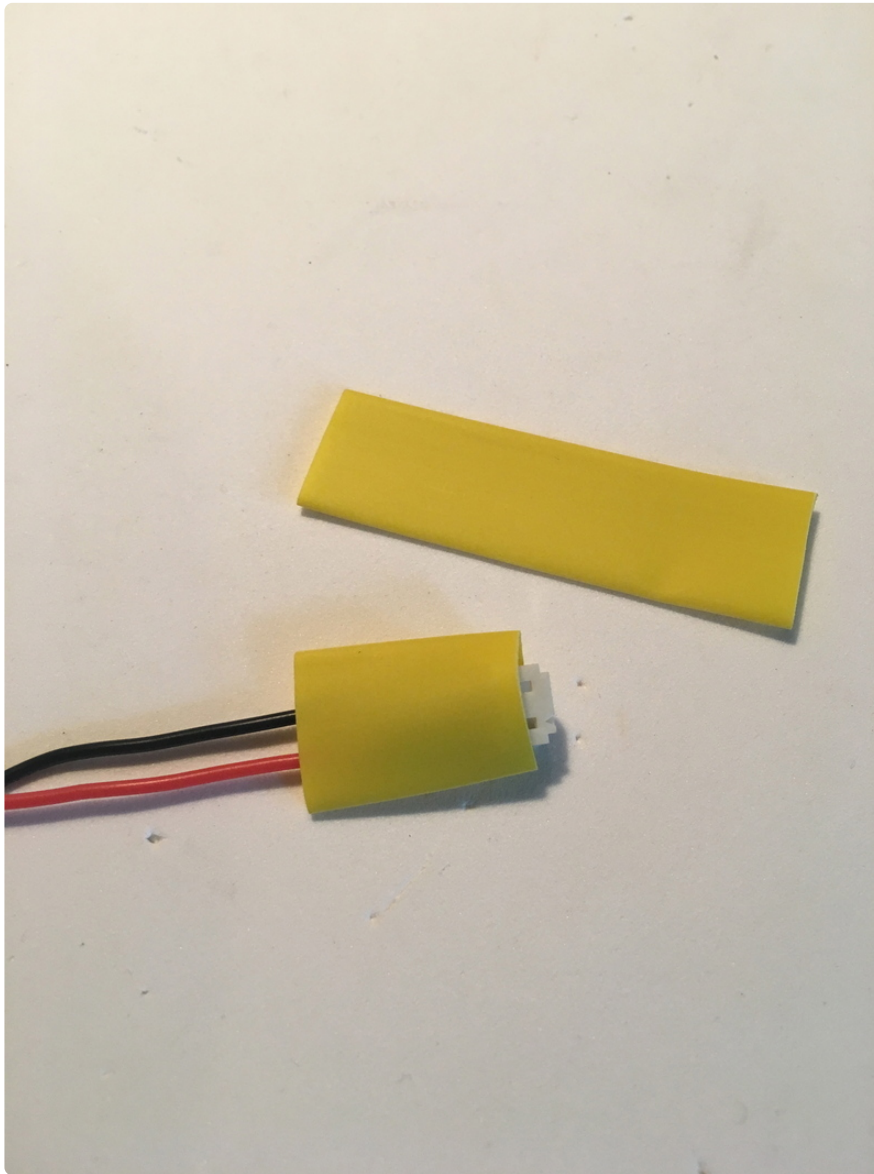
Plug your male JST connector into the female plug coming from your battery and take careful note of which side is power and which is ground.

Slide a small piece of heat shrink onto one of the power and one of the ground wires you just attached to your LED strand. Carefully solder each of these wires onto the JST connector and cover with the heat shrink.

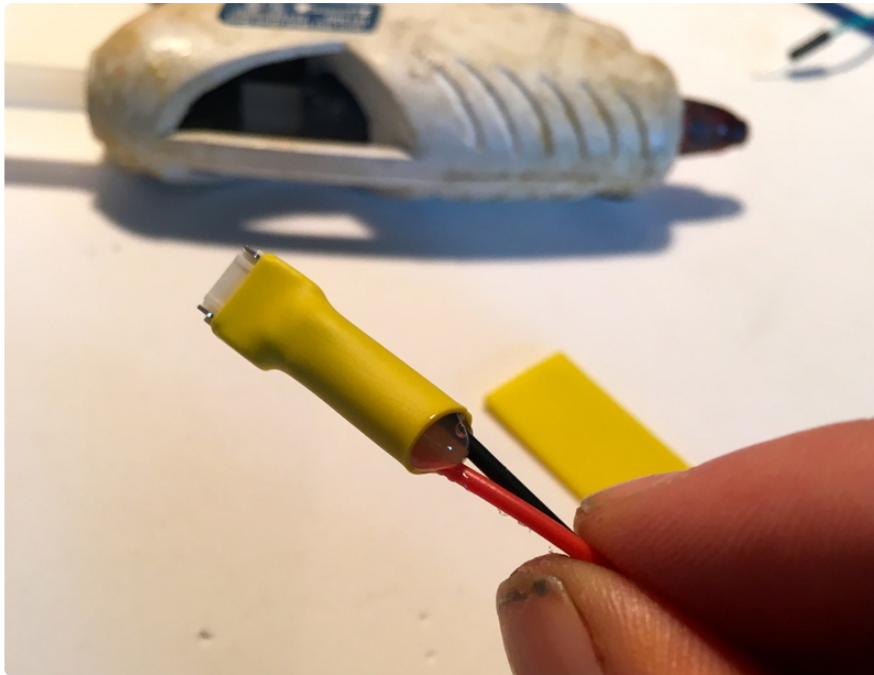


Now we're going to add hardcore strain relief to this connector. It's the piece that takes the most tugging and wear and tear, so we want to be sure it won't break while you're changing batteries (in the dark, in the middle of the night, most likely).

Slide a large 1/2" or 3/8" piece of heat shrink over the connector, leaving the front open but covering the back and the connections you just made completely. Don't heat it yet!



Take a hot glue gun and fill the back side of the heat shrink with hot glue. Then, while your glue is still liquid, hit the whole assembly with a heat gun to shrink the heat shrink in a controlled manner. Be careful not to get glue inside the connector itself.

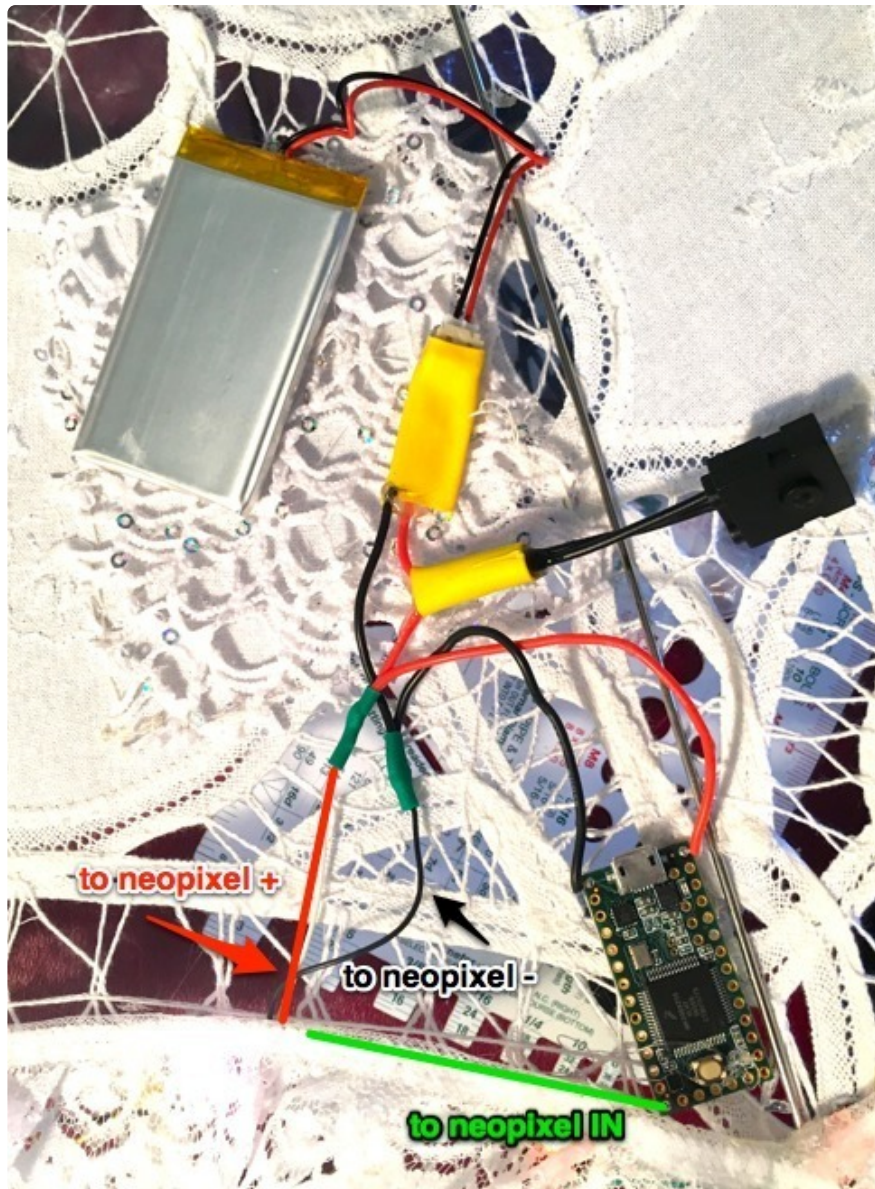


Once this sets up, your connector will be very tug-proof and will hold up beautifully to night time battery fumbling.

## Power Switch

Connect the red wire from your power connector to one of the leads on your power switch. Connect another red wire to the other lead.





## Power Wires

Splice two wires each onto the ground wire (coming from the power connector) and the red wire (coming from the power switch). Connect one of these wires to the neopixel strand + and -, and the other to Vin and G on the Teensy.

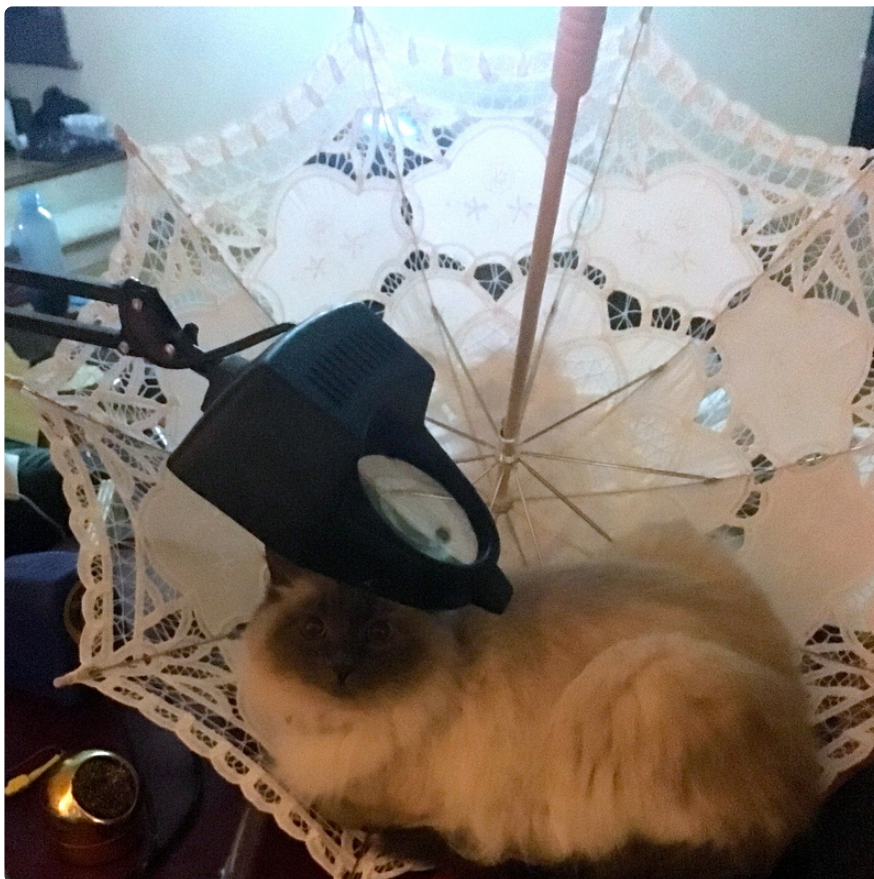
Plug your battery in and watch the lights twinkle.





---

## Parasol Assembly

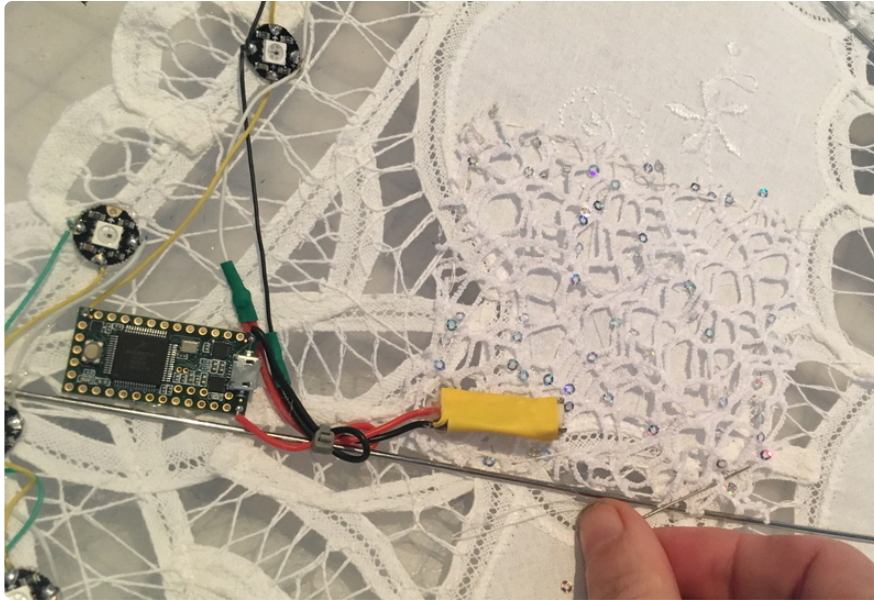


Open your parasol and remove any cats.

Glue or stitch the teensy next to (not on top of) one of the parasol's spokes.

Set the LEDs facing inwards along the edge of your parasol. You can use glue or a needle and thread to secure them. Hot glue works great for this, and goes on quickly, but may not last as long or be as robust as a good fabric embellishing glue.

Or, you can take the time to hand-stitch the LEDs to the parasol using a needle and thread. This takes a bit longer but will very likely hold up the best in the long run.



Take a small piece of netting or fabric and make a pocket for the battery along one of the parasol's spokes. I like using netting since I can poke the connector through one of the netting's holes and it'll stay neatly in place.



Open and close your parasol a few times to make sure everything works and stays put.

**Take note: This parasol is NOT WATERPROOF.** If you are adding these LEDs to a true umbrella instead of a lacy parasol, you may want to add some plastic or waterproof fabric to your electronics so you can use it in the rain.





If you want, finish up by gluing or sewing lace trim over the top of the LED strand to hide the wiring. The lace will diffuse the lights and give a softer and more finished look. It also makes the parasol look a bit brighter from the back since some of the light is being reflected back onto the parasol's top.



---

# Creating Color Palettes

## FastLED Color Palettes

Using the FastLED library opens up a world of LED coding tools that are fairly easy to use.

There's a fantastic tool called **PaletteKnife** that works great for artist-minded people who think in colors instead of in code, and this project uses it.

Watch the video above for a demo of just how easy this is.

### Want to try PaletteKnife and grab some new color palettes?

Here's what you'll definitely need:

1. A basic understanding of how FastLED palettes are used. Check out the FastLED ColorPalette example, as well as FastLED Fire2012WithPalette example, both of which can be found in the [FastLED library \(https://adafru.it/ezy\)](https://adafru.it/ezy) Examples folder.
2. You'll need Chrome or Safari. PaletteKnife doesn't work with Firefox or IE.
3. You will also need a little patience; it's not terribly slick but it works a dream.

Basically, you browse to the color palette that you want, and click PaletteKnife (a bookmarklet) in your browser. Then you COPY the resulting code, and PASTE it into your FastLED animation source code.

Full instructions on how to get started with PaletteKnife are here:

<http://fastled.io/tools/paletteknife/> (<https://adafru.it/mcn>)

