CircuitPython Animated Holiday Wreath Lights

Created by Kattni Rembor

https://learn.adafruit.com/circuitpython-animated-holiday-wreath-lights

Last updated on 2023-08-29 04:33:33 PM EDT
# Table of Contents

## Overview
- Parts
- Options for Power
- Additional Parts

## Assembly

## Install CircuitPython
- Set Up CircuitPython
- Option 1 - Load with UF2 Bootloader
- Try Launching UF2 Bootloader
- Option 2 - Use esptool to load BIN file
- Option 3 - Use Chrome Browser To Upload BIN file

## Code
- Installing Project Code
- Customisations

## Usage
Overview

It's that time of the year again! And, what better way to enjoy the season than with the Adafruit MagTag and NeoPixel LEDs! You can light it up and display an image on the handy eInk display.

This project is a super simple way to turn your MagTag into a festive light display. There is no soldering required - the MagTag comes with a convenient JST port designed to work with 3-pin JST NeoPixel strips. You simply need to plug in the strip to get started. CircuitPython makes programming it really easy. Plug it into your computer, copy the libraries, edit the code, and you're all set to go.

This guide will show you how to assemble a light-up wreath using the Adafruit MagTag and a one meter NeoPixel strip. It will explain how to install CircuitPython and the necessary libraries. Finally, it will show you all of the customisable options available to you in the code so you can make the project your own.

This project requires more power than can be provided by a small lipoly battery, so it is suggested that you power it from a wall plug using an appropriate USB type C cable or adapter.
Parts

Adafruit MagTag - 2.9" Grayscale E-Ink WiFi Display
The Adafruit MagTag combines the new ESP32-S2 wireless module and a 2.9" grayscale E-Ink display to make a low-power IoT display that can show data on its screen even when power...
https://www.adafruit.com/product/4800

Mini Magnet Feet for RGB LED Matrices (Pack of 4)
Got a glorious RGB Matrix project you want to mount and display in your workspace or home? If you have one of the matrix panels listed below, you'll need a pack of these...
https://www.adafruit.com/product/4631

Adafruit NeoPixel LED Strip with 3-pin JST Connector - 1 meter
Plug in and glow, this Adafruit NeoPixel LED Strip with JST PH Connector has 30 total LEDs and is 1 meter long, in classy Adafruit...
https://www.adafruit.com/product/4801

Alternately, instead of purchasing the MagTag and Mini Magnet Feet separately, you could purchase the MagTag starter kit, which includes the magnet feet. Either way, you still need to purchase the LED strip separately.
Adafruit MagTag Starter Kit - ADABOX017 Essentials
The Adafruit MagTag combines the new ESP32-S2 wireless module and a 2.9” grayscale E-Ink display to make a low-power IoT display that can show data on its screen...
https://www.adafruit.com/product/4819

Options for Power

- Plug it into a wall plug using an appropriate cable or adapter. One possible combination is shown below.

  USB Type A to Type C Cable - approx 1 meter / 3 ft long
  As technology changes and adapts, so does Adafruit. This USB Type A to Type C cable will help you with the transition to USB C, even if you’re still...
  https://www.adafruit.com/product/4474

  5V 2A Switching Power Supply w/ USB-A Connector
  Our 5V 2A USB power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that’s power-hungry! This adapter was...
Additional Parts

- Wreath or other festive item to hold LEDs and MagTag. You can be creative here!
- Zip-ties or other appropriate mounting option to attach the LEDs and MagTag to the chosen festive item.

Assembly

This project is super easy to put together. Let's get started!

Gather together the necessary parts.

Attach the mini magnet feet to the MagTag by screwing them into the standoffs in the corners on the back.
Plug the JST connector on the LED strip into JST port on the MagTag labeled D10.

Lay the strip out how you want to attach it. This will give you an idea of how many zip ties you'll need, and where you'll want to attach them.
Start by securing the two ends together snugly. This will make securing the rest of the strip much easier. Then place a zip tie intermittently around the strip on the rest of the wreath. You can leave them relatively loose so you have some ability to move the strip around if necessary.

Secure the MagTag by nestling it into the wreath and wrapping a few of the wreath bits around the magnet feet, or if desired, you can use zip ties.

Fluff the wreath to hide the strip.

Use a wall power supply with enough current to power the project and a USB C cable - plug these into the MagTag.

Install CircuitPython

CircuitPython () is a derivative of MicroPython () designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get
prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the CIRCUITPY drive to iterate.

Set Up CircuitPython

Follow the steps to get CircuitPython installed on your MagTag.

Click the link above and download the latest .BIN and .UF2 file

(performing on how you program the ESP32S2 board you may need one or the other, might as well get both)

Download and save it to your desktop (or wherever is handy).
Plug your MagTag into your computer using a known-good USB cable.

A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

Option 1 - Load with UF2 Bootloader

This is by far the easiest way to load CircuitPython. However it requires your board has the UF2 bootloader installed. Some early boards do not (we hadn't written UF2 yet!) - in which case you can load using the built in ROM bootloader.

Still, try this first!

Try Launching UF2 Bootloader

Loading CircuitPython by drag-n-drop UF2 bootloader is the easier way and we recommend it. If you have a MagTag where the front of the board is black, your MagTag came with UF2 already on it.

Launch UF2 by double-clicking the Reset button (the one next to the USB C port). You may have to try a few times to get the timing right.
If the UF2 bootloader is installed, you will see a new disk drive appear called MAGTAGBOOT.

Copy the UF2 file you downloaded at the first step of this tutorial onto the MAGTAGBOOT drive.

If you're using Windows and you get an error at the end of the file copy that says Error 0x800701B1: A device which does not exist was specified. You can ignore this error, the bootloader sometimes disconnects without telling Windows, the install completed just fine and you can continue. If it's really annoying, you can also upgrade the bootloader (the latest version of the UF2 bootloader fixes this warning).

Your board should auto-reset into CircuitPython, or you may need to press reset. A CIRCUITPY drive will appear. You're done! Go to the next pages.
Option 2 - Use esptool to load BIN file

If you have an original MagTag with while soldermask on the front, we didn't have UF2 written for the ESP32S2 yet so it will not come with the UF2 bootloader.

You can upload with esptool to the ROM (hardware) bootloader instead!

Follow the initial steps found in the Run esptool and check connection section of the ROM Bootloader page () to verify your environment is set up, your board is successfully connected, and which port it's using.

In the final command to write a binary file to the board, replace the port with your port, and replace "firmware.bin" with the file you downloaded above.

The output should look something like the output in the image.

Press reset to exit the bootloader.

Your CIRCUITPY drive should appear!

You’re all set! Go to the next pages.

Option 3 - Use Chrome Browser To Upload BIN file

If for some reason you cannot get esptool to run, you can always try using the Chrome-browser version of esptool we have written. This is handy if you don’t have Python on your computer, or something is really weird with your setup that makes esptool not run (which happens sometimes and isn’t worth debugging!) You can follow
along on the [Web Serial ESPTool](#) page and either load the UF2 bootloader and then come back to Option 1 on this page, or you can download the CircuitPython BIN file directly using the tool in the same manner as the bootloader.

## Code

This project is coded using CircuitPython which makes it super simple to edit. There are a number of customisations you can update to make this project fit your needs. If you also want to change the style and type of LED animations, there is a guide on using the CircuitPython LED Animation library that covers all the features.

This page will show you how to customise a few things about the three included animations, including LED colors. However, before the code will run, you need to load the LED Animation library onto your CIRCUITPY drive.

If you're having difficulty running this example, it could be because your MagTag CircuitPython firmware or library needs to be upgraded! Please be sure to follow [https://learn.adafruit.com/adafruit-magtag/circuitpython](https://learn.adafruit.com/adafruit-magtag/circuitpython) to install the latest CircuitPython firmware and then also replace/update ALL the MagTag-specific libraries mentioned here [https://learn.adafruit.com/adafruit-magtag/circuitpython-libraries-2](https://learn.adafruit.com/adafruit-magtag/circuitpython-libraries-2)

### Installing Project Code

To use with CircuitPython, you need to first install a few libraries, into the lib folder on your CIRCUITPY drive. Then you need to update code.py with the example script.

Thankfully, we can do this in one go. In the example below, click the Download Project Bundle button below to download the necessary libraries and the code.py file.
in a zip file. Extract the contents of the zip file, open the directory MagTag_CircuitPython_Smart_Holiday_Lights/ and then click on the directory that matches the version of CircuitPython you're using and copy the contents of that directory to your CIRCUITPY drive.

Your CIRCUITPY drive should now look similar to the following image:

```
# SPDX-FileCopyrightText: 2020 Kattni Rembor for Adafruit Industries
# SPDX-License-Identifier: MIT

import board
import digitalio
import neopixel
from adafruit_magtag.magtag import MagTag
from adafruit_led_animation.animation.comet import Comet
from adafruit_led_animation.animation.sparkle import Sparkle
from adafruit_led_animation.animation.solid import Solid
from adafruit_led_animation.animation.colorcycle import ColorCycle
from adafruit_led_animation.sequence import AnimationSequence, AnimateOnce
from adafruit_led_animation.group import AnimationGroup
from adafruit_led_animation.color import RED, GREEN, BLUE, WHITE, GOLD

# =============== CUSTOMISATIONS ================
# The pin to which you connected your NeoPixel strip.
strip_pin = board.D10
# The number of NeoPixels on the strip.
strip_num = 30

# The MagTag LED brightness, where 0.0 is 0% (off) and 1.0 is 100% brightness, e.g. 0.3 is 30%.
pixel_brightness = 0.5
# The strip LED brightness, where 0.0 is 0% (off) and 1.0 is 100% brightness, e.g. 0.3 is 30%.
strip_brightness = 1
```
# IF YOU ADD NEW COLORS, YOU MUST IMPORT THEM.
# The colors to cycle through. Can be any number of colors.
color_cycle_colors = (RED, GREEN)
# The speed of the color cycle in seconds. Decreasing speeds it up, increasing
# slows it down.
cycle_speed = 0.5

# The sparkle color.
sparkle_color = GOLD
# The sparkle speed in seconds. Decreasing speeds it up, increasing slows it down.
sparkle_speed = 0.1

# The comet colors.
comet_one_color = WHITE
comet_two_color = BLUE

# The speed of the comet on the MagTag NeoPixels.
magtag_comet_speed = 0.06
# The length of the comet tail on the MagTag NeoPixels.
magtag_comet_tail = 3
# The speed of the comet on the strip of NeoPixels.
strip_comet_speed = 0.03
# The length of the comet tail on the strip of NeoPixels.
strip_comet_tail = 15

# Setup MagTag library.
magtag = MagTag()

# Setup pixels.
pixels = magtag.peripherals.neopixels
pixels.brightness = pixel_brightness
magtag.peripherals.neopixel_disable = False
strip = neopixel.NeoPixel(strip_pin, strip_num, brightness=strip_brightness,
auto_write=False)

# Create animations in sequences and groups.
animations = AnimationSequence(
   AnimationGroup(
      ColorCycle(pixels, cycle_speed, color_cycle_colors),
      ColorCycle(strip, cycle_speed, color_cycle_colors),
      sync=True,
   ),
   AnimationGroup(
      Sparkle(pixels, sparkle_speed, sparkle_color, 15),
      Sparkle(strip, sparkle_speed, sparkle_color, 1),
   ),
   AnimationSequence(
      AnimateOnce(
         AnimationGroup(
            Comet(pixels, magtag_comet_speed, comet_one_color,
            tail_length=magtag_comet_tail),
            Comet(strip, strip_comet_speed, comet_one_color,
            tail_length=strip_comet_tail),
         ),
         AnimationGroup(
            Comet(pixels, magtag_comet_speed, comet_two_color,
            tail_length=magtag_comet_tail),
            Comet(strip, strip_comet_speed, comet_two_color,
            tail_length=strip_comet_tail),
         ),
      ),
   ),
   AnimationGroup(
      # Turn the LEDs off.
      Solid(pixels, 0),
      Solid(strip, 0),
   ),
   auto_clear=True,
The code loads the background image and the three separate text elements. The display will refresh, and there will be a pause before the LEDs begin color cycling. This is expected behavior.

There are a few things that you can easily update to customise this project to fit your aesthetic and hardware setup. Let's take a look!
Customisations

Hardware Setup

The first thing you can set is the pin to which you connected your NeoPixel strip and the number of NeoPixels on the strip.

```python
strip_pin = board.D10
strip_num = 30
```

The code defaults to pin D10 (which is the JST port on the left side of the bottom of the MagTag if you're looking at it from the back), and 30 pixels (which matches the two JST-ready NeoPixel strips carried in the Adafruit shop).

You can also set the pixel brightness for both the MagTag LEDs and the LED strip separately. Depending on how you mounted the LED strip, you may want to make it brighter than the MagTag LEDs.

```python
pixel_brightness = 0.5
strip_brightness = 1
```

NeoPixels can get really bright. The code defaults to 0.5, or 50% brightness, for the MagTag LEDs, and 1, or 100% brightness, for the strip. The strip cannot get any brighter than 100%, however you can increase the MagTag LED brightness number to make them brighter. You can decrease either of the numbers to make them dimmer, however, be aware that the gold color used for the sparkle animation looks different when the LEDs are very dim.

Available Colors

The colors available by default are RED, GREEN, BLUE, WHITE, and GOLD. The LED Animation library has more colors available, however, if you wish to use them, you must also import them at the top of the file. If, for example, you wanted to add MAGENTA, you would need to change the following import.

```python
from adafruit_led_animation.color import RED, GREEN, BLUE, WHITE, GOLD, MAGENTA
```

Once imported at the top, it is available for you to use in any of the animations.
Color Cycle Settings

The first animation cycles between two colors - red and green. You can change this easily by updating the following.

```
color_cycle_colors = (RED, GREEN)
```

To cycle between **RED** and **WHITE** instead, update it to the following.

```
color_cycle_colors = (RED, WHITE)
```

You can also cycle between more than two colors by adding more colors to the list.

```
color_cycle_colors = (RED, GREEN, BLUE, WHITE)
```

To change the speed in seconds at which the colors change, update the following line. Increasing the number slows down the change, decreasing it speeds it up.

```
cycle_speed = 0.5
```
Sparkle Settings

The sparkle animation defaults to \textcolor{red}{GOLD}. You can change this by updating the following.

\begin{quote}
\texttt{sparkle\_color = GOLD}
\end{quote}

To change the sparkle speed in seconds, update the following. Increasing it slows down the animation, decreasing it speeds it up.

\begin{quote}
\texttt{sparkle\_speed = 0.1}
\end{quote}

Comet Settings

There are two alternating comets. They default to \textcolor{red}{WHITE} and \textcolor{red}{BLUE}. You can change this by updating the following.

\begin{quote}
\end{quote}
The comets display differently on the strip and the MagTag LEDs, so there are separate settings for each set of LEDs.

First, you set the speed in seconds of the comet on the MagTag LEDs, and the tail length. The speed defaults to 0.06 and the tail length defaults to 3. You can change this by updating the following.

```
magtag_comet_speed = 0.06
magtag_comet_tail = 3
```

Finally, you set the speed in seconds of the comet on the strip, and the tail length. The speed defaults to 0.03 and the tail length defaults to 15. You can change this by updating the following.

```
strip_comet_speed = 0.03
strip_comet_tail = 15
```

That's all there is to customising your MagTag CircuitPython Smart Holiday Lights!

---

**Usage**

Once the LED strip and MagTag are mounted, and the background image and code are loaded onto the MagTag, using this project is super easy! Press the first three buttons for three different animations, and the fourth button to turn off the LEDs. Let's take a look!

The button functions are displayed on the built in eInk display.
Once the image and text are loaded, the LEDs will automatically begin color cycling between the chosen colors (red and green by default).

Press button B to begin the sparkle animation.

Press button C to begin the comet animation.
Press button A to return to the color cycle animation.

Finally, press button D to turn off the LEDs.

That's all there is to using the MagTag CircuitPython Smart Holiday Lights!