Square NeoPixel Display with Black LED Acrylic

Created by Ruiz Brothers

https://learn.adafruit.com/sqaure-neopixel-display-with-black-led-acrylic

Last updated on 2021-12-08 04:58:26 PM EST
# Table of Contents

## Overview
- NeoMatrix Square Pixels
- Parts

## Circuit Diagram
- Adafruit Library for Fritzing
- Wired Connections
- Powering

## 3D Printing
- Parts List
- CAD assembly
- Slicing Parts
- Design Source Files

## CircuitPython on Feather M4 Express
- Set up CircuitPython Quick Start!
- Further Information

## CircuitPython Libraries
- The Adafruit CircuitPython Library Bundle
- Downloading the Adafruit CircuitPython Library Bundle
- The CircuitPython Community Library Bundle
- Downloading the CircuitPython Community Library Bundle
- Understanding the Bundle
- Example Files
- Copying Libraries to Your Board
- Understanding Which Libraries to Install
- Example: ImportError Due to Missing Library
- Library Install on Non-Express Boards
- Updating CircuitPython Libraries and Examples

## Code
- Coding
- Download the Adafruit CircuitPython Library Bundle
- Required Libraries
- The Mu Python Editor
- Installing or upgrading CircuitPython
- Adjust Code
- Upload Code
- Customizing LED Animations

## Acrylic Cutting
- Acrylic Template
- Desktop CNC
- Tool Library
- 2D Contour
- Otherplan

## Wiring
- Feather Headers
• Wiring Switch 29
• Wiring JST Cable 29
• Wiring Proto FeatherWing 29
• Switch Wired 30
• 3-Pin JST-PH Cable 30
• Wired Proto FeatherWing 30
• Wiring NeoMatrix 31
• Wired NeoMatrix 31
• Test Circuit 31

Assembly 32
• Hardware for Proto FeatherWing 32
• Install FeatherWing 32
• Secure FeatherWing PCB 33
• Standoffs (Optional for Battery Power) 33
• Install Switch 34
• Install NeoMatrix 34
• Installed NeoMatrix 35
• Install Grid 35
• Install Acrylic 36
• Install Cover 36
• Connect NeoMatrix to FeatherWing 36
• USB Power 37
• Final Build 37
Overview

NeoMatrix Square Pixels
Build a square pixel display with Adafruit's 8x8 NeoMatrix and Feather M4. Use Black LED acrylic and 3D printed grid to create an evenly diffused LED effect. Electronics are housed in an elegant looking snap fit case with built-in on/off switch. LED animations are easily customizable with CircuitPython.

Parts

Adafruit Feather M4 Express - Featuring ATSAMD51
It's what you've been waiting for, the Feather M4 Express featuring ATSAMD51. This Feather is fast like a swift, smart like an owl, strong like a ox-bird (it's half ox,...
https://www.adafruit.com/product/3857

Black LED Diffusion Acrylic Panel 12" x 12" - 0.1" / 2.6mm thick
A nice whoppin' slab of some lovely black acrylic to add some extra diffusion to your LED Matrix project. This material is 2.6mm (0.1") thick and is made of special cast...
https://www.adafruit.com/product/4594
Adafruit NeoPixel NeoMatrix 8x8 - 64 RGB LED Pixel Matrix
Put on your sunglasses before wiring up this LED matrix - 64 eye-blistering RGB LEDs adorn the NeoMatrix for a blast of configurable color. Arranged in an 8x8 matrix, each pixel is...
https://www.adafruit.com/product/1487

FeatherWing Proto - Prototyping Add-on For All Feather Boards
A Feather board without ambition is a Feather board without FeatherWings! This is the FeatherWing Proto - a prototyping add-on for all Feather boards. Using our...
https://www.adafruit.com/product/2884

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 x 1200mAh Battery</td>
<td>3.7V Lipo Battery</td>
<td><a href="https://www.adafruit.com/product/258">https://www.adafruit.com/product/258</a></td>
</tr>
<tr>
<td>1 x Slide Switch</td>
<td>Slide Switch</td>
<td><a href="https://www.adafruit.com/product/805">https://www.adafruit.com/product/805</a></td>
</tr>
<tr>
<td>1 x Header Kit for Feathers</td>
<td>12-pin and 16-pin Female Headers</td>
<td><a href="https://www.adafruit.com/product/2886">https://www.adafruit.com/product/2886</a></td>
</tr>
<tr>
<td>1 x 10-Wire Ribbon Cable</td>
<td>28AWG Silicone Cover Stranded Core</td>
<td><a href="https://www.adafruit.com/product/3890">https://www.adafruit.com/product/3890</a></td>
</tr>
<tr>
<td>1 x JST PH 3-pin Plug-Plug Cable</td>
<td>100mm long</td>
<td><a href="https://www.adafruit.com/product/4336">https://www.adafruit.com/product/4336</a></td>
</tr>
<tr>
<td>1 x JST PH 3-Pin Socket</td>
<td>Color Coded Cable - 200mm</td>
<td><a href="https://www.adafruit.com/product/4046">https://www.adafruit.com/product/4046</a></td>
</tr>
<tr>
<td>1 x M2.5 Hardware Kit</td>
<td>Black Nylon Screw and Stand-off Set</td>
<td><a href="https://www.adafruit.com/product/3299">https://www.adafruit.com/product/3299</a></td>
</tr>
</tbody>
</table>
1 x USB Cable
Fully Reversible Pink/Purple USB A to micro B Cable - 1m

[Image of a USB cable]

Circuit Diagram

The diagram below provides a visual reference for wiring of the components. This diagram was created using the software package Fritzing (https://adafruit.it/oEP).

Adafruit Library for Fritzing

Use Adafruit's Fritzing parts library to create circuit diagrams for your projects. Download the library or just grab individual parts. Get the library and parts from GitHub - Adafruit Fritzing Parts (https://adafruit.it/AYZ).
Wired Connections

The Feather M4 Express is installed on top of the Proto FeatherWing using Headers. The NeoMatrix is wired to the Proto FeatherWing so that the Feather can be easily swapped.

- 5V from NeoMatrix to 3V on Proto FeatherWing
- GND from NeoMatrix to GND on Proto FeatherWing
- DIN from NeoMatrix to Pin #6 on Proto FeatherWing
- Switch to EN and GND on Proto FeatherWing

Powering

The Adafruit board can be powered via USB or JST using a 3.7v lipo battery. In this project, a 1200mAh lipo battery is used. The lipo battery is rechargeable via the USB port on the board. The switch is wired to the enable and ground pins on the board.
3D Printing

Parts List
STL files for 3D printing are oriented to print "as-is" on FDM style machines. Parts are designed to 3D print without any support material. Original design source may be downloaded using the links below.

File names
- cover.stl
- pcb-mount.stl
- neomatrix-grid.stl
- acrylic.stl

Fusion 30 Share Link
https://adafru.it/MOe

Download CAD files from Prusa Printers
https://adafru.it/MOf

Download CAD files from Thingiverse
https://adafru.it/MOA

CAD assembly
The Proto FeatherWing is secured to the pcb-mount with M2.5 hardware screws (standoffs are optional). The Feather M4 snaps into the female headers on the Proto FeatherWing. The NeoMatrix is fitted into the built-in standoffs in the pcb-mount. The neomatrix-grid is fitted over the NeoMatrix PCB. The piece of acrylic is placed inside the cover. The cover is fitted over the neomatrix-grid and snap fits onto the edges of the pcb-mount.
Slicing Parts
No supports are required. Slice with setting for PLA material.

The parts were sliced using CURA using the slice settings below.

- PLA filament 220c extruder
- 0.2 layer height
- 10% gyroid infill
- 60mm/s print speed
- 60c heated bed

Design Source Files
The project assembly was designed in Fusion 360. This can be downloaded in different formats like STEP, STL and more. Electronic components like Adafruit's board, displays, connectors and more can be downloaded from the Adafruit CAD parts GitHub Repo (https://adafruit.it/AW8).

CircuitPython on Feather M4 Express

CircuitPython (https://adafruit.it/tB7) is a derivative of MicroPython (https://adafruit.it/BeZ) 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.

The following instructions will show you how to install CircuitPython. If you've already installed CircuitPython but are looking to update it or reinstall it, the same steps work for that as well!

Set up CircuitPython Quick Start!

Follow this quick step-by-step for super-fast Python power :)
Download the latest version of CircuitPython for this board via CircuitPython.org
https://adafruit.it/Emh

Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).

Plug your Feather M4 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.

Double-click the Reset button next to the USB connector on your board, and you will see the NeoPixel RGB LED turn green. If it turns red, check the USB cable, try another USB port, etc. Note: The little red LED next to the USB connector will pulse red. That’s ok!

If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!
You will see a new disk drive appear called FEATHERBOOT.

Drag the adafruit_circuitpython_etc.uf2 file to FEATHERBOOT.

The LED will flash. Then, the FEATHERBOOT drive will disappear and a new disk drive called CIRCUITPY will appear.

That's it, you're done! :)

Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython (https://adafruit.it/Amd).
CircuitPython Libraries

As CircuitPython development continues and there are new releases, Adafruit will stop supporting older releases. Visit https://circuitpython.org/downloads to download the latest version of CircuitPython for your board. You must download the CircuitPython Library Bundle that matches your version of CircuitPython. Please update CircuitPython and then visit https://circuitpython.org/libraries to download the latest Library Bundle.

Each CircuitPython program you run needs to have a lot of information to work. The reason CircuitPython is so simple to use is that most of that information is stored in other files and works in the background. These files are called libraries. Some of them are built into CircuitPython. Others are stored on your CIRCUITPY drive in a folder called lib. Part of what makes CircuitPython so great is its ability to store code separately from the firmware itself. Storing code separately from the firmware makes it easier to update both the code you write and the libraries you depend.

Your board may ship with a lib folder already, it’s in the base directory of the drive. If not, simply create the folder yourself. When you first install CircuitPython, an empty lib directory will be created for you.

CircuitPython libraries work in the same way as regular Python modules so the Python docs (https://adafru.it/rar) are an excellent reference for how it all should work. In Python terms, you can place our library files in the lib directory because it’s part of the Python path by default.

One downside of this approach of separate libraries is that they are not built in. To use them, one needs to copy them to the CIRCUITPY drive before they can be used. Fortunately, there is a library bundle.
The bundle and the library releases on GitHub also feature optimized versions of the libraries with the .mpy file extension. These files take less space on the drive and have a smaller memory footprint as they are loaded.

Due to the regular updates and space constraints, Adafruit does not ship boards with the entire bundle. Therefore, you will need to load the libraries you need when you begin working with your board. You can find example code in the guides for your board that depends on external libraries.

Either way, as you start to explore CircuitPython, you'll want to know how to get libraries on board.

**The Adafruit CircuitPython Library Bundle**

Adafruit provides CircuitPython libraries for much of the hardware they provide, including sensors, breakouts and more. To eliminate the need for searching for each library individually, the libraries are available together in the Adafruit CircuitPython Library Bundle. The bundle contains all the files needed to use each library.

**Downloading the Adafruit CircuitPython Library Bundle**

You can download the latest Adafruit CircuitPython Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

Match up the bundle version with the version of CircuitPython you are running. For example, you would download the 6.x library bundle if you're running any version of CircuitPython 6, or the 7.x library bundle if you're running any version of CircuitPython 7, etc. If you mix libraries with major CircuitPython versions, you will get incompatible mpy errors due to changes in library interfaces possible during major version changes.

[Click to visit circuitpython.org for the latest Adafruit CircuitPython Library Bundle](https://adafru.it/ENC)
Download the bundle version that matches your CircuitPython firmware version. If you don't know the version, check the version info in boot_out.txt file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.

There's also a py bundle which contains the uncompressed python files, you probably don't want that unless you are doing advanced work on libraries.

The CircuitPython Community Library Bundle

The CircuitPython Community Library Bundle is made up of libraries written and provided by members of the CircuitPython community. These libraries are often written when community members encountered hardware not supported in the Adafruit Bundle, or to support a personal project. The authors all chose to submit these libraries to the Community Bundle make them available to the community.

These libraries are maintained by their authors and are not supported by Adafruit. As you would with any library, if you run into problems, feel free to file an issue on the GitHub repo for the library. Bear in mind, though, that most of these libraries are supported by a single person and you should be patient about receiving a response. Remember, these folks are not paid by Adafruit, and are volunteering their personal time when possible to provide support.

Downloading the CircuitPython Community Library Bundle

You can download the latest CircuitPython Community Library Bundle release by clicking the button below. The libraries are being constantly updated and improved, so you'll always want to download the latest bundle.

Click for the latest CircuitPython Community Library Bundle release
https://adafru.it/VCn

The link takes you to the latest release of the CircuitPython Community Library Bundle on GitHub. There are multiple versions of the bundle available. Download the bundle version that matches your CircuitPython firmware version. If you don't know the version, check the version info in boot_out.txt file on the CIRCUITPY drive, or the initial prompt in the CircuitPython REPL. For example, if you're running v7.0.0, download the 7.x library bundle.
Understanding the Bundle

After downloading the zip, extract its contents. This is usually done by double clicking on the zip. On Mac OSX, it places the file in the same directory as the zip.

Open the bundle folder. Inside you'll find two information files, and two folders. One folder is the lib bundle, and the other folder is the examples bundle.

Now open the lib folder. When you open the folder, you'll see a large number of .mpy files, and folders.

Example Files

All example files from each library are now included in the bundles in an examples directory (as seen above), as well as an examples-only bundle. These are included for two main reasons:

• Allow for quick testing of devices.
• Provide an example base of code, that is easily built upon for individualized purposes.
Copying Libraries to Your Board

First open the lib folder on your CIRCUITPY drive. Then, open the lib folder you extracted from the downloaded zip. Inside you'll find a number of folders and .mpy files. Find the library you'd like to use, and copy it to the lib folder on CIRCUITPY.

If the library is a directory with multiple .mpy files in it, be sure to copy the entire folder to CIRCUITPY/lib.

This also applies to example files. Open the examples folder you extracted from the downloaded zip, and copy the applicable file to your CIRCUITPY drive. Then, rename it to code.py to run it.

If a library has multiple .mpy files contained in a folder, be sure to copy the entire folder to CIRCUITPY/lib.

Understanding Which Libraries to Install

You now know how to load libraries on to your CircuitPython-compatible microcontroller board. You may now be wondering, how do you know which libraries you need to install? Unfortunately, it's not always straightforward. Fortunately, there is an obvious place to start, and a relatively simple way to figure out the rest. First up: the best place to start.

When you look at most CircuitPython examples, you'll see they begin with one or more `import` statements. These typically look like the following:

- `import library_or_module`

However, `import` statements can also sometimes look like the following:

- `from library_or_module import name`
They can also have more complicated formats, such as including a `try / except` block, etc.

The important thing to know is that an `import` statement will always include the name of the module or library that you're importing.

Therefore, the best place to start is by reading through the `import` statements.

Here is an example import list for you to work with in this section. There is no setup or other code shown here, as the purpose of this section involves only the import list.

```python
import time
import board
import neopixel
import adafruit_lis3dh
import usb_hid
from adafruit_hid.consumer_control import ConsumerControl
from adafruit_hid.consumer_control_code import ConsumerControlCode
```

Keep in mind, not all imported items are libraries. Some of them are almost always built-in CircuitPython modules. How do you know the difference? Time to visit the REPL.

In the Interacting with the REPL section (https://adafru.it/Awz) on The REPL page (https://adafru.it/Awz) in this guide, the `help("modules")` command is discussed. This command provides a list of all of the built-in modules available in CircuitPython for your board. So, if you connect to the serial console on your board, and enter the REPL, you can run `help("modules")` to see what modules are available for your board. Then, as you read through the `import` statements, you can, for the purposes of figuring out which libraries to load, ignore the statement that import modules.

The following is the list of modules built into CircuitPython for the Feather RP2040. Your list may look similar or be anything down to a significant subset of this list for smaller boards.
Now that you know what you’re looking for, it’s time to read through the import statements. The first two, `time` and `board`, are on the modules list above, so they’re built-in.

The next one, `neopixel`, is not on the module list. That means it’s your first library! So, you would head over to the bundle zip you downloaded, and search for neopixel. There is a neopixel.mpy file in the bundle zip. Copy it over to the lib folder on your CIRCUITPY drive. The following one, `adafruit_lis3dh`, is also not on the module list. Follow the same process for adafruit_lis3dh, where you’ll find adafruit_lis3dh.mpy, and copy that over.

The fifth one is `usb_hid`, and it is in the modules list, so it is built in. Often all of the built-in modules come first in the import list, but sometimes they don’t! Don’t assume that everything after the first library is also a library, and verify each import with the modules list to be sure. Otherwise, you’ll search the bundle and come up empty!

The final two imports are not as clear. Remember, when `import` statements are formatted like this, the first thing after the `from` is the library name. In this case, the library name is `adafruit_hid`. A search of the bundle will find an adafruit_hid folder. When a library is a folder, you must copy the entire folder and its contents as it is in the bundle to the lib folder on your CIRCUITPY drive. In this case, you would copy the entire adafruit_hid folder to your CIRCUITPY/lib folder.

Notice that there are two imports that begin with `adafruit_hid`. Sometimes you will need to import more than one thing from the same library. Regardless of how many times you import the same library, you only need to load the library by copying over the adafruit_hid folder once.

That is how you can use your example code to figure out what libraries to load on your CircuitPython-compatible board!
There are cases, however, where libraries require other libraries internally. The internally required library is called a dependency. In the event of library dependencies, the easiest way to figure out what other libraries are required is to connect to the serial console and follow along with the `ImportError` printed there. The following is a very simple example of an `ImportError`, but the concept is the same for any missing library.

**Example: ImportError Due to Missing Library**

If you choose to load libraries as you need them, or you're starting fresh with an existing example, you may end up with code that tries to use a library you haven't yet loaded. This section will demonstrate what happens when you try to utilise a library that you don't have loaded on your board, and cover the steps required to resolve the issue.

This demonstration will only return an error if you do not have the required library loaded into the lib folder on your CIRCUITPY drive.

Let's use a modified version of the Blink example.

```python
import board
import time
import simpleio
led = simpleio.DigitalOut(board.LED)

while True:
    led.value = True
    time.sleep(0.5)
    led.value = False
    time.sleep(0.5)
```

Save this file. Nothing happens to your board. Let's check the serial console to see what's going on.

You have an `ImportError`. It says there is no module named `'simpleio'`. That's the one you just included in your code!
Click the link above to download the correct bundle. Extract the lib folder from the downloaded bundle file. Scroll down to find simpleio.mpy. This is the library file you're looking for! Follow the steps above to load an individual library file.

The LED starts blinking again! Let's check the serial console.

![Serial Console](image)

No errors! Excellent. You've successfully resolved an `ImportError`!

If you run into this error in the future, follow along with the steps above and choose the library that matches the one you're missing.

**Library Install on Non-Express Boards**

If you have an M0 non-Express board such as Trinket M0, Gemma M0, QT Py M0, or one of the M0 Trinkeys, you'll want to follow the same steps in the example above to install libraries as you need them. Remember, you don't need to wait for an `ImportError` if you know what library you added to your code. Open the library bundle you downloaded, find the library you need, and drag it to the lib folder on your CIRCUITPY drive.

You can still end up running out of space on your M0 non-Express board even if you only load libraries as you need them. There are a number of steps you can use to try to resolve this issue. You'll find suggestions on the [Troubleshooting page](https://adafruit.it/Den).

**Updating CircuitPython Libraries and Examples**

Libraries and examples are updated from time to time, and it's important to update the files you have on your CIRCUITPY drive.

To update a single library or example, follow the same steps above. When you drag the library file to your lib folder, it will ask if you want to replace it. Say yes. That's it!
A new library bundle is released every time there's an update to a library. Updates include things like bug fixes and new features. It's important to check in every so often to see if the libraries you’re using have been updated.

**Code**

This project uses the LED animation rainbow example code from the CircuitPython LED Animation guide. The code is modified slightly to accommodate the additional pixels used in the sign.

**Download the Adafruit CircuitPython Library Bundle**

In order to run the code, we'll need to download a few libraries. Libraries contain code to help interface with hardware a lot easier for us.

Use the [Adafruit M4 Express page on Installing Libraries](https://adafruit.it/CVs) to get the library that matches the major version of CircuitPython you are using noted above.

To run the code for this project, we need the three libraries in the Required Libraries list below. Unzip the library bundle and search for the libraries. Drag and drop the files into a folder named lib on the CIRCUITPY drive (which appears when your board is plugged into your computer via a known good USB cable) if the directory is not already on the Feather M4 Express).

**Required Libraries**

- neopixel.mpy
- adafruitLed_animation
The Mu Python Editor

Mu is a simple Python editor that works with Adafruit CircuitPython hardware. It's written in Python and works on Windows, MacOS, Linux and Raspberry Pi. The serial console is built right in, so you get immediate feedback from your board's serial output! While you can use any text editor with your code, Mu makes it super simple. Instructions for Mu are available here (https://adafruit.it/ANO).

Installing or upgrading CircuitPython

You should ensure you have CircuitPython 5.0 or greater on your board. Plug your board in with a known good data + power cable (not the cheesy USB cable that comes with USB power packs, they are power only). You should see a new flash drive pop up.

If the drive is CIRCUITPY, then open the boot_out.txt file to ensure the version number is 5.0 or greater.

Adafruit CircuitPython 5.3.1 on 2020-07-13; Adafruit Feather M4 Express with samd51j19

Click on the download link below to grab the project code directly from GitHub.

# SPDX-License-Identifier: MIT

This example shows usage of the PixelMap helper to easily treat a single strip as a horizontal or
vertical grid for animation purposes.

For NeoPixel FeatherWing. Update pixel_pin and pixel_num to match your wiring if using a different form of NeoPixels. Note that if you are using a number of pixels other than 32, you will need to alter the PixelMap values as well for this example to work.

This example does not work on SAMD21 (M0) boards.

```python
import board
import neopixel
from adafruit_led_animation.animation.comet import Comet
from adafruit_led_animation.animation.rainbowcomet import RainbowComet
from adafruit_led_animation.animation.rainbowchase import RainbowChase
from adafruit_led_animation.animation.rainbow import Rainbow
from adafruit_led_animation.sequence import AnimationSequence
from adafruit_led_animation.color import PURPLE, JADE, AMBER

# Update to match the pin connected to your NeoPixels
pixel_pin = board.D6
# Update to match the number of NeoPixels you have connected
pixel_num = 32

pixels = neopixel.NeoPixel(pixel_pin, pixel_num, brightness=0.5, auto_write=False)

pixel_wing_vertical = helper.PixelMap.vertical_lines(
    pixels, 8, 4, helper.horizontal_strip_gridmap(8, alternating=False)
)
pixel_wing_horizontal = helper.PixelMap.horizontal_lines(
    pixels, 8, 4, helper.horizontal_strip_gridmap(8, alternating=False)
)

comet_h = Comet(
    pixel_wing_horizontal, speed=0.1, color=PURPLE, tail_length=3, bounce=True
)
comet_v = Comet(pixel_wing_vertical, speed=0.1, color=AMBER, tail_length=6, bounce=True)
chase_h = Chase(pixel_wing_horizontal, speed=0.1, size=3, spacing=6, color=JADE)
rainbow_chase_v = RainbowChase(
    pixel_wing_vertical, speed=0.1, size=3, spacing=2, step=8
)
rainbow_comet_v = RainbowComet(
    pixel_wing_vertical, speed=0.1, tail_length=7, bounce=True
)
rainbow_v = Rainbow(pixel_wing_vertical, speed=0.1, period=2)
rainbow_chase_h = RainbowChase(pixel_wing_horizontal, speed=0.1, size=3, spacing=3)

animations = AnimationSequence(
    rainbow_v,
    comet_h,
    rainbow_comet_v,
    chase_h,
    rainbow_chase_v,
    comet_v,
    rainbow_chase_h,
    advance_interval=5,
)

while True:
    animations.animate()
```
Adjust Code

Look for the following lines in the code and adjust to fit the 8x8 NeoMatrix. Use any text editor or favorite IDE to modify the code. We suggest using Mu as noted above.

```python
# Update to match the pin connected to your NeoPixels
pixel_pin = board.D6
# Update to match the number of NeoPixels you have connected
pixel_num = 64

pixel_wing_vertical = helper.PixelMap.vertical_lines(
    pixels, 8, 8, helper.horizontal_strip_gridmap(8, alternating=False)
)
pixel_wing_horizontal = helper.PixelMap.horizontal_lines(
    pixels, 8, 8, helper.horizontal_strip_gridmap(8, alternating=False)
)
```

Upload Code

Ensure the file is named code.py and drop it onto the CIRCUITPY drive main (root) directory that appears when your Feather is plugged into your computer via a known good USB data cable. The code will run properly when all of the files have been uploaded including libraries.

Customizing LED Animations

Take a moment to walk through the LED animation library for CircuitPython learn guide. The guide covers the how to create animations with horizontal and vertical grids.

Learn Guide: CircuitPython LED Animations – Pixel Mapping
https://adafru.it/MOB
Acrylic Cutting

Acrylic Template
A piece of stock can be from the 12 x 12in sheet can using a table saw or a scoring tool (plastic cutting knife). Our stock was cut to 114x114m to accommodate the size of the Bantamtools CNC spoilboard. Use the SVG file to CNC mill or laser cut the piece. Optionally print the template for reference.

Download Acrylic Template PDF
https://adafruit.it/MOC

Download Acrylic Outline SVG file
https://adafruit.it/MOD

Desktop CNC

Cutting the Black LED acrylic was done using the Bantam Tools Desktop CNC (https://adafruit.it/Cg-). This guide covers using Fusion 360 and bantam tools software to cut the acrylic.
Tool Library
Download the tool library from the Bantam Tools website to use in Fusion 360. Install the library by using the Tool Library in the manufacture workspace. Select Local in the side bar and click the Import libraries icon. Navigate and choose the bantam tools Fusion 360 tool library. Use the 1/8in flat end mill to cut the piece of acrylic.

2D Contour
In Fusion 360 manufacture workspace, create a New Setup and set stock mode as relative size box. Add a 2mm to the stock side offset. Use the 2D contour operation to cut the shape out the material. Under the Tool tab, select the 1/8in flat end mill from the imported tools under Local library. Under the geometry tab, in the contour selection area, click on the lower edge of the model. Under the Passes tab, enable Multiple Depths and add 0.1mm to Maximum Roughing. Use the recommended settings (https://adafruit.it/MOF) for cutting acrylic with a 1/8in flat end mill. Right click on the setup in the browser select post process. Use the othermill as the processor and name your gcode file accordingly.

- Tool: 1/8" flat end mill
- Feed rate: 59 in/min (1500 mm/min)
- Plunge rate: 1.81 in/min (46 mm/min)
- Spindle speed: 16,400 RPM
- Max pass depth: 0.005" (0.13 mm)
Otherplan

Use the open files button and navigate to the exported gcode file from Fusion 360. Set the material to Generic using the dropdown. Enter the dimensions of your stock acrylic in the material size dropdown. Set the X and Y to 0mm and add a 0.2mm to the Z to accommodate for the double-sided tape used to secure the stock to the spoil board. Offset the placement of the gcode file by 2mm in the X and Y. Set the Z to 0.2mm for stock to leave behind. Select the 1/8in flat end mill. Click start milling button when the stock is secured and the 1/8in end mill is installed and probed.

Wiring

Feather Headers

The Proto FeatherWing is setup with 12-pin and 16-pin female headers. The Feather M4 Express will have a male strip of headers, 12-pin and 16-pin respectively.
Wiring Switch
The slide switch is wired up to a piece of ribbon cable with two connections going to the middle pin and either side. The wire was peeled off from a roll of 10-wire ribbon cable. The cable length is approximately 5cm (2in) in length.

Wiring JST Cable
A male (socket plug) 3-pin JST-PH cable is setup to connect to the Proto FeatherWing. This wire was peeled from a roll of 10-wire ribbon cable. The cable length is roughly 8cm (3in) in length.

Wiring Proto FeatherWing
The slide switch and 3-pin JST cable is soldered to the bottom of the Proto FeatherWing PCB.
Switch Wired
The slide switch is wired to the EN and GND pins on the Proto FeatherWing. The wires are soldered from the bottom of the PCB. Reference the pin labels on the top of the PCB.

3-Pin JST-PH Cable
The 3-pin JST-PH cable is wired to 3V, GND and D6 pins on the Proto FeatherWing. The wires are soldered from the bottom of the PCB. Reference the pin labels on the top of the PCB.

Wired Proto FeatherWing
Double check the wiring is correct.
Wiring NeoMatrix
A 3-pin JST (female socket) cable is wired to the 5V, GND and DIN pins on the NeoMatrix PCB. The red wire is connected to the 5V pin, black to the GND pin and white to the DIN pin on the NeoMatrix. The wires are approximately 5cm (2in) in length.

Wired NeoMatrix
Double check the wiring is correct. Pieces of heat shrink tubing can be used to keep the wires bundled together.

Test Circuit
With the code and libraries uploaded to the Feather, the circuit should power on. Install the Feather on top of the Proto Feather Wing. Plug in the battery to the Feather. Connect the two JST cables together. Use the slide switch to power the circuit on.
Assembly

Hardware for Proto FeatherWing
The Proto FeatherWing is secured to the PCB mount using screws and hex nuts. Use the following hardware.

- 4x M2.5 x 8mm screws
- 4x M2.5 hex nuts

Install FeatherWing
Place the FeatherWing over the four tabs with the mounting holes lined up. Reference the photo for correct placement. While holding in place, insert M2.5 x 8mm screws into mounting holes on the FeatherWing PCB.
Secure FeatherWing PCB
Install and fasten the M2.5 hex nuts to threads of the screws to secure the FeatherWing to the PCB mount. The FeatherWing was mounted with the USB port close to the edge. Double check the placement of the PCB with the Feather M4 installed on top of the Proto FeatherWing.

Standoffs (Optional for Battery Power)
For battery power, standoffs will provide clearance for fitting a battery in between the Proto FeatherWing and NeoMatrix PCB. Use eight M.25 x 6mm long screws and 10mm tall M2.5 standoffs to secure the FeatherWing to the mounting tabs on the case.
Install Switch
The slide switch is inserted at an angle into the built in holder. The actuator pokes through the little hole on the side of the mount. The body of the slide switch will stay in place when the NeoMatrix PCB installed. It essentially holds the switch in place preventing it from coming out the top.

Install NeoMatrix
The NeoMatrix PCB is fitted into the pcb mount with the mounting holes lined up with the built-in standoffs. Orient the PCB so the mounting holes are lined up. Reference the photos for correct placement. Place the NeoMatrix PCB over the standoffs and press down so the pegs are fitted into the mounting holes.
Installed NeoMatrix
The built-in standoffs provide approximately 4mm of clearance. This in turn makes the PCB mount thick (11.5mm) but overall a design decision that accommodates for wiring and connectors.

Install Grid
The grid is designed to fit over the NeoMatrix PCB. The walls in the grid feature cut outs specifically sized for the on-board 0805 capacitors on the NeoMatrix PCB. Orient the grid so the cutouts are lined up with the caps on the PCB. Place the grid over the PCB and press the grid into the PCB mount.
Install Acrylic
The piece of acrylic is press fitted into the cover with the matte side facing down and the glossy side facing up. The lip around the bottom surface keeps the acrylic from falling out.

Install Cover
The cover fits over the grid and PCB mount. Orient the cover so the cutout is lined up with the switch on the PCB mount. Snap fits on the edges grasp onto the nubs on the side of the PCB mount.

Connect NeoMatrix to FeatherWing
Connect the JST cable from the NeoMatrix and Proto FeatherWing together.
USB Power
Connect a USB cable to the USB port on the Feather. Use a computer’s USB port or USB 5V wall adapter to power the Feather. Optionally use a battery.

Final Build
And there you have it! Enjoy your new square pixel display and create something awesome with it!