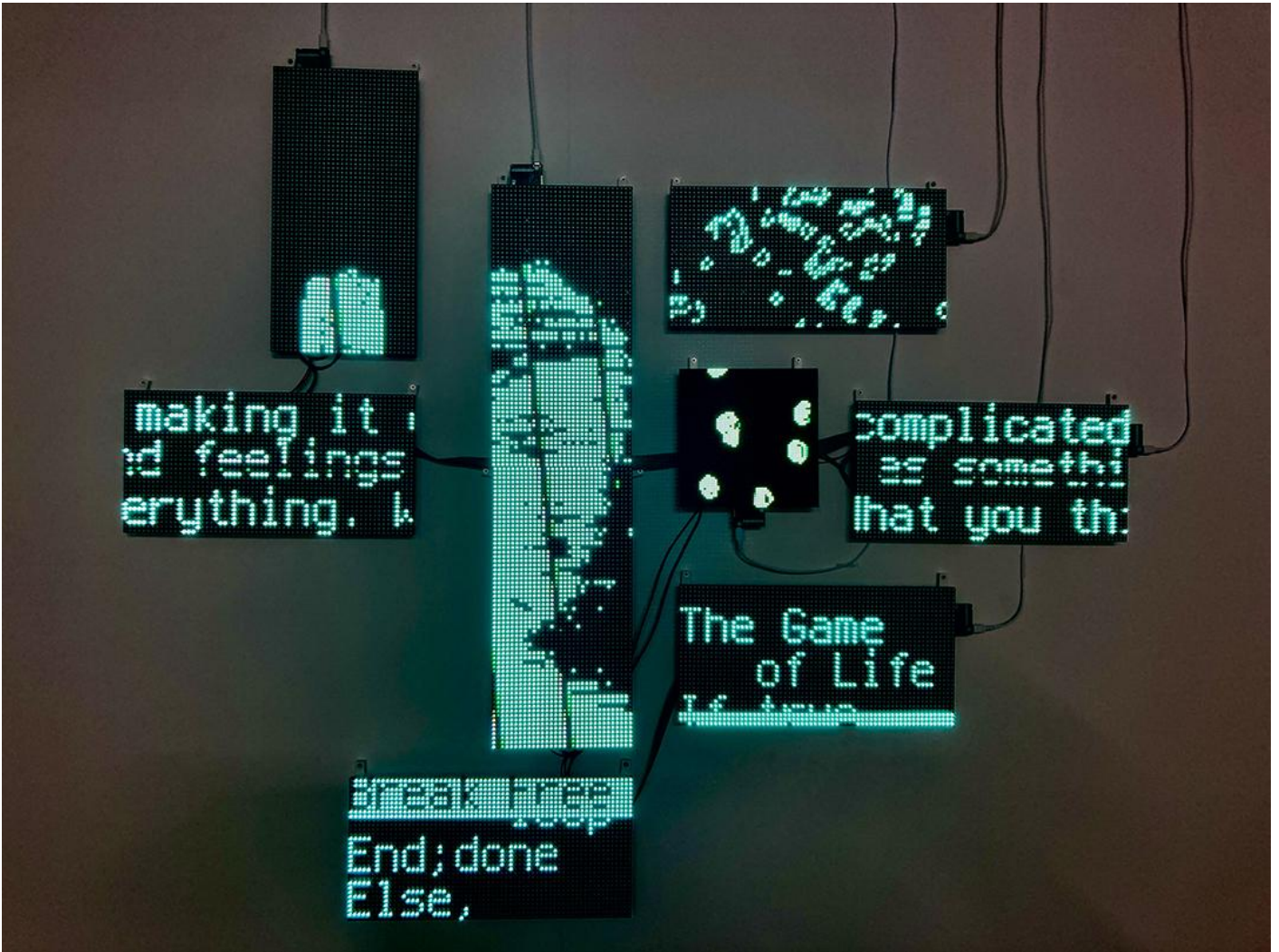


Using MatrixPortal M4 for animation

How to use MatrixPortal M4 for animation

We have another [tutorial](#) for setting up the MatrixPortal board and covering the basics. This tutorial will focus on creating an animation to display on a 64x32 matrix.



CircuitPython and libraries versions

In this tutorial, we are using CircuitPython 9.0.5 (11/10/2024), all libraries and code used are compatible with this version. Please double-check the latest version of CircuitPython you have installed and use updated and compatible libraries.

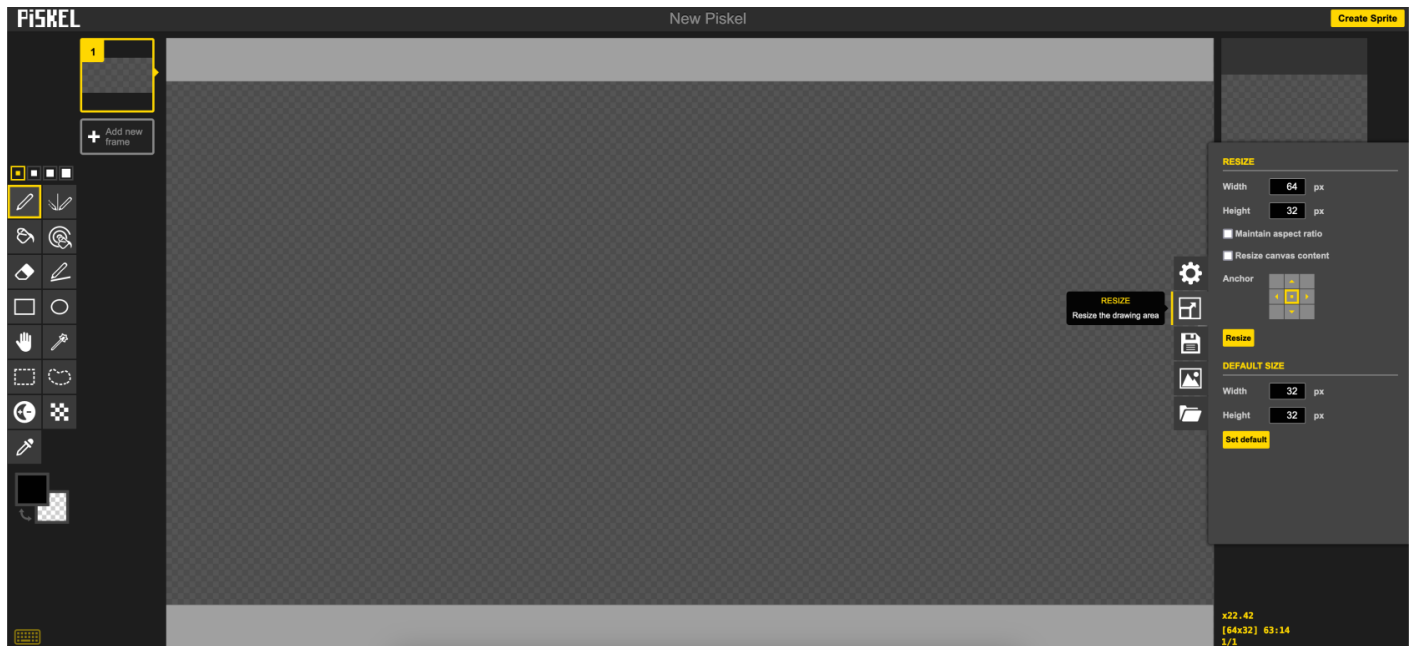
Creating the Spitesheet

A spritesheet is a single image file that contains a collection of smaller images (called sprites) arranged in a grid or some other layout. These individual sprites can represent various frames of an animation, characters, objects, or other visual elements in a video game or graphic application.

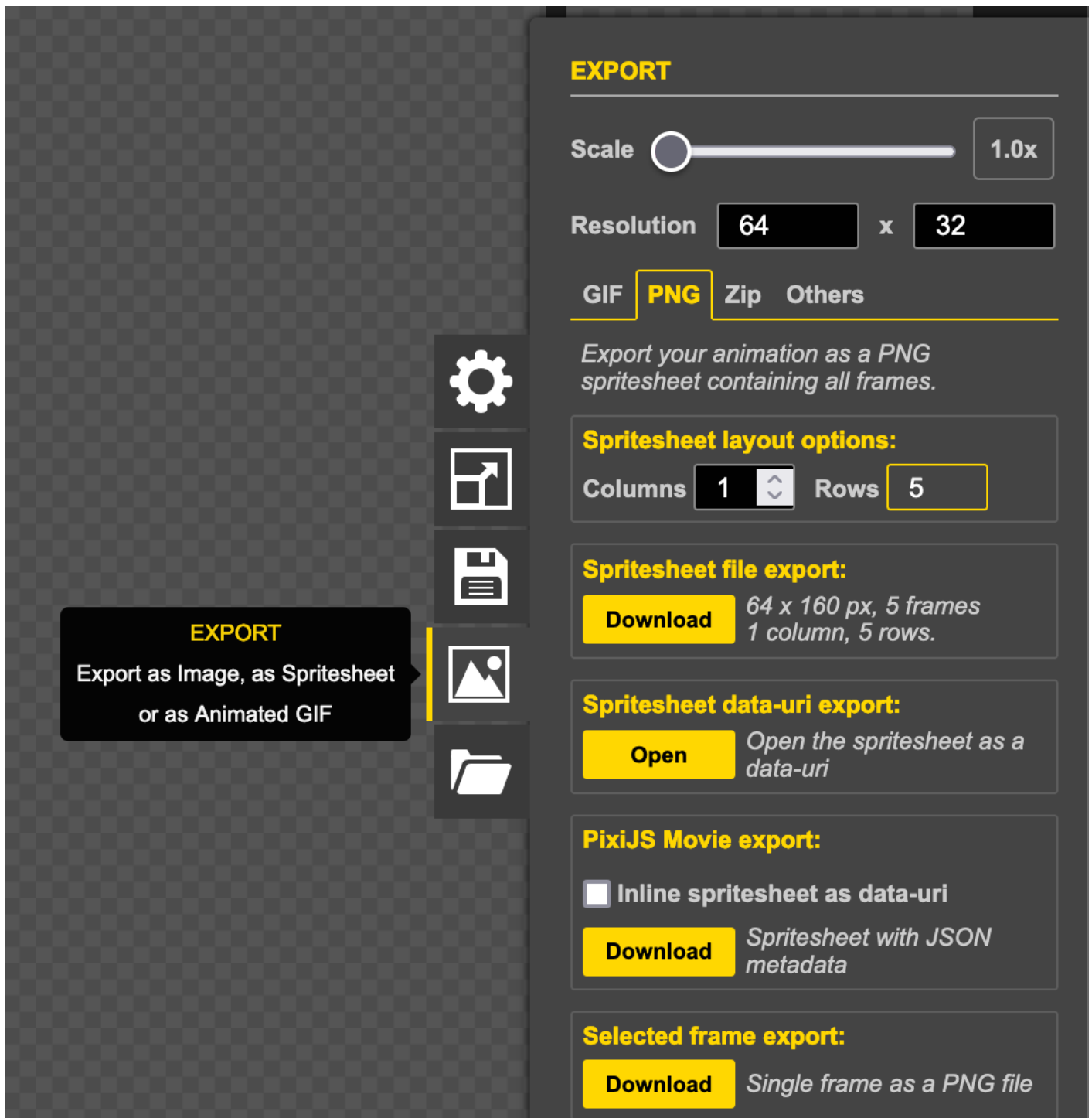
We will need to use a bitmap spritesheet. If you just want to test the code, you can download the `test.bmp` and skip this part for now.

1. Piskel

Piskel is a free online editor for animated sprites & pixel art.



- Go to `resize` to set the canvas size to 64 x 32 px
- Draw whatever you want
- `Add new frame` and draw
- until you finish



- Go to **EXPORT**
- Select **PNG**
- Change **Columns** to 1
- **Download**
- You now have a very long PNG file

2. Photoshop



- Open the image with Photoshop
- Save a copy in BMP format
- Choose `Windows` (doesn't matter what computer you use) and `16 Bit`
- You now have a ready-to-go spitesheet BMP file

Importing the Image

1. Create a folder called `bmp` in the **CIRCUITPY** drive.
2. Copy the bitmap file you have created into the folder.

Code

```
# SPDX-FileCopyrightText: 2020 John Park for Adafruit Industries
#
# SPDX-License-Identifier: MIT
```

```
import time
import os
import board
import displayio

from digitalio import DigitalInOut, Pull
from adafruit_matrixportal.matrix import Matrix
from adafruit_debouncer import Debouncer

SPRITESHEET_FOLDER = "/bmp"
DEFAULT_FRAME_DURATION = 0.1 # 100ms

FRAME_DURATION_OVERRIDES = {
    "KIRBY.bmp": 0.05,
}

# --- Display setup ---
matrix = Matrix(width=64, height=64, bit_depth=6)
sprite_group = displayio.Group()
matrix.display.root_group = sprite_group

file_list = sorted(
    [
        f
        for f in os.listdir(SPRITESHEET_FOLDER)
        if (f.endswith(".bmp") and not f.startswith("."))
    ]
)

if len(file_list) == 0:
    raise RuntimeError("No images found")

current_image = None
current_frame = 0
current_loop = 0
frame_count = 0
frame_duration = DEFAULT_FRAME_DURATION
direction = 1 #1 for forward, -1 for backward

def load_image():
```

```

"""
Load an image as a sprite
"""

# pylint: disable=global-statement
global current_frame, current_loop, frame_count, frame_duration
while sprite_group:
    sprite_group.pop()

filename = SPRITESHEET_FOLDER + "/" + file_list[current_image]

# # CircuitPython 7+ compatible
bitmap = displayio.OnDiskBitmap(filename)
sprite = displayio.TileGrid(
    bitmap,
    pixel_shader=bitmap.pixel_shader,
    tile_width=bitmap.width,
    tile_height=matrix.display.height,
)

sprite_group.append(sprite)

current_frame = 0
current_loop = 0
frame_count = int(bitmap.height / matrix.display.height)
frame_duration = DEFAULT_FRAME_DURATION
if file_list[current_image] in FRAME_DURATION_OVERRIDES:
    frame_duration = FRAME_DURATION_OVERRIDES[file_list[current_image]]
direction = 1

def advance_image():
    """
    Advance to the next image in the list and loop back at the end
    """
    # pylint: disable=global-statement
    global current_image
    if current_image is not None:
        current_image += 1
    if current_image is None or current_image >= len(file_list):
        current_image = 0
    load_image()

```

```
def advance_frame():
    """
    Advance to the next frame and loop back at the end
    """
    # pylint: disable=global-statement
    global current_frame, current_loop, direction
    current_frame += direction
    if current_frame >= frame_count:
        current_frame = frame_count - 1
        direction = -1 # Reverse direction
    elif current_frame < 0:
        current_frame = 0
        direction = 1 # Forward direction
    sprite_group[0][0] = current_frame

advance_image()

while True:
    advance_frame()
    time.sleep(frame_duration)
```

Revision #3

Created 11 October 2024 10:46:03 by Joanne Leung

Updated 16 December 2024 10:30:07 by Joanne Leung