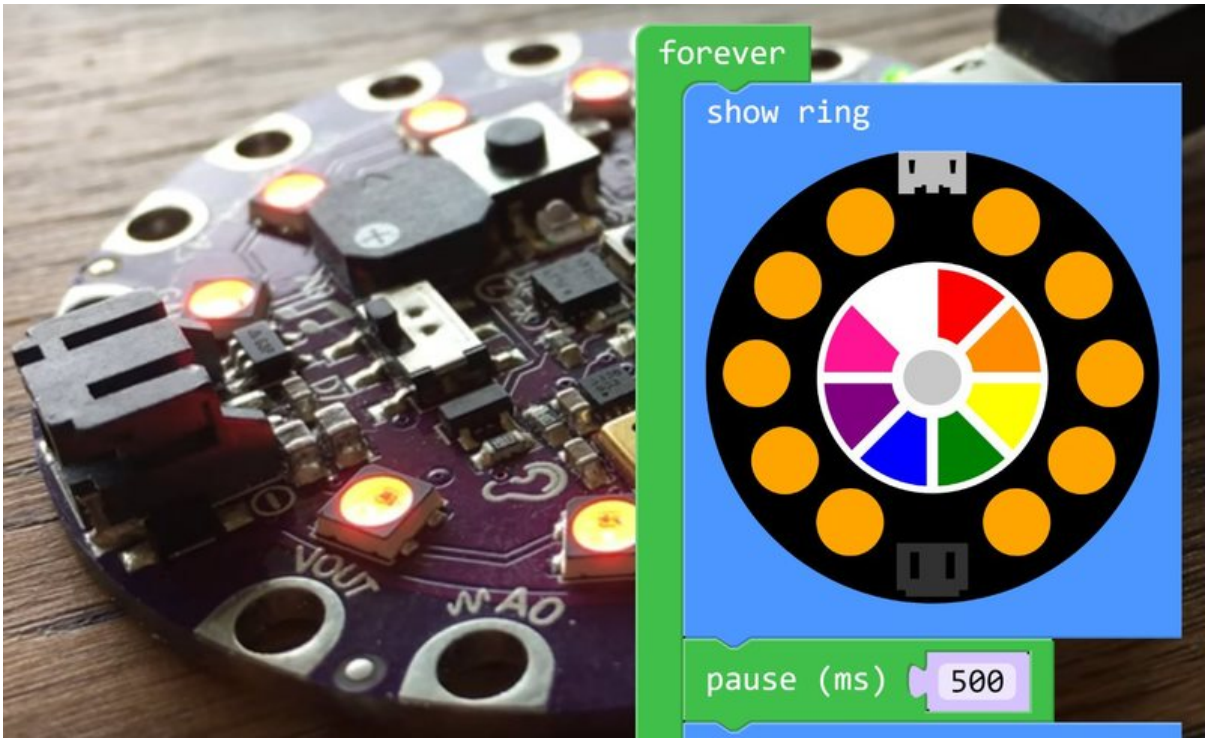
NeoPixels with MakeCode

Created by Peli de Halleux



https://learn.adafruit.com/neopixels-with-makecode

Last updated on 2024-06-03 02:08:01 PM EDT

Table of Contents

Overview

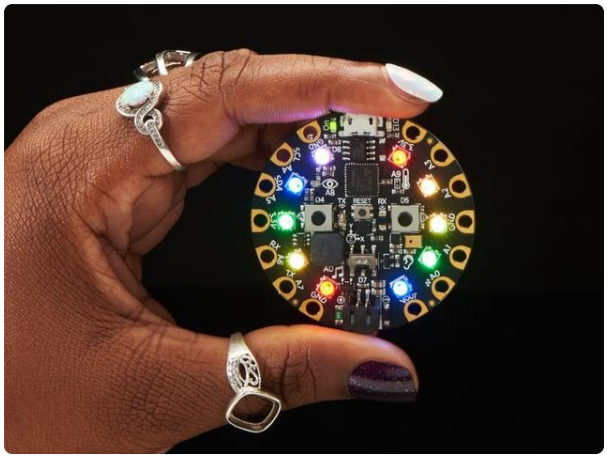
The Circuit Playground Express is equipped with 10 NeoPixel LEDs that can be controlled with code to create tons of cool visual effects. In this guide, you will learn the various ways to use the Microsoft MakeCode editor to use those LEDs. The guide also covers how to use additional NeoPixel strips.

Make sure to read the MakeCode primer if you are not familiar with MakeCode and checkout the Adafruit NeoPixel Uberguide for more details on those amazing LEDs.

Show Ring

The show ring block lets you choose the colors of the 10 LEDs. You can find it find under the `light` category.

You use multiple `show ring` blocks to create basic animations. The example below is a blinking siren animation that repeat in a forever loop.



Circuit Playground Express
Circuit Playground Express is the next step towards a perfect introduction to electronics and programming. We've taken the original Circuit Playground Classic and...
https://www.adafruit.com/product/3333

Built-in Animations

The show animation block provides a set of builtin animations. You can find it find under the `light` category.

The block will run the animation for the requested time, then continue the program. If another animation is already running, it will be queued and wait for its turn to run.

The example below shows the `Rainbow` animation running in a `forever` loop. The runtime maintains an **animation queue** and waits until the animation is fully executed before continuing to run.

Responsive Animations

When you need to run an animation to respond to an event, you'd typically want it to run immediately. To avoid waiting for other animations to finish, make sure add the stop all animations block. This block stops the current animation and clears the animation queue.

In the example below, we want to run the `Sparkle` animation as soon as a `shake` event is detected. To do so, we add a stop all animations block before the `show animation` block.

Frame by frame Animations

In some case, you might want to be able to control frame by frame how the animation runs. This is also possible but requires a bit more setup.

In the example below, we store the `Theater Chase` animation and we show a new frame in a loop if button `A` is pressed.

Photon

Photon is a simple logo turtle for NeoPixels. The `Photon` is a color cursor that leaves a trail color as it moves. You can move Photon forward or backward, change the trail color or lift/lower the pen.

In the example below, we repeat 9 times a move forward and a color shift. Then, we set the photon in eraser mode and have it go back to clear up the circle. This creates a cool effect of bouncing rainbow animation.

To get started, try playing with these two blocks:

- `photon forward` let you move the photon on the LED strip
- `photon set hue` lets you change the color **hue** from 0 to 255.

Photon is inspired from LightLogo from Brian Silverman.

External Strips

MakeCode provides a rich library to more NeoPixel strips (see the API reference) connected on the pins of the Circuit Playground.

If you want to support an external NeoPixel strip, not the 10 built-in LEDs, you can create a strip instance and store it in a variable. You can configure the number and type of LEDs.

```
// mount an external Neopixel strip on pin A1 with 24 RGB pixels
let strip = light.createStrip(pins.A1, 24);
```

The example below mounts a NeoPixel strip on pin **A1** and turns it to blue or red when buttons **A** or **B** are pressed.

To set all the colors on the pixels, use setAll:

```
// show blue on all pixels
strip.setAll(Colors.Blue)
```

You can also use the set pixel color block to set an individual pixel color.

```
// set colors one by one
for(let i = 0; i < strip.length(); ++i) {
    strip.setPixelColor(i, Colors.Green);
}
```

Ranges

When working with long strips of NeoPixel, it is quite useful to chunk them into ranges and apply independent colors and animations to them. The range allows to create a new NeoPixel strip instance over a subset of the pixels.

```
// mount an external Neopixel strip on pin A1 with 24 RGB pixels
let strip = light.createStrip(pins.A1, 24);

// create 2 ranges for each half of the strip
let head = strip.range(0, 12);
let tail = strip.range(12, 12);
```

Once you have your rangers defined, you can easily apply colors and animation without worrying about complicated index computations.

```
// turn on 1 pixel on each range
head.setPixelColor(0, Colors.Blue)
tail.setPixelColor(11, Colors.Red)
```

The example below create 2 ranges and moves the pixels on opposite directions on each of them.

Buffering

By default, all operations on the NeoPixels are immediately transferred to the lights. This works great to get started and for a small number of pixels. However, for large number of lights and for precise control of the timings, you might want to decide when to send the updates to the hardware.

To turn on buffering, call **setBuffered**.

```
...
strip.setBuffered(false)
```

Once buffering is on, you need to call **show** to send the color updates into the NeoPixels.

```
... some operations on the colors
strip.show();
```

Bitmap animations (Beta)

This is a cool - yet still rather experimental feature! This page might change...

NeoAnim is a way to encode NeoPixel animation into bitmaps. It's a surprisingly easy way to design cool animation (once you wrap your head around the concept).

The pxt-neoanim implements this technique for MakeCode. It also comes with a converter that takes an image and turns into the JavaScript code that MakeCode can understand.

- open the converter https://microsoft.github.io/pxt-neoanim/index
- paste your image
- open https://makecode.adafruit.com
- add the **neoanim** package (go to Advanced -> Add Package)
- copy/paste the generated code into MakeCode and run!

The image below is an example of possible animation and the editor shows the animation once converted.