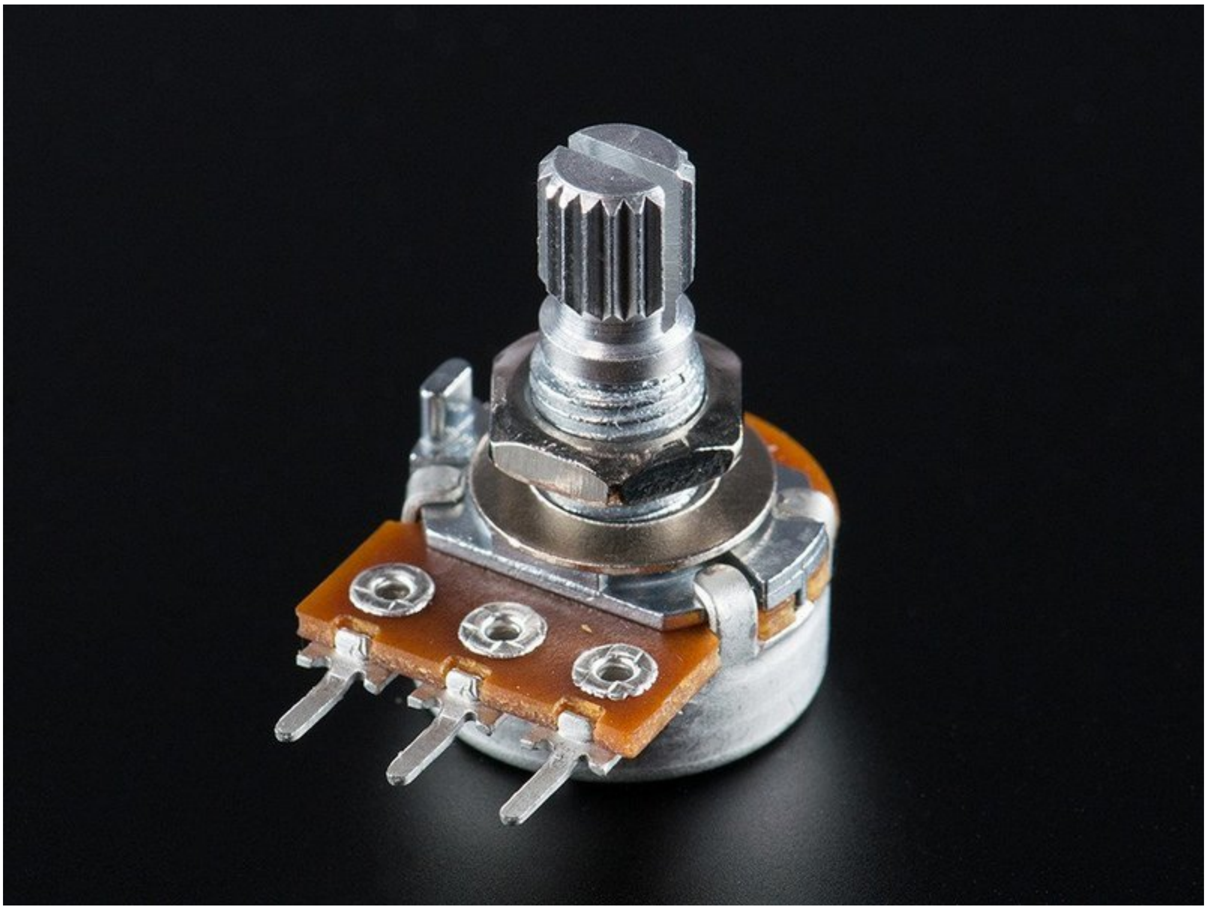
Make It Change: Potentiometers

Created by Anne Barela

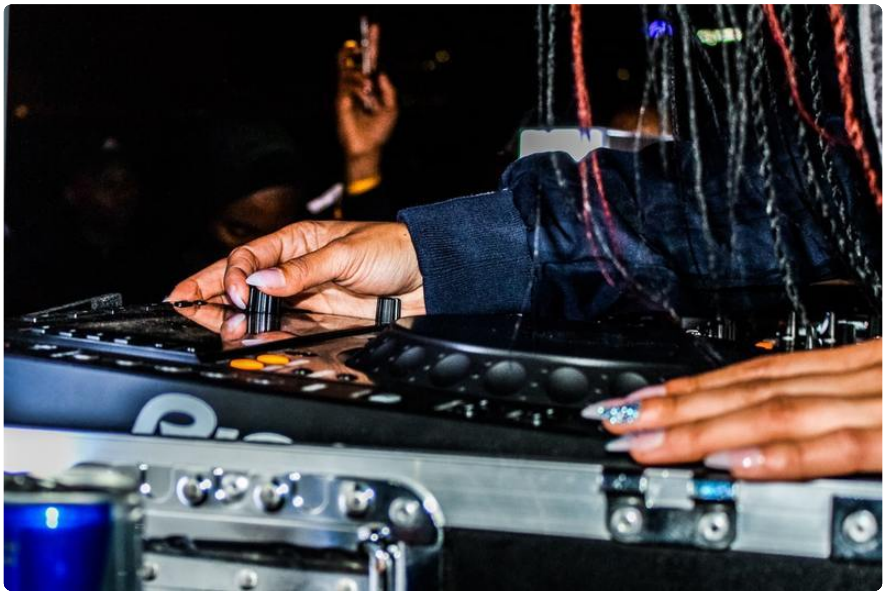


https://learn.adafruit.com/make-it-change-potentiometers

Last updated on 2024-06-03 02:29:19 PM EDT

Table of Contents

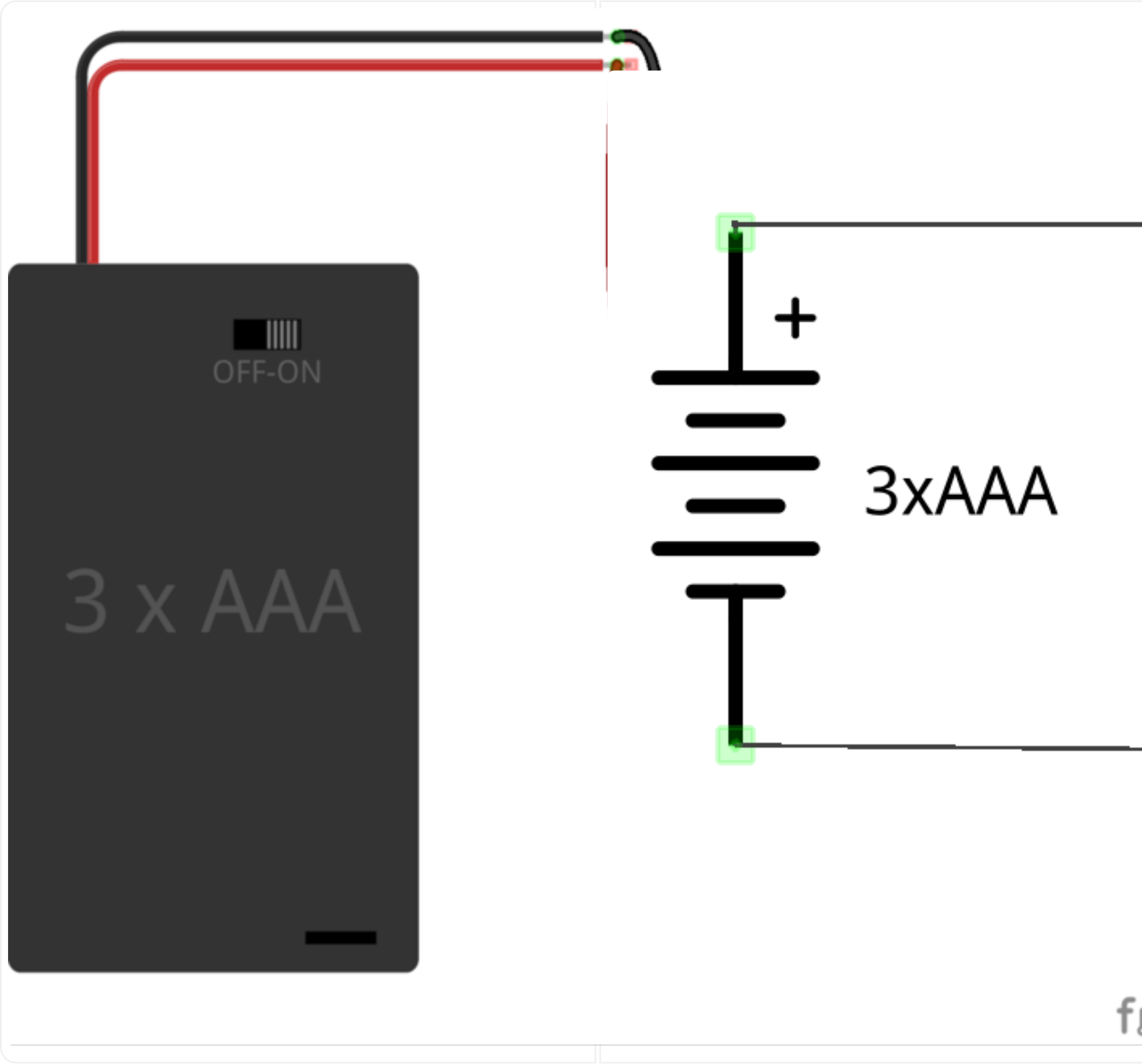
Overview



Our electronics talk to us through lights and displays. But how do we easily communicate back to our electronics. Ok, if touch screens don't count, the second and the oldest way to easily "tell" something to an electronics project is to turn a knob.

Most often, that knob is usually connected to a variable resistor, or potentiometer in electrical-speak.

The term potentiometer comes from the simple circuit shown below.

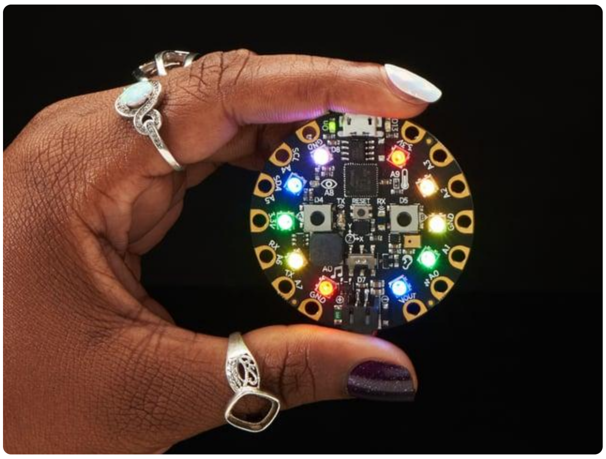


A voltage is placed across the outer, fixed terminals and taken from ground and the middle terminal which is connected to a wiper (a movable arm on top of the resistor). At J1, the output terminals, you read with your meter a voltage from zero to the battery voltage (here 4.5 volts) which changes as you turn the shaft of the potentiometer. **Potential is another name for voltage**, and meter means measure - hence a potentiometer is a way of measuring/changing a voltage.

This guide will help you understand the different types of potentiometers and how you may want to use them in your projects.

While the concepts in this guide can be used with ANY microcontroller or single board computer, they will be demonstrated using the Circuit Playground Express.

Parts List



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

Panel Mount 10K potentiometer (Breadboard Friendly)

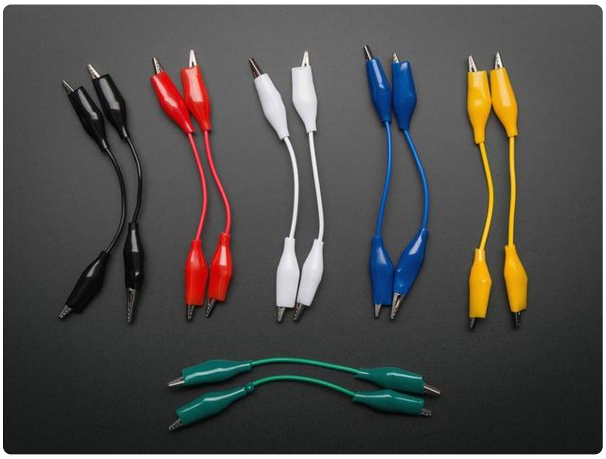
This potentiometer is a two-in-one, good in a breadboard or with a panel. It's a fairly standard linear taper 10K ohm potentiometer, with a grippy shaft. It's smooth and easy...

https://www.adafruit.com/product/562

Potentiometer Knob - Soft Touch T18 - Red

Oh say can you seeBy the knob's early light...Sorry - we thought that was clever. And while it wasn't really, this potentiometer knob...

https://www.adafruit.com/product/2046



Short Wire Alligator Clip Test Lead (set of 12)

Connect this to that without soldering using these handy mini alligator clip test leads. Approximately 4.5" overall cables with alligator clip on each end, color coded. You get 12...

https://www.adafruit.com/product/1592

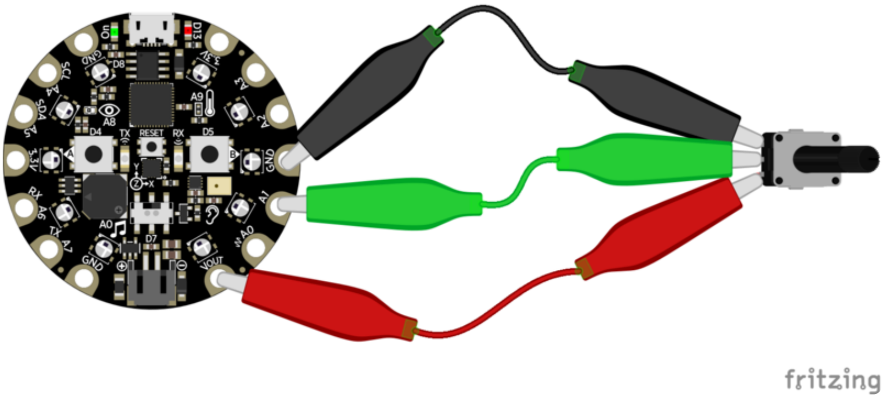
If you'd like to work with a slide potentiometer, this 3.5 cm 10K linear model is nice. As usual with potentiometers and resistors, there are many types. This one is $1.29/each in single quantities at the time this guide was written from digikey.com (https://adafru.it/CwN).

1 x Digi-key Slide Potentiometer

10 kOhms 0.1W, through Hole Slide Potentiometer Top Adjustment Type, 20 mm adjust, 35 mm long

https://www.digikey.com/products/en?keywords=PTA2043-2015CPB103-ND

Connections



The circuit above shows the connections for hooking a potentiometer to a Circuit Playground Express board. The center terminal of the potentiometer (the wiper) is connected to pad **A1** although it could also be connected to pads **A2** through **A7**. We tend to skip **A0** as that is the analog output pin for sound and many potentiometer projects are related to sound for volume control, varying the tempo, and other sound-related features.

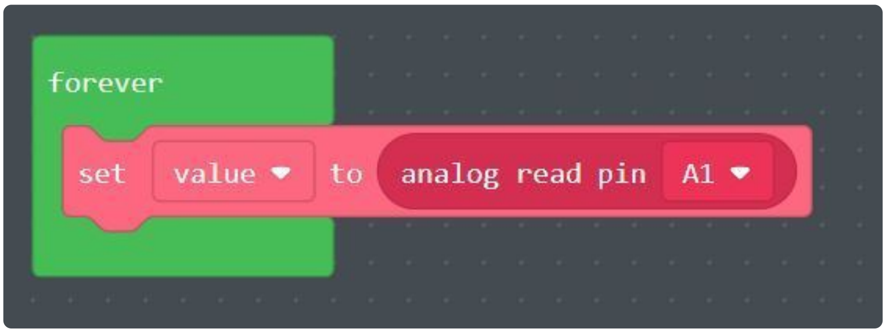
The value of the potentiometer is not crucial. The part suggested is the Adafruit 10 kilo-ohm (KΩ) model (http://adafru.it/562) but other values should give similar results. It is suggested avoiding under 1000 ohms and over a mega-ohm to get the current flow in a good range. If you decide to use a breadboard instead of alligator clips, most Adafruit potentiometers are breadboard friendly (look in their description to make sure).

See the last page (https://adafru.it/CwO) on what types of potentiometers are available out in the wild if you have specific requirements.

MakeCode

Reading a potentiometer value takes one analog pin on a microcontroller. Analog pins read in a voltage and provide a value based on the voltage it reads. The number depends on the accuracy of the analog to digital conversion..

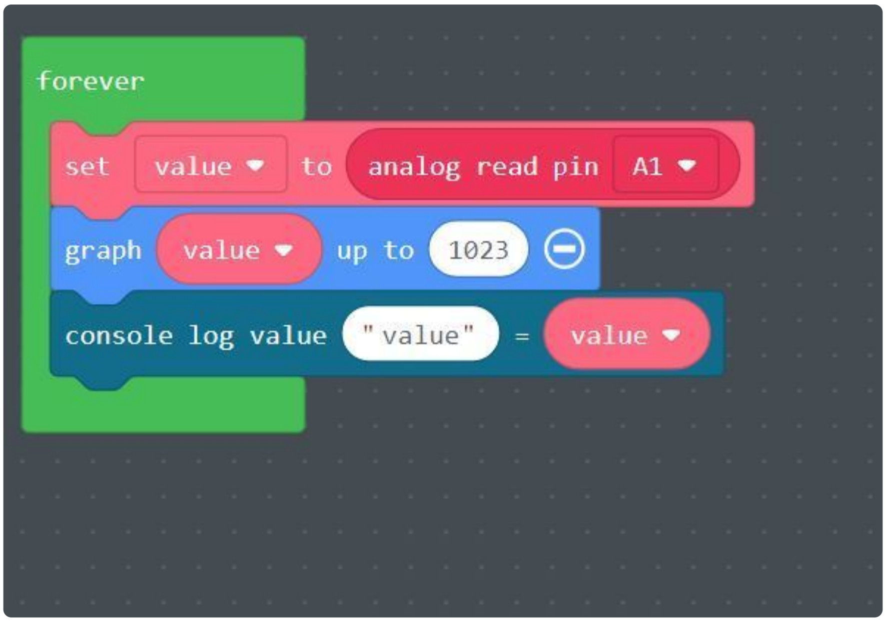
The basic code to read the value in MakeCode is shown below:



```
forever
    set value ▾ to analog read pin A1 ▾
```

This code reads the analog pin **A1** (it can be changed from A1 to A7 for Circuit Playground Express) and the result is put in a variable called value. The number ranges from 0 (corresponding to 0 volts) at full counterclockwise to 512 (1.65 volts) at the center point to 1023 (3.3 volts) at the far clockwise point.

This code would be more useful if we could see the reading. The following code, Circuit Playground-specific, lights the LEDs from 0 to 9 based on the reading from the potentiometer and the `analog read pin`:



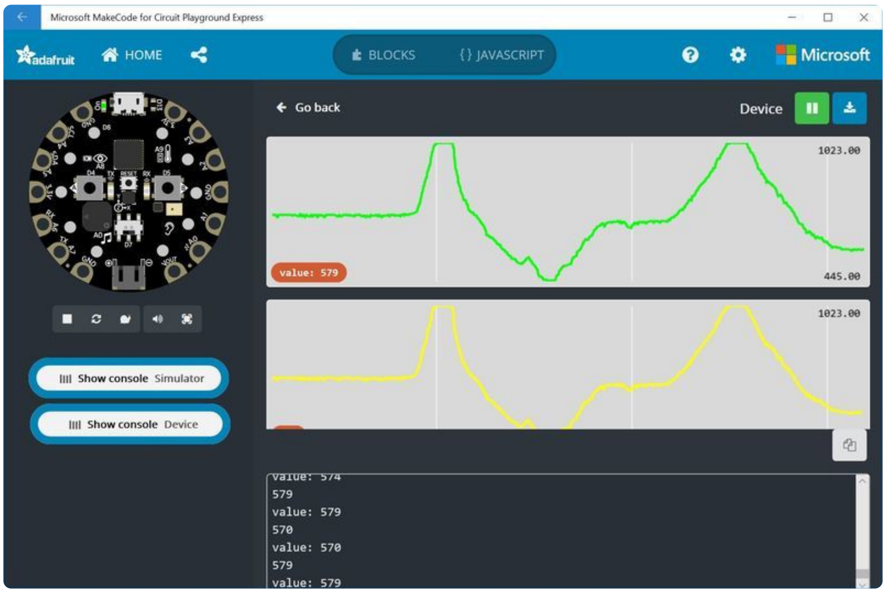
```
forever
    set value to analog read pin A1
    graph value up to 1023
    console log value "value" = value
```

Open this example in MakeCode

https://adafru.it/CwP

This code will light the Circuit Playground Express LEDs proportional to the travel of the potentiometer wiper.

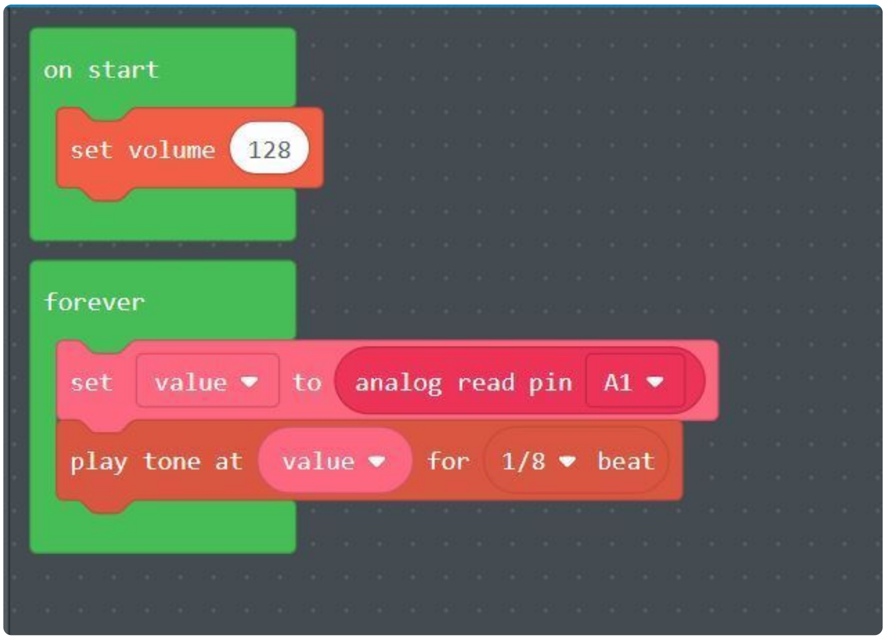
Also if you have console capability (Windows 10 MakeCode App), you can see the value printed out, shown below.



```
value: 574
579
value: 579
570
value: 570
579
value: 579
```

Changing Music

The second example varies a tone on the Circuit Playground Express speaker depending on the value read from the potentiometer:



```
on start
    set volume 128

forever
    set value to analog read pin A1
    play tone at value for 1/8 beat
```

Open this example in MakeCode

https://adafru.it/CwQ

This example makes really annoying noises!

You will find this example somewhat hard on the ears. The potentiometer will set the tone value from 0 to 1023 hertz - the volume will cut off at a value more than zero. And the values will not often be perfect musical note values. You can change 1/8 to other values for interesting effects.

CircuitPython

CircuitPython is a programming language based on Python, one of the fastest growing programming languages in the world. It is specifically designed to simplify experimenting and learning to code on low-cost microcontroller boards. You can learn more about CircuitPython in this guide (https://adafru.it/cpy-welcome).

CircuitPython is easiest to use within the Mu Editor. If you haven't previously used Mu, this guide will get you started (https://adafru.it/ANO). Mu is available for PC, mac, and Linux and it has some very handy features we'll use.

Analog Reads

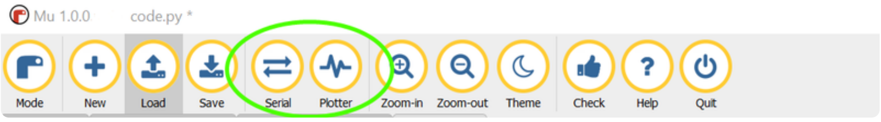
The CircuitPython `analogio` module provides the analog pin functions comparable to MakeCode `analog read block` and Arduino `analogRead` function.

Save the following program to your Circuit Playground Express **CIRCUITPY** drive as **code.py**.

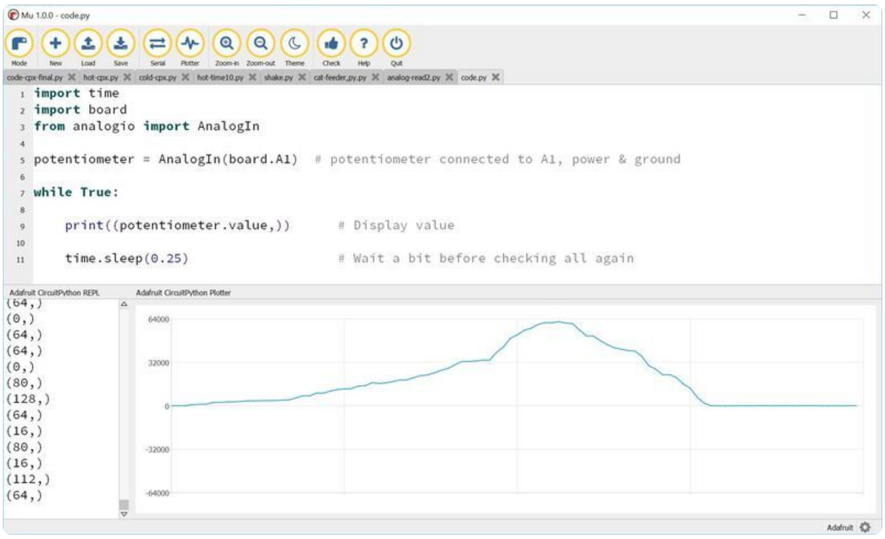
```
# SPDX-FileCopyrightText: 2018 Anne Barela for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time
import board
from analogio import AnalogIn

potentiometer = AnalogIn(board.A1)  # potentiometer connected to A1, power & ground

while True:

    print((potentiometer.value,))      # Display value

    time.sleep(0.25)                   # Wait a bit before checking all again
```

`AnalogIn` creates an object, here called `potentiometer`, on pad A1 on Circuit Playground Express. The `while` loop indefinitely prints out the value of the potentiometer object to the Serial connection then waits a bit.

Opening the **Serial** window in Mu (button circled in picture below), you will see that the range of values returned from `AnalogIn` is much higher in CircuitPython: 0 to 65535 (was 65520 before CircuitPython 8.0.0). Take that into account when scaling or mapping values to a different range for display, etc.



In Mu, you can press the **Plotter** button to see a plot of the values. As you turn the knob of the potentiometer, the values change.



```
import time
import board
from analogio import AnalogIn

potentiometer = AnalogIn(board.A1)  # potentiometer connected to A1, power & ground

while True:

    print((potentiometer.value,))      # Display value

    time.sleep(0.25)                   # Wait a bit before checking all again
```

```
(64,)
(0,)
(64,)
(64,)
(0,)
(80,)
(128,)
(64,)
(16,)
(80,)
(16,)
(112,)
(64,)
```

Display on the Circuit Playground Express NeoPixels

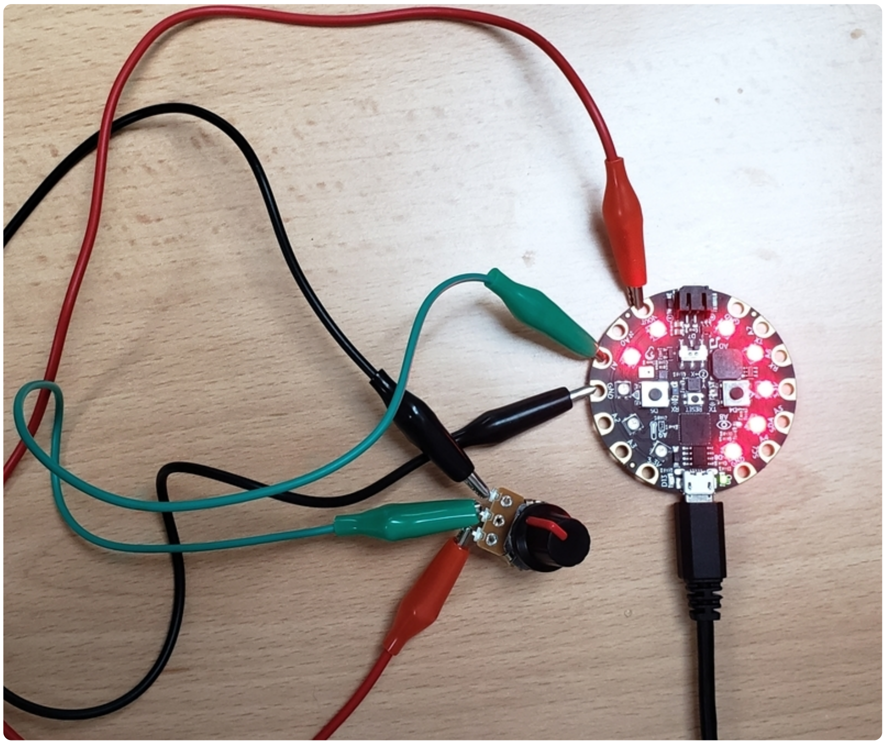
Any string of NeoPixels makes a great indicator for the value of a potentiometer. In the "old days" they would put markings on the case to tell you the amount you had turned the knob. Now you can do this electronically. A stick of NeoPixels (http://adafru.it/1426) or a ring of NeoPixels (http://adafru.it/1463) works great.

If you are using the Circuit Playground Express, the built-in NeoPixels are perfect as an indicator. The code is shown below.

Save the following program to your Circuit Playground Express **CIRCUITPY** drive as **code.py**.

```
# SPDX-FileCopyrightText: 2018 Anne Barela for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time
import board
from analogio import AnalogIn
import neopixel

NUMPIXELS = 10  # Circuit Playground Express has 10 pixels
pixels = neopixel.NeoPixel(board.NEOPIXEL, NUMPIXELS, auto_write=False)  # CPX
NeoPixels
potentiometer = AnalogIn(board.A1)  # potentiometer connected to A1, power & ground

def show_value(val):            # Show value 0-9 on CPX NeoPixels
    for i in range(val):
        pixels[i] = (50, 0, 0)  # Red
    for i in range(val, NUMPIXELS):
        pixels[i] = (0, 0, 0)
    pixels.show()

while True:

    show_value(int(potentiometer.value / 65520 * NUMPIXELS))  # Show on NeoPixels
    print((potentiometer.value,))                             # Print value
    time.sleep(0.25)                   # Wait a bit before checking all again
```

You can change the neopixel.NeoPixel call to reference an external strip or ring of pixels for your board and the number of pixels being addressed. `NUMPIXELS` should be changed to the number you have.

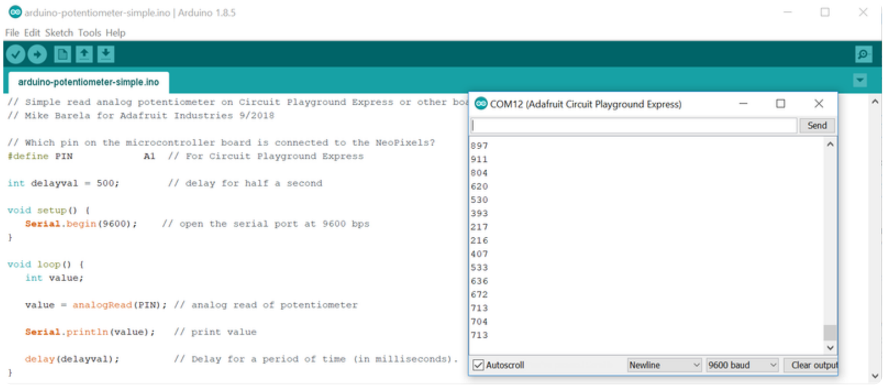


Hopefully this has given you an idea how to read a potentiometer in CircuitPython and use the value in your project.

Arduino

If you are new to Arduino IDE programming, see this guide (https://adafru.it/Clw). If you are familiar with Arduino but not on the Circuit Playground Express, see this guide (https://adafru.it/C3t).

Reading analog pins is built into the Arduino environment, available as the standard function call `analogRead`. The simple sketch below reads the value of a potentiometer attached to the pin defined by `PIN`. For Circuit Playground Express this is `A1`, but it could be any analog pin.



arduino-potentiometer-simple.ino | Arduino 1.8.5
File Edit Sketch Tools Help
arduino-potentiometer-simple.ino
COM12 (Adafruit Circuit Playground Express)
Send
☑ Autoscroll
Newline
9600 baud
Clear output

```
// Simple read analog potentiometer on Circuit Playground Express or other boa
// Mike Barela for Adafruit Industries 9/2018

// Which pin on the microcontroller board is connected to the NeoPixels?
#define PIN            A1  // For Circuit Playground Express

int delayval = 500;        // delay for half a second

void setup() {
   Serial.begin(9600);    // open the serial port at 9600 bps
}

void loop() {
   int value;

   value = analogRead(PIN); // analog read of potentiometer

   Serial.println(value);   // print value

   delay(delayval);         // Delay for a period of time (in milliseconds).
}
```

```
897
911
804
620
530
393
217
216
407
533
636
672
713
704
713
```

Display to Serial

The following is a simple program that reads a potentiometer connected to A1 and displays that value to the Arduino Serial console. The sample output is shown above.

```
// SPDX-FileCopyrightText: 2018 Anne Barela for Adafruit Industries
//
// SPDX-License-Identifier: MIT

// Simple read analog potentiometer on Circuit Playground Express or other board
with pin change
// Anne Barela for Adafruit Industries 9/2018

#define ANALOGPIN      A1  // For Circuit Playground Express

int delayval = 500;        // delay for half a second

void setup() {
   Serial.begin(9600);    // open the serial port at 9600 bps
}

void loop() {
   int value;

   value = analogRead(ANALOGPIN); // analog read of potentiometer

   Serial.println(value);   // print value

   delay(delayval);         // Delay for a period of time (in milliseconds).
}
```

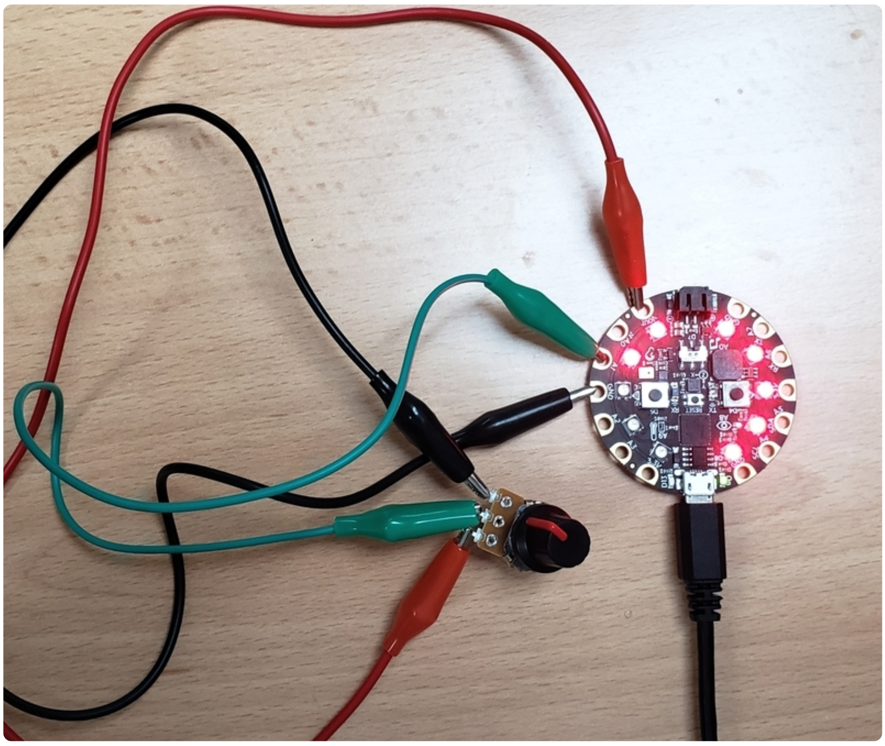
Display on NeoPixels

As in previous examples, using a potentiometer on a microcontroller, either the Circuit Playground Express or another board allows use of NeoPixels. External pixels are fine. The code below uses the onboard pixels on a Circuit Playground Express to display the value.

```
// SPDX-FileCopyrightText: 2018 Anne Barela for Adafruit Industries
//
// SPDX-License-Identifier: MIT

// Read analog potentiometer on Circuit Playground Express or other board with
```

```
changes
// Anne Barela for Adafruit Industries 9/2018 based on
// NeoPixel Ring simple sketch (c) 2013 Shae Erisson
// released under the GPLv3 license to match the rest of the AdaFruit NeoPixel
library

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h>
#endif

// Which pin on the microcontroller board is connected to the NeoPixels?
#define PIN             8  // For Circuit Playground Express

// How many NeoPixels are attached to the board?
#define NUMPIXELS      10

// When we setup the NeoPixel library, we tell it how many pixels, and which pin to
use to send signals.
// Note that for older NeoPixel strips you might need to change the third
parameter--see the strandtest
// example for more information on possible values.
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

int delayval = 500; // delay for half a second

void setup() {
  Serial.begin(9600);
  pixels.begin(); // This initializes the NeoPixel library.
}

void loop() {
   int i;              // loop variable
   int value;          // analog read of potentiometer
   int display_value;  // number of NeoPixels to display out of NUMPIXELS

   // Read PIN value and scale from 0 to NUMPIXELS -1
   value = analogRead(A1);
   Serial.print(value);
   Serial.print(", ");
   display_value = int(value  * NUMPIXELS / 1023);
   Serial.println(display_value);

   // For a set of NeoPixels the first NeoPixel is 0, second is 1, all the way up
to the count of pixels minus one

   for(i=0; i<display_value; i++){

      // pixels.Color takes RGB values, from 0,0,0 up to 255,255,255
      pixels.setPixelColor(i, 0, 050, 0); // Moderately bright green color
   }
   for(i=display_value; i<NUMPIXELS; i++) {
      pixels.setPixelColor(i, 0, 0, 0);    // turn off all pixels after value
displayed
   }

   pixels.show(); // This sends the updated pixel color to the hardware.

   delay(delayval); // Delay for a period of time (in milliseconds).
}
```

Circuit Playground Express NeoPixels are on digital pin 8. The analogRead reads the potentiometer on pin A1.



Types of Potentiometers Available

There are probably at least as many types of potentiometers as there are resistors. Potentiometers for all types of purposes. This page dives in deeper to let you know many of the types you'll find available.

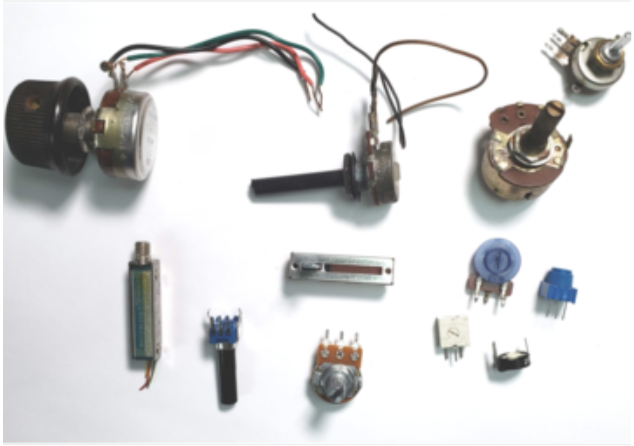
Potentiometer Values

Exactly like normal, fixed resistors, potentiometers are rated for a maximum resistance measured in Ohms. Often it is in thousands of Ohms (1000 = 1K), or millions of Ohms (1,000,000 = 1000K = 1M megohm).

For reading varying analog voltage readings like in this guide, 10K to 100K Ohm potentiometers are ideal. Be aware there are many different values available from resellers.

Physical Size

Not all potentiometers are the same physical size. Most often the size relates to the power capacity of the potentiometer but not always.



If you are breadboarding a circuit, look for potentiometers that are **breadboard friendly** meaning the terminals fit a 0.1" grid of pins. Adafruit sells many breadboard friendly potentiometers (https://adafru.it/CwS).

For other uses, you will want to measure the space you have in your project or project enclosure to see if your planned potentiometers fit. Also consider ergonomics, will the potentiometer be placed to be easy for the user?

Potentiometers made to be mounted on an enclosure are called **Panel Mount Potentiometers**. They come with a nut to hold the potentiometer to the panel. The shaft most often makes accomodation for a knob to move the shaft.

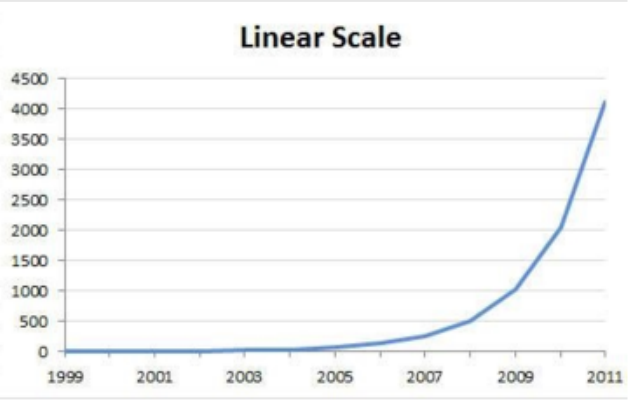
Adafruit sells potentiometers that are both breadboard friendly for prototyping and panel mountable (https://adafru.it/CwT) for making a finished project easily. The best of both worlds for Makers.

Rotary vs. Slider (Twist vs. Straight)

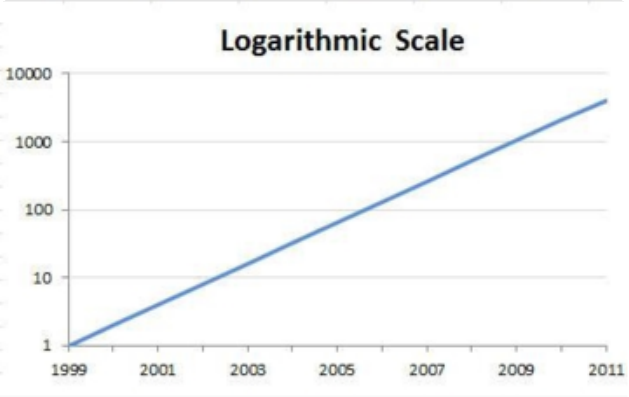
The most common potentiometers have a knob or shaft of some sort and and you twist the knob to change the value.

Sliders, straight running potentiometers, move along a rectangular track to move only up and down. They are often used for audio mixing and lighting control.

Linear Taper vs. Logarithmic Taper



Getting more exotic. Most potentiometers for regular use are linear taper potentiometers. The value of the resistance varies linearly as you adjust it, from 0 to 100% of the adjustment. In the examples in this guide, this gives a nice, even set of numbers and pixels as the potentiometer is adjusted.



In some circuits/projects, a different type of resistance is better. For example, the human ear doesn't perceive sound linearly but rather logarithmically. So some audio applications and other circuits use potentiometers that vary the resistance logarithmically to mechanically vary the resistance to a log scale. They still give 0 to 100% of the value, just not in a "twist 10 degrees = x" amount of resistance through the whole adjustment range.

Adafruit sells potentiometers (https://adafru.it/CwU) with analog and logarithmic tapers - when you are browsing the catalog, be aware of this distinction.

You want to ensure you use the correct type of potentiometer for your application. If you are unsure, go with linear and you can substitute later.

Dual, Stacked and Switched Potentiometers

Some applications, such as audio, benefit from adjusting two potentiometers via one shaft. Those are called **dual potentiometers**. More rarely there could be three or more stacked on one shaft. These come in linear and logarithmic tapers.

Some potentiometers come with an on/off switch mounted on the back of either a single, dual, or stacked potentiometer. This is great for using the same knob to both turn on a circuit and adjust parameters.

Fixed or Not?

You may see potentiometers on circuit boards rather than on a panel for user control. These most often are usually left in one place, maybe adjusted during calibration of a circuit. These are called fixed potentiometers as they keep a fixed value unless you open things up and tweak them. They may be adjustable via finger or screwdriver.

It seems strange to fix a value of an adjustable device but it does make things handy for tweaking things on an irregular basis.

Single or Multi Turn

Most potentiometers adjust in a single motion from one extreme to the other (0 to 100%). There are some potentiometers, usually smaller ones inside a circuit) that can turn more than the standard amount, they can be adjusted (usually via screwdriver) over multiple turns of the wiper shaft. They provide a greater range than single turn potentiometers, providing greater accuracy. Most often they are not adjusted often, maybe during a circuit's calibration.

Materials

Like regular resistors, most potentiometers use a carbon composition to provide the resistance. The potentiometer will be rated for a certain power capacity, the voltage times the amperage it can handle, measured in Watts.

For hobby applications, low wattage potentiometers should be fine although if you are making something for sale or production, checking the wattage is necessary.

You may find potentiometers made of other types of materials: ceramic/metal (cermet), conductive plastic, or wire wound. Wire wound resistors usually are rotary and you might feel the individual wires as you turn the knob. Wire wound is most often used for high power/wattage applications or potentiometers with a small resistance.