
Adafruit CircuitPlayground Library Documentation

Release 1.0

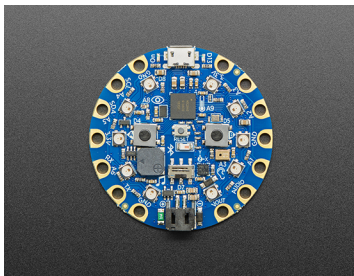
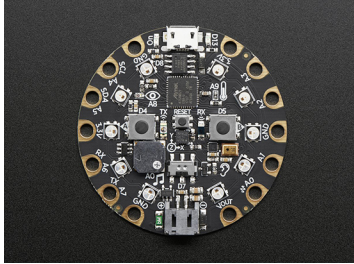
Scott Shawcroft

Mar 21, 2020

Contents

1	Installation	3
2	Usage Example	5
3	Circuit Playground Library Details	7
4	Contributing	9
5	Documentation	11
6	Table of Contents	13
6.1	Simple test	13
6.2	<code>adafruit_circuitplayground.circuit_playground_base</code>	27
6.3	<code>adafruit_circuitplayground.bluefruit</code>	45
6.3.1	Implementation Notes	46
6.4	<code>adafruit_circuitplayground.express</code>	47
7	Indices and tables	49
	Python Module Index	51
	Index	53

This high level library provides objects that represent Circuit Playground Express and Bluefruit hardware.



CHAPTER 1

Installation

For Circuit Playground Express, simply install CircuitPython to use this library - the library itself and all of its dependencies are built into CircuitPython for Circuit Playground Express.

For Circuit Playground Bluefruit, you must install this library and all of its dependencies. Please download [the latest Adafruit CircuitPython library bundle](#). Open the resulting zip file, open the lib folder within, and copy the following folders and files to the lib folder on your CIRCUITPY drive:

- adafruit_bus_device/
- adafruit_circuitplayground/
- adafruit_lis3dh.mpy
- adafruit_thermistor.mpy
- neopixel.mpy

CHAPTER 2

Usage Example

Using this library is super simple. Simply import the `cp` variable from the module and then use it.

```
from adafruit_circuitplayground import cp

while True:
    if cp.button_a:
        print("Temperature:", cp.temperature)
    cp.red_led = cp.button_b
```

To learn more about all the features of this library, check out the [CircuitPython Made Easy on Circuit Playground Express and Bluefruit guide](#) on the Adafruit Learn System.

CHAPTER 3

Circuit Playground Library Details

For a detailed explanation of how the Circuit Playground library functions, see [The Technical Side](#) page of the CircuitPython Made Easy on Circuit Playground Express and Bluefruit guide.

CHAPTER 4

Contributing

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.

CHAPTER 5

Documentation

For information on building library documentation, please check out [this guide](#).

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/circuitplayground_acceleration.py

```
1  """
2  This example uses the accelerometer on the Circuit Playground. It prints the values.
   ↳ Try moving
3  the board to see the values change. If you're using Mu, open the plotter to see the
   ↳ values plotted.
4  """
5  import time
6  from adafruit_circuitplayground import cp
7
8  while True:
9      x, y, z = cp.acceleration
10     print((x, y, z))
11
12     time.sleep(0.1)
```

Listing 2: examples/circuitplayground_pixels_simpletest.py

```
1  """This example lights up the NeoPixels with a rainbow swirl."""
2  import time
3  from adafruit_circuitplayground import cp
4
5
6  def wheel(pos):
7      # Input a value 0 to 255 to get a color value.
8      # The colours are a transition r - g - b - back to r.
9      if (pos < 0) or (pos > 255):
10         return (0, 0, 0)
```

(continues on next page)

(continued from previous page)

```

11     if pos < 85:
12         return (int(pos * 3), int(255 - (pos * 3)), 0)
13     if pos < 170:
14         pos -= 85
15         return (int(255 - pos * 3), 0, int(pos * 3))
16     pos -= 170
17     return (0, int(pos * 3), int(255 - pos * 3))
18
19
20 def rainbow_cycle(wait):
21     for j in range(255):
22         for i in range(cp.pixels.n):
23             idx = int((i * 256 / len(cp.pixels)) + j)
24             cp.pixels[i] = wheel(idx & 255)
25             cp.pixels.show()
26             time.sleep(wait)
27
28
29 cp.pixels.auto_write = False
30 cp.pixels.brightness = 0.3
31 while True:
32     rainbow_cycle(0.001)  # rainbowcycle with 1ms delay per step

```

Listing 3: examples/circuitplayground_shake.py

```

1  """This example prints to the serial console when the Circuit Playground is shaken."""
2  from adafruit_circuitplayground import cp
3
4  while True:
5      if cp.shake():
6          print("Shake detected!")

```

Listing 4: examples/circuitplayground_tapdetect_single_double.py

```

1  """This example shows how you can use single-tap and double-tap together with a delay_
   ↳ between.
2  Single-tap the board twice and then double-tap the board twice to complete the_
   ↳ program."""
3  from adafruit_circuitplayground import cp
4
5  # Set to check for single-taps.
6  cp.detect_taps = 1
7  tap_count = 0
8
9  # We're looking for 2 single-taps before moving on.
10 while tap_count < 2:
11     if cp.tapped:
12         tap_count += 1
13 print("Reached 2 single-taps!")
14
15 # Now switch to checking for double-taps
16 tap_count = 0
17 cp.detect_taps = 2
18
19 # We're looking for 2 double-taps before moving on.
20 while tap_count < 2:

```

(continues on next page)

(continued from previous page)

```

21     if cp.tapped:
22         tap_count += 1
23 print("Reached 2 double-taps!")
24 print("Done.")
25 while True:
26     cp.red_led = True

```

Listing 5: examples/circuitplayground_tapdetect.py

```

1  """This example prints to the serial console when the board is double-tapped."""
2  import time
3  from adafruit_circuitplayground import cp
4
5  # Change to 1 for single-tap detection.
6  cp.detect_taps = 2
7
8  while True:
9      if cp.tapped:
10         print("Tapped!")
11         time.sleep(0.05)

```

Listing 6: examples/circuitplayground_tone.py

```

1  """This example plays a different tone for each button, while the button is pressed."""
2  ↪
3  from adafruit_circuitplayground import cp
4
5  while True:
6      if cp.button_a:
7         cp.start_tone(262)
8      elif cp.button_b:
9         cp.start_tone(294)
10     else:
11         cp.stop_tone()

```

Listing 7: examples/circuitplayground_touched.py

```

1  """This example prints to the serial console when you touch the capacitive touch pads.
2  ↪ """
3  from adafruit_circuitplayground import cp
4
5  while True:
6      if cp.touch_A1:
7         print("Touched pad A1")
8      if cp.touch_A2:
9         print("Touched pad A2")
10     if cp.touch_A3:
11         print("Touched pad A3")
12     if cp.touch_A4:
13         print("Touched pad A4")
14     if cp.touch_A5:
15         print("Touched pad A5")
16     if cp.touch_A6:
17         print("Touched pad A6")
18     if cp.touch_TX:
19         print("Touched pad TX")

```

Listing 8: examples/circuitplayground_acceleration_neopixels.py

```

1  """If the switch is to the right, it will appear that nothing is happening. Move the
   ↳switch to the
2  left to see the NeoPixels light up in colors related to the accelerometer! The
   ↳Circuit Playground
3  has an accelerometer in the center that returns (x, y, z) acceleration values. This
   ↳program uses
4  those values to light up the NeoPixels based on those acceleration values."""
5  from adafruit_circuitplayground import cp
6
7  # Main loop gets x, y and z axis acceleration, prints the values, and turns on
8  # red, green and blue, at levels related to the x, y and z values.
9  while True:
10     if not cp.switch:
11         # If the switch is to the right, it returns False!
12         print("Slide switch off!")
13         cp.pixels.fill((0, 0, 0))
14         continue
15     R = 0
16     G = 0
17     B = 0
18     x, y, z = cp.acceleration
19     print((x, y, z))
20     cp.pixels.fill((R + abs(int(x))), (G + abs(int(y))), (B + abs(int(z))))

```

Listing 9: examples/circuitplayground_button_a.py

```

1  """This example turns on the little red LED when button A is pressed."""
2  from adafruit_circuitplayground import cp
3
4  while True:
5     if cp.button_a:
6         print("Button A pressed!")
7         cp.red_led = True

```

Listing 10: examples/circuitplayground_button_b.py

```

1  """This example turns the little red LED on only while button B is currently being
   ↳pressed."""
2  from adafruit_circuitplayground import cp
3
4  # This code is written to be readable versus being Pylint compliant.
5  # pylint: disable=simplifiable-if-statement
6
7  while True:
8     if cp.button_b:
9         cp.red_led = True
10     else:
11         cp.red_led = False
12
13  # Can also be written as:
14  # cp.red_led = cp.button_b

```


Listing 11: examples/circuitplayground_buttons_1_neopixel.py

```

1  """This example lights up the third NeoPixel while button A is being pressed, and
   ↪lights up the
2  eighth NeoPixel while button B is being pressed."""
3  from adafruit_circuitplayground import cp
4
5  cp.pixels.brightness = 0.3
6  cp.pixels.fill((0, 0, 0)) # Turn off the NeoPixels if they're on!
7
8  while True:
9      if cp.button_a:
10         cp.pixels[2] = (0, 255, 0)
11     else:
12         cp.pixels[2] = (0, 0, 0)
13
14     if cp.button_b:
15         cp.pixels[7] = (0, 0, 255)
16     else:
17         cp.pixels[7] = (0, 0, 0)

```

Listing 12: examples/circuitplayground_buttons_neopixels.py

```

1  """This example lights up half the NeoPixels red while button A is being pressed, and
   ↪half the
2  NeoPixels green while button B is being pressed."""
3  from adafruit_circuitplayground import cp
4
5  cp.pixels.brightness = 0.3
6  cp.pixels.fill((0, 0, 0)) # Turn off the NeoPixels if they're on!
7
8  while True:
9      if cp.button_a:
10         cp.pixels[0:5] = [(255, 0, 0)] * 5
11     else:
12         cp.pixels[0:5] = [(0, 0, 0)] * 5
13
14     if cp.button_b:
15         cp.pixels[5:10] = [(0, 255, 0)] * 5
16     else:
17         cp.pixels[5:10] = [(0, 0, 0)] * 5

```

Listing 13: examples/circuitplayground_ir_receive.py

```

1  """THIS EXAMPLE REQUIRES A SEPARATE LIBRARY BE LOADED ONTO YOUR CIRCUITPY DRIVE.
2  This example requires the adafruit_irremote.mpy library.
3
4  THIS EXAMPLE WORKS WITH CIRCUIT PLAYGROUND EXPRESS ONLY.
5
6  This example uses the IR receiver found near the center of the board. Works with
   ↪another Circuit
7  Playground Express running the circuitplayground_ir_transmit.py example. The
   ↪NeoPixels will light
8  up when the buttons on the TRANSMITTING Circuit Playground Express are pressed!"""
9  import pulseio
10 import board

```

(continues on next page)

(continued from previous page)

```

11 import adafruit_irremote
12 from adafruit_circuitplayground import cp
13
14 # Create a 'pulseio' input, to listen to infrared signals on the IR receiver
15 try:
16     pulsein = pulseio.PulseIn(board.IR_RX, maxlen=120, idle_state=True)
17 except AttributeError:
18     raise NotImplementedError(
19         "This example does not work with Circuit Playground Bluefruti!"
20     )
21 # Create a decoder that will take pulses and turn them into numbers
22 decoder = adafruit_irremote.GenericDecode()
23
24 while True:
25     cp.red_led = True
26     pulses = decoder.read_pulses(pulsein)
27     try:
28         # Attempt to convert received pulses into numbers
29         received_code = decoder.decode_bits(pulses)
30     except adafruit_irremote.IRNECRepeatException:
31         # We got an unusual short code, probably a 'repeat' signal
32         continue
33     except adafruit_irremote.IRDecodeException:
34         # Something got distorted
35         continue
36
37     print("Infrared code received: ", received_code)
38     if received_code == [66, 84, 78, 65]:
39         print("Button A signal")
40         cp.pixels.fill((100, 0, 155))
41     if received_code == [66, 84, 78, 64]:
42         print("Button B Signal")
43         cp.pixels.fill((210, 45, 0))

```

Listing 14: examples/circuitplayground_ir_transmit.py

```

1 """THIS EXAMPLE REQUIRES A SEPARATE LIBRARY BE LOADED ONTO YOUR CIRCUITPY DRIVE.
2 This example requires the adafruit_irremote.mpy library.
3
4 THIS EXAMPLE WORKS WITH CIRCUIT PLAYGROUND EXPRESS ONLY.
5
6 This example uses the IR transmitter found near the center of the board. Works with_
7 ↪another Circuit
8 Playground Express running the circuitplayground_ir_receive.py example. Press the_
9 ↪buttons to light
10 up the NeoPixels on the RECEIVING Circuit Playground Express!"""
11 import time
12 import pulseio
13 import board
14 import adafruit_irremote
15 from adafruit_circuitplayground import cp
16
17 # Create a 'pulseio' output, to send infrared signals from the IR transmitter
18 try:
19     pwm = pulseio.PWMOut(board.IR_TX, frequency=38000, duty_cycle=2 ** 15)
20 except AttributeError:

```

(continues on next page)

(continued from previous page)

```

19     raise NotImplementedError(
20         "This example does not work with Circuit Playground Bluefruit!"
21     )
22 pulseout = pulseio.PulseOut(pwm) # pylint: disable=no-member
23 # Create an encoder that will take numbers and turn them into NEC IR pulses
24 encoder = adafruit_irremote.GenericTransmit(
25     header=[9500, 4500], one=[550, 550], zero=[550, 1700], trail=0
26 )
27
28 while True:
29     if cp.button_a:
30         print("Button A pressed! \n")
31         cp.red_led = True
32         encoder.transmit(pulseout, [66, 84, 78, 65])
33         cp.red_led = False
34         # wait so the receiver can get the full message
35         time.sleep(0.2)
36     if cp.button_b:
37         print("Button B pressed! \n")
38         cp.red_led = True
39         encoder.transmit(pulseout, [66, 84, 78, 64])
40         cp.red_led = False
41         time.sleep(0.2)

```

Listing 15: examples/circuitplayground_light_neopixels.py

```

1  """
2  This example uses the light sensor on the Circuit Playground, located next to the_
3  ↪picture of the
4  eye on the board. Once you have the library loaded, try shining a flashlight on your_
5  ↪Circuit
6  Playground to watch the number of NeoPixels lit up increase, or try covering up the_
7  ↪light sensor
8  to watch the number decrease.
9  """
10
11 import time
12 from adafruit_circuitplayground import cp
13
14 cp.pixels.auto_write = False
15 cp.pixels.brightness = 0.3
16
17 def scale_range(value):
18     """Scale a value from 0-320 (light range) to 0-9 (NeoPixel range).
19     Allows remapping light value to pixel position."""
20     return round(value / 320 * 9)
21
22 while True:
23     peak = scale_range(cp.light)
24     print(cp.light)
25     print(int(peak))
26
27     for i in range(10):
28         if i <= peak:

```

(continues on next page)

(continued from previous page)

```

28         cp.pixels[i] = (0, 255, 255)
29     else:
30         cp.pixels[i] = (0, 0, 0)
31     cp.pixels.show()
32     time.sleep(0.05)

```

Listing 16: examples/circuitplayground_light.py

```

1  """This example uses the light sensor on your Circuit Playground, located next to the
   ↳ picture of
2  the eye. Try shining a flashlight on your Circuit Playground, or covering the light
   ↳ sensor with
3  your finger to see the values increase and decrease."""
4  import time
5  from adafruit_circuitplayground import cp
6
7  while True:
8      print("Light:", cp.light)
9      time.sleep(0.2)

```

Listing 17: examples/circuitplayground_neopixel_0_1.py

```

1  """This example lights up the first and second NeoPixel, red and blue respectively."""
2  from adafruit_circuitplayground import cp
3
4  cp.pixels.brightness = 0.3
5
6  while True:
7      cp.pixels[0] = (255, 0, 0)
8      cp.pixels[1] = (0, 0, 255)

```

Listing 18: examples/circuitplayground_light_plotter.py

```

1  """If you're using Mu, this example will plot the light levels from the light sensor
   ↳ (located next
2  to the eye) on your Circuit Playground. Try shining a flashlight on your Circuit
   ↳ Playground, or
3  covering the light sensor to see the plot increase and decrease."""
4  import time
5  from adafruit_circuitplayground import cp
6
7  while True:
8      print("Light:", cp.light)
9      print((cp.light,))
10     time.sleep(0.1)

```

Listing 19: examples/circuitplayground_play_file_buttons.py

```

1  """THIS EXAMPLE REQUIRES A WAV FILE FROM THE examples FOLDER IN THE
2  Adafruit_CircuitPython_CircuitPlayground REPO found at:
3  https://github.com/adafruit/Adafruit_CircuitPython_CircuitPlayground/tree/master/
   ↳ examples
4
5  Copy the "dip.wav" and "rise.wav" files to your CIRCUITPY drive.
6

```

(continues on next page)

(continued from previous page)

```

7  Once the files are copied, this example plays a different wav file for each button_
   ↪pressed!"""
8  from adafruit_circuitplayground import cp
9
10 while True:
11     if cp.button_a:
12         cp.play_file("dip.wav")
13     if cp.button_b:
14         cp.play_file("rise.wav")

```

Listing 20: examples/circuitplayground_play_file.py

```

1  """THIS EXAMPLE REQUIRES A WAV FILE FROM THE examples FOLDER IN THE
2  AdafruitPython_CircuitPlayground REPO found at:
3  https://github.com/adafruit/Adafruit_CircuitPython_CircuitPlayground/tree/master/
   ↪examples
4
5  Copy the "dip.wav" file to your CIRCUITPY drive.
6
7  Once the file is copied, this example plays a wav file!"""
8  from adafruit_circuitplayground import cp
9
10 cp.play_file("dip.wav")

```

Listing 21: examples/circuitplayground_play_tone_buttons.py

```

1  """This example plays a different tone for a duration of 1 second for each button_
   ↪pressed."""
2  from adafruit_circuitplayground import cp
3
4  while True:
5      if cp.button_a:
6          cp.play_tone(262, 1)
7      if cp.button_b:
8          cp.play_tone(294, 1)

```

Listing 22: examples/circuitplayground_play_tone.py

```

1  """This example plays two tones for 1 second each. Note that the tones are not in a_
   ↪loop - this is
2  to prevent them from playing indefinitely!"""
3  from adafruit_circuitplayground import cp
4
5  cp.play_tone(262, 1)
6  cp.play_tone(294, 1)

```

Listing 23: examples/circuitplayground_red_led_blinky.py

```

1  """This is the "Hello, world!" of CircuitPython: Blinky! This example blinks the_
   ↪little red LED on
2  and off!"""
3  import time
4  from adafruit_circuitplayground import cp
5
6  while True:

```

(continues on next page)

(continued from previous page)

```

7     cp.red_led = True
8     time.sleep(0.5)
9     cp.red_led = False
10    time.sleep(0.5)

```

Listing 24: examples/circuitplayground_red_led_blnky_short.py

```

1  """This is the "Hello, world!" of CircuitPython: Blinky! This example blinks the
   ↳ little red LED on
2  and off! It's a shorter version of the other Blinky example."""
3  import time
4  from adafruit_circuitplayground import cp
5
6  while True:
7      cp.red_led = not cp.red_led
8      time.sleep(0.5)

```

Listing 25: examples/circuitplayground_red_led.py

```

1  """This example turns on the little red LED."""
2  from adafruit_circuitplayground import cp
3
4  while True:
5      cp.red_led = True

```

Listing 26: examples/circuitplayground_slide_switch_red_led.py

```

1  """This example uses the slide switch to control the little red LED."""
2  from adafruit_circuitplayground import cp
3
4  # This code is written to be readable versus being Pylint compliant.
5  # pylint: disable=simplifiable-if-statement
6
7  while True:
8      if cp.switch:
9          cp.red_led = True
10     else:
11         cp.red_led = False

```

Listing 27: examples/circuitplayground_slide_switch_red_led_short.py

```

1  """This example uses the slide switch to control the little red LED. When the switch
   ↳ is to the
2  right it returns False, and when it's to the left, it returns True."""
3  from adafruit_circuitplayground import cp
4
5  while True:
6      cp.red_led = cp.switch

```

Listing 28: examples/circuitplayground_slide_switch.py

```

1  """This example prints the status of the slide switch. Try moving the switch back and
   ↳ forth to see
2  what's printed to the serial console!"""
3  import time

```

(continues on next page)

(continued from previous page)

```

4 from adafruit_circuitplayground import cp
5
6 while True:
7     print("Slide switch:", cp.switch)
8     time.sleep(0.1)

```

Listing 29: examples/circuitplayground_sound_meter.py

```

1 """This example uses the sound sensor, located next to the picture of the ear on your_
   ↳board, to
2 light up the NeoPixels as a sound meter. Try talking to your Circuit Playground or_
   ↳clapping, etc,
3 to see the NeoPixels light up!"""
4 import array
5 import math
6 import board
7 import audiobusio
8 from adafruit_circuitplayground import cp
9
10
11 def constrain(value, floor, ceiling):
12     return max(floor, min(value, ceiling))
13
14
15 def log_scale(input_value, input_min, input_max, output_min, output_max):
16     normalized_input_value = (input_value - input_min) / (input_max - input_min)
17     return output_min + math.pow(normalized_input_value, 0.630957) * (
18         output_max - output_min
19     )
20
21
22 def normalized_rms(values):
23     minbuf = int(sum(values) / len(values))
24     return math.sqrt(
25         sum(float(sample - minbuf) * (sample - minbuf) for sample in values)
26         / len(values)
27     )
28
29
30 mic = audiobusio.PDMIn(
31     board.MICROPHONE_CLOCK, board.MICROPHONE_DATA, sample_rate=16000, bit_depth=16
32 )
33
34 samples = array.array("H", [0] * 160)
35 mic.record(samples, len(samples))
36 input_floor = normalized_rms(samples) + 10
37
38 # Lower number means more sensitive - more LEDs will light up with less sound.
39 sensitivity = 500
40 input_ceiling = input_floor + sensitivity
41
42 peak = 0
43 while True:
44     mic.record(samples, len(samples))
45     magnitude = normalized_rms(samples)
46     print((magnitude,))

```

(continues on next page)

(continued from previous page)

```

47
48     c = log_scale(
49         constrain(magnitude, input_floor, input_ceiling),
50         input_floor,
51         input_ceiling,
52         0,
53         10,
54     )
55
56     cp.pixels.fill((0, 0, 0))
57     for i in range(10):
58         if i < c:
59             cp.pixels[i] = (i * (255 // 10), 50, 0)
60         if c >= peak:
61             peak = min(c, 10 - 1)
62         elif peak > 0:
63             peak = peak - 1
64         if peak > 0:
65             cp.pixels[int(peak)] = (80, 0, 255)
66     cp.pixels.show()

```

Listing 30: examples/circuitplayground_tap_red_led.py

```

1  """This example turns on the little red LED and prints to the serial console when you
   ↳double-tap
2  the Circuit Playground!"""
3  import time
4  from adafruit_circuitplayground import cp
5
6  # Change to 1 for detecting a single-tap!
7  cp.detect_taps = 2
8
9  while True:
10     if cp.tapped:
11         print("Tapped!")
12         cp.red_led = True
13         time.sleep(0.1)
14     else:
15         cp.red_led = False

```

Listing 31: examples/circuitplayground_temperature_neopixels.py

```

1  """
2  This example use the temperature sensor on the Circuit Playground, located next to
   ↳the picture of
3  the thermometer on the board. Try warming up the board to watch the number of
   ↳NeoPixels lit up
4  increase, or cooling it down to see the number decrease. You can set the min and max
   ↳temperatures
5  to make it more or less sensitive to temperature changes.
6  """
7  import time
8  from adafruit_circuitplayground import cp
9
10 cp.pixels.auto_write = False
11 cp.pixels.brightness = 0.3

```

(continues on next page)

(continued from previous page)

```

12
13 # Set these based on your ambient temperature in Celsius for best results!
14 minimum_temp = 24
15 maximum_temp = 30
16
17
18 def scale_range(value):
19     """Scale a value from the range of minimum_temp to maximum_temp (temperature_
    ↳range) to 0-10
20     (the number of NeoPixels). Allows remapping temperature value to pixel position."""
    ↳"""
21     return int((value - minimum_temp) / (maximum_temp - minimum_temp) * 10)
22
23
24 while True:
25     peak = scale_range(cp.temperature)
26     print(cp.temperature)
27     print(int(peak))
28
29     for i in range(10):
30         if i <= peak:
31             cp.pixels[i] = (0, 255, 255)
32         else:
33             cp.pixels[i] = (0, 0, 0)
34     cp.pixels.show()
35     time.sleep(0.05)

```

Listing 32: examples/circuitplayground_temperature_plotter.py

```

1 """If you're using Mu, this example will plot the temperature in C and F on the_
    ↳plotter! Click
2 "Plotter" to open it, and place your finger over the sensor to see the numbers change.
    ↳ The
3 sensor is located next to the picture of the thermometer on the CPX."""
4 import time
5 from adafruit_circuitplayground import cp
6
7 while True:
8     print("Temperature C:", cp.temperature)
9     print("Temperature F:", cp.temperature * 1.8 + 32)
10    print((cp.temperature, cp.temperature * 1.8 + 32))
11    time.sleep(0.1)

```

Listing 33: examples/circuitplayground_temperature.py

```

1 """This example uses the temperature sensor on the Circuit Playground, located next_
    ↳to the image of
2 a thermometer on the board. It prints the temperature in both C and F to the serial_
    ↳console. Try
3 putting your finger over the sensor to see the numbers change!"""
4 import time
5 from adafruit_circuitplayground import cp
6
7 while True:
8     print("Temperature C:", cp.temperature)
9     print("Temperature F:", cp.temperature * 1.8 + 32)

```

(continues on next page)

(continued from previous page)

```
time.sleep(1)
```

Listing 34: examples/circuitplayground_touch_pixel_fill_rainbow.py

```
1 """This example uses the capacitive touch pads on the Circuit Playground. They are
   ↳ located around
2 the outer edge of the board and are labeled A1-A6 and TX. (A0 is not a touch pad.)
   ↳ This example
3 lights up all the NeoPixels a different color of the rainbow for each pad touched!"""
4 import time
5 from adafruit_circuitplayground import cp
6
7 cp.pixels.brightness = 0.3
8
9 while True:
10     if cp.touch_A1:
11         print("Touched A1!")
12         cp.pixels.fill((255, 0, 0))
13     if cp.touch_A2:
14         print("Touched A2!")
15         cp.pixels.fill((210, 45, 0))
16     if cp.touch_A3:
17         print("Touched A3!")
18         cp.pixels.fill((155, 100, 0))
19     if cp.touch_A4:
20         print("Touched A4!")
21         cp.pixels.fill((0, 255, 0))
22     if cp.touch_A5:
23         print("Touched A5!")
24         cp.pixels.fill((0, 135, 125))
25     if cp.touch_A6:
26         print("Touched A6!")
27         cp.pixels.fill((0, 0, 255))
28     if cp.touch_TX:
29         print("Touched TX!")
30         cp.pixels.fill((100, 0, 155))
31     time.sleep(0.1)
```

Listing 35: examples/circuitplayground_touch_pixel_rainbow.py

```
1 """This example uses the capacitive touch pads on the Circuit Playground. They are
   ↳ located around
2 the outer edge of the board and are labeled A1-A6 and TX. (A0 is not a touch pad.)
   ↳ This example
3 lights up the nearest NeoPixel to that pad a different color of the rainbow!"""
4 import time
5 from adafruit_circuitplayground import cp
6
7 cp.pixels.brightness = 0.3
8
9 while True:
10     if cp.touch_A1:
11         print("Touched A1!")
12         cp.pixels[6] = (255, 0, 0)
13     if cp.touch_A2:
14         print("Touched A2!")
```

(continues on next page)

(continued from previous page)

```

15     cp.pixels[8] = (210, 45, 0)
16     if cp.touch_A3:
17         print("Touched A3!")
18         cp.pixels[9] = (155, 100, 0)
19     if cp.touch_A4:
20         print("Touched A4!")
21         cp.pixels[0] = (0, 255, 0)
22     if cp.touch_A5:
23         print("Touched A5!")
24         cp.pixels[1] = (0, 135, 125)
25     if cp.touch_A6:
26         print("Touched A6!")
27         cp.pixels[3] = (0, 0, 255)
28     if cp.touch_TX:
29         print("Touched TX!")
30         cp.pixels[4] = (100, 0, 155)
31     time.sleep(0.1)

```

6.2 adafruit_circuitplayground.circuit_playground_base

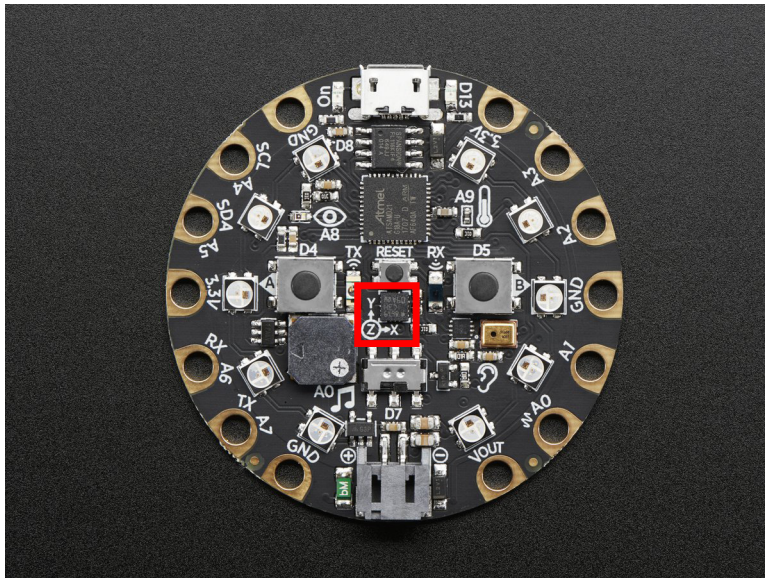
CircuitPython base class for Circuit Playground.

- [Circuit Playground Express](#)
- [Circuit Playground Bluefruit](#).
- Author(s): Kattni Rembor, Scott Shawcroft

class adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase
Circuit Playground base class.

acceleration

Obtain data from the x, y and z axes.



This example prints the values. Try moving the board to see how the printed values change.

To use with the Circuit Playground Express or Bluefruit:

```

from adafruit_circuitplayground import cp

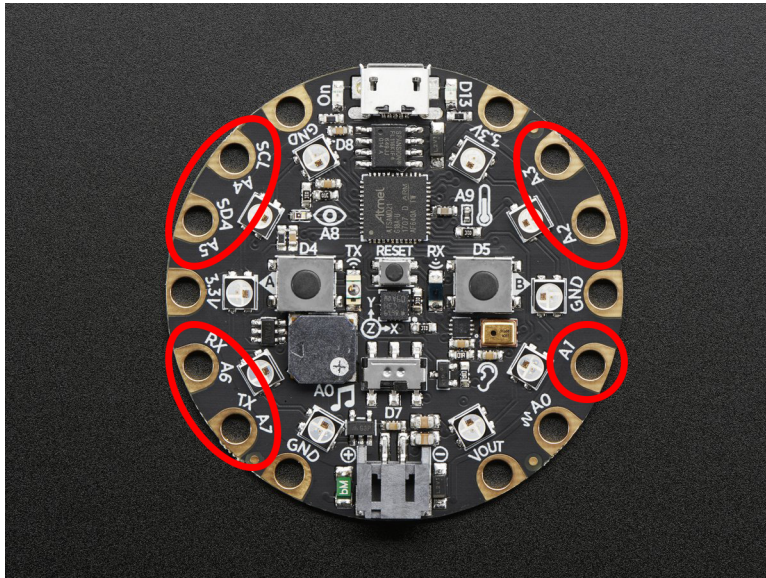
while True:
    x, y, z = cp.acceleration
    print(x, y, z)

```

adjust_touch_threshold(*adjustment*)

Adjust the threshold needed to activate the capacitive touch pads. Higher numbers make the touch pads less sensitive.

Parameters **adjustment** (*int*) – The desired threshold increase



To use with the Circuit Playground Express or Bluefruit:

```

from adafruit_circuitplayground import cp

cp.adjust_touch_threshold(200)

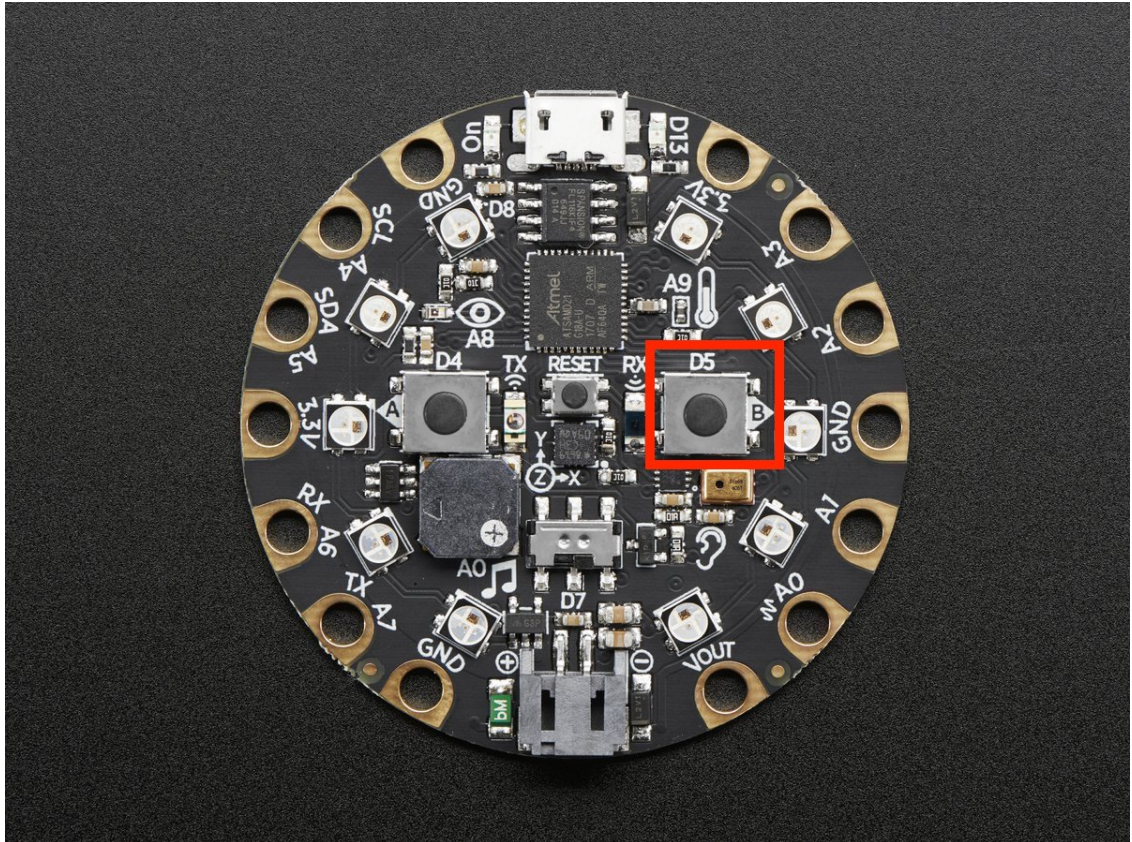
while True:
    if cp.touch_A1:
        print('Touched pad A1')

```

button_a

True when Button A is pressed. False if not.





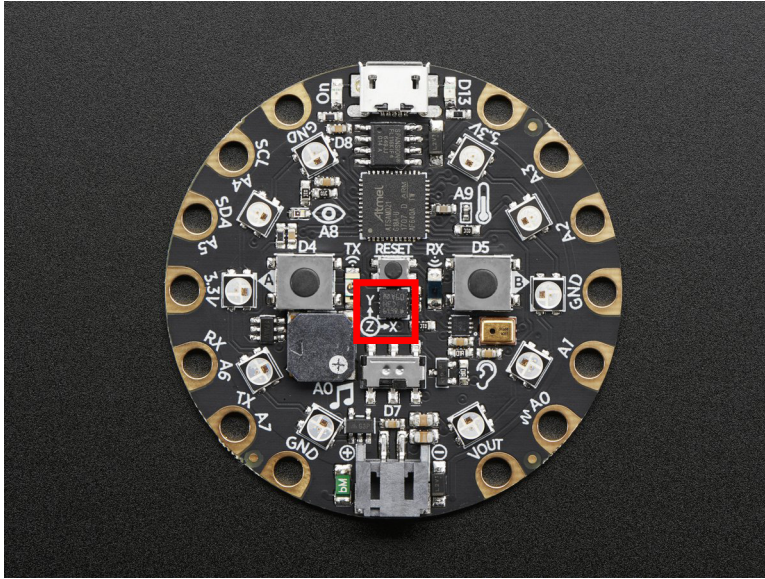
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.button_b:
        print("Button B pressed!")
```

detect_taps

Configure what type of tap is detected by `cp.tapped`. Use 1 for single-tap detection and 2 for double-tap detection. This does nothing without `cp.tapped`.



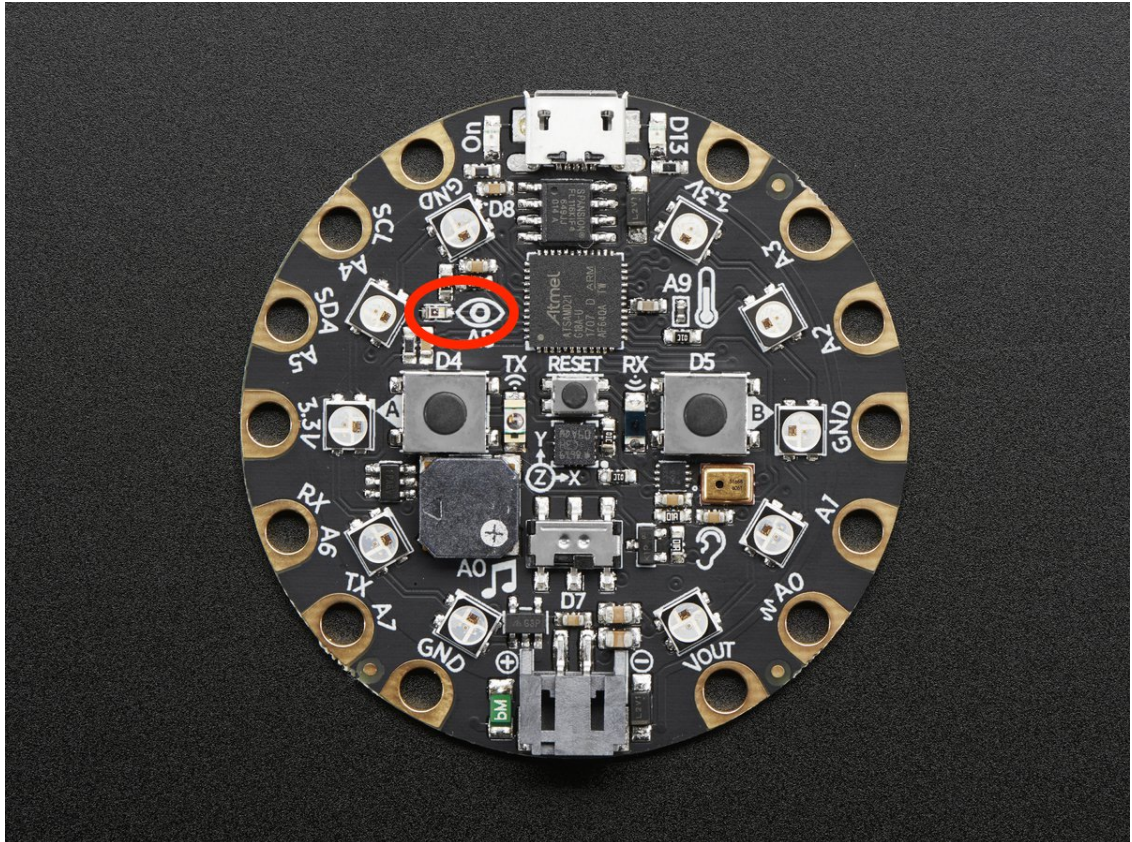
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

cp.detect_taps = 1
while True:
    if cp.tapped:
        print("Single tap detected!")
```

light

The light level.



Try covering the sensor next to the eye to see it change.

To use with the Circuit Playground Express or Bluefruit:

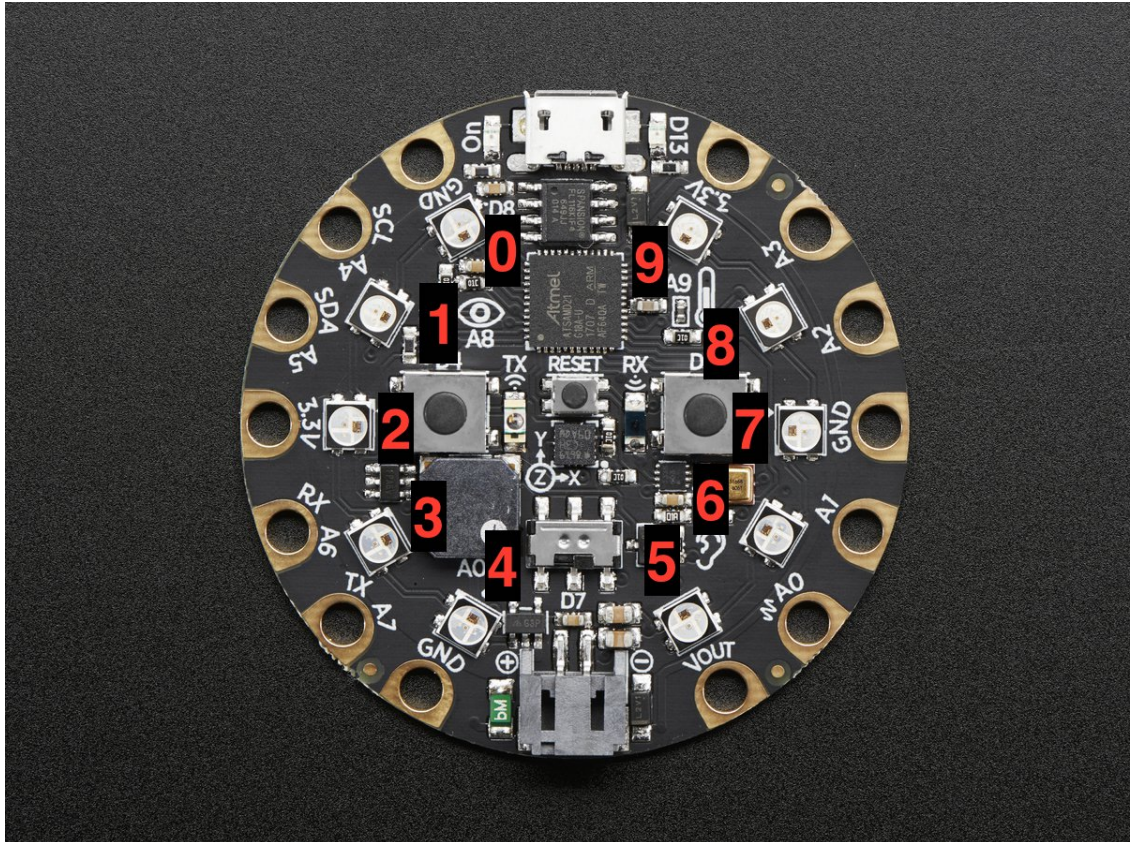
```
from adafruit_circuitplayground import cp
import time

while True:
    print("Light:", cp.light)
    time.sleep(1)
```

pixels

Sequence-like object representing the ten NeoPixels around the outside of the Circuit Playground. Each pixel is at a certain index in the sequence as labeled below. Colors can be RGB hex like 0x110000 for red where each two digits are a color (0xRRGGBB) or a tuple like (17, 0, 0) where (R, G, B). Set the global brightness using any number from 0 to 1 to represent a percentage, i.e. 0.3 sets global brightness to 30%.

See `neopixel.NeoPixel` for more info.



Here is an example that sets the first pixel green and the ninth red.

To use with the Circuit Playground Express or Bluefruit:

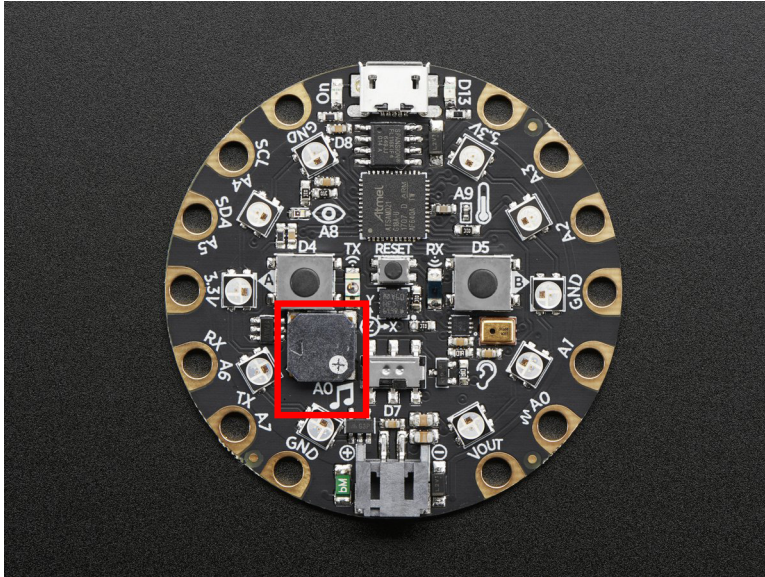
```
from adafruit_circuitplayground import cp

cp.pixels.brightness = 0.3
cp.pixels[0] = 0x00FF00
cp.pixels[9] = (255, 0, 0)
```

play_file(*file_name*)

Play a .wav file using the onboard speaker.

Parameters *file_name* – The name of your .wav file in quotation marks including .wav



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

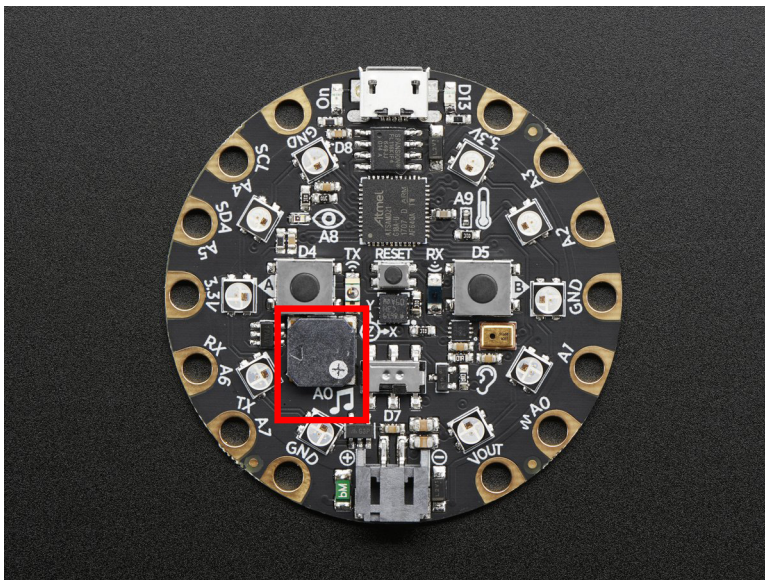
while True:
    if cp.button_a:
        cp.play_file("laugh.wav")
    elif cp.button_b:
        cp.play_file("rimshot.wav")
```

play_tone (*frequency*, *duration*)

Produce a tone using the speaker. Try changing frequency to change the pitch of the tone.

Parameters

- **frequency** (*int*) – The frequency of the tone in Hz
- **duration** (*float*) – The duration of the tone in seconds



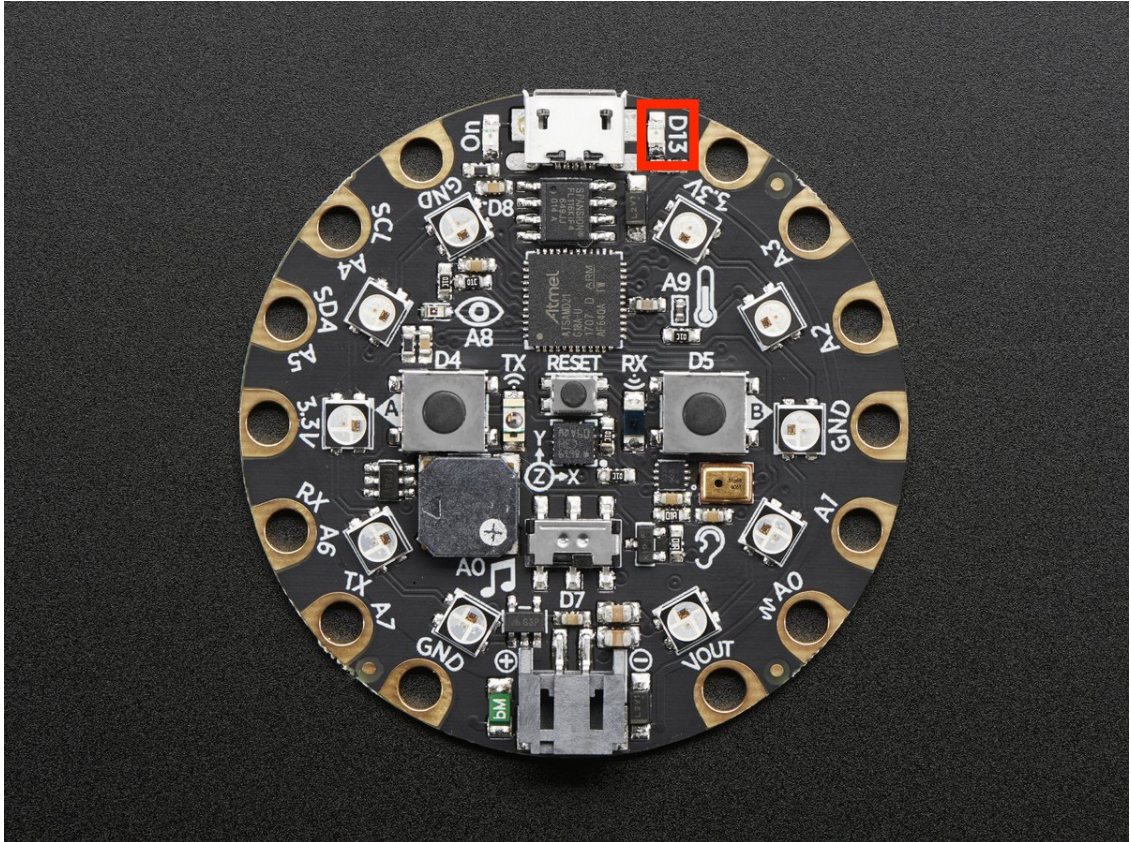
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

cp.play_tone(440, 1)
```

red_led

The red led next to the USB plug marked D13.



To use with the Circuit Playground Express or Bluefruit:

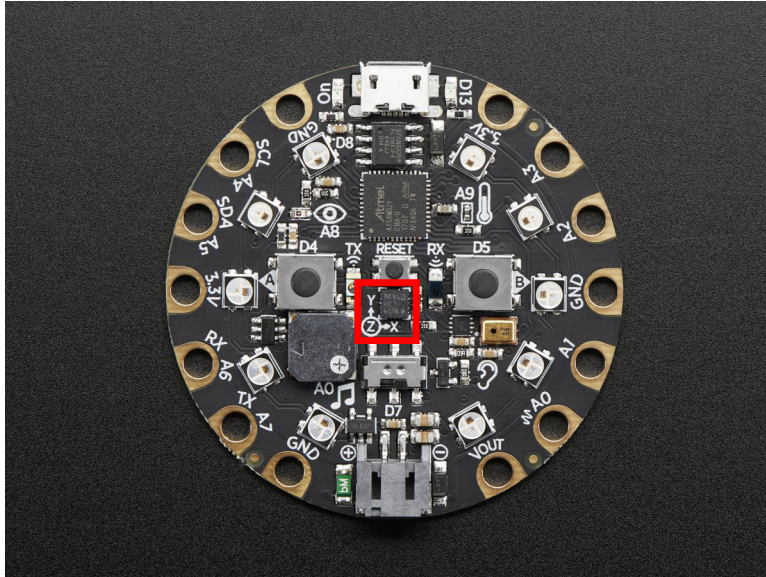
```
from adafruit_circuitplayground import cp
import time

while True:
    cp.red_led = True
    time.sleep(0.5)
    cp.red_led = False
    time.sleep(0.5)
```

shake (shake_threshold=30)

Detect when device is shaken.

Parameters `shake_threshold` (*int*) – The threshold shake must exceed to return true (Default: 30)



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.shake():
        print("Shake detected!")
```

Decreasing `shake_threshold` increases shake sensitivity, i.e. the code will return a shake detected more easily with a lower `shake_threshold`. Increasing it causes the opposite. `shake_threshold` requires a minimum value of 10 - 10 is the value when the board is not moving, therefore anything less than 10 will erroneously report a constant shake detected.

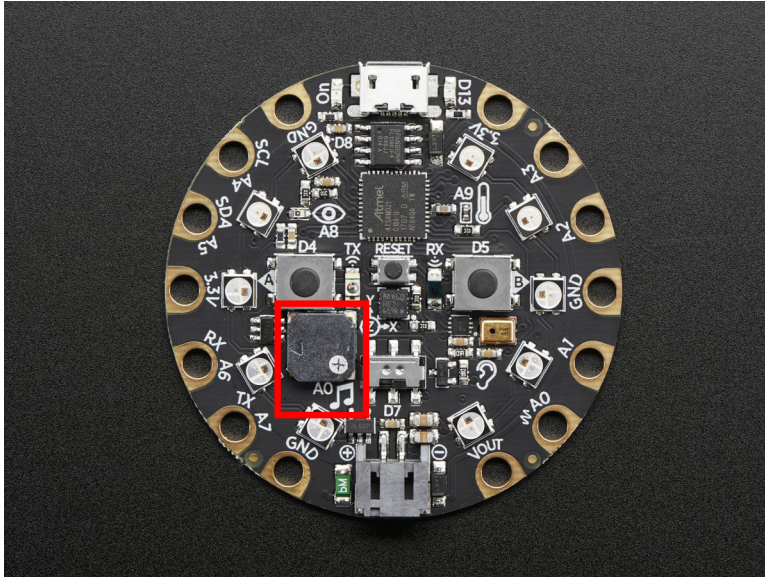
```
from adafruit_circuitplayground import cp

while True:
    if cp.shake(shake_threshold=20):
        print("Shake detected more easily than before!")
```

start_tone (*frequency*)

Produce a tone using the speaker. Try changing frequency to change the pitch of the tone.

Parameters **frequency** (*int*) – The frequency of the tone in Hz



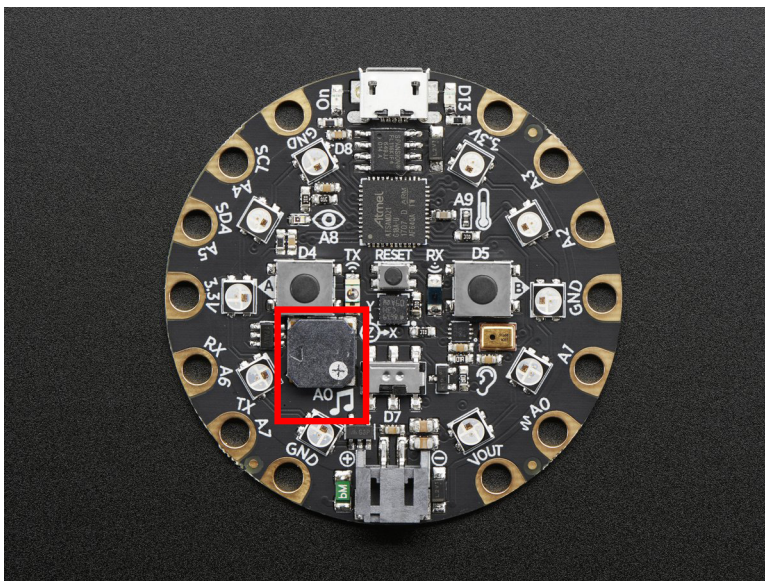
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.button_a:
        cp.start_tone(262)
    elif cp.button_b:
        cp.start_tone(294)
    else:
        cp.stop_tone()
```

stop_tone()

Use with start_tone to stop the tone produced.



To use with the Circuit Playground Express or Bluefruit:

```

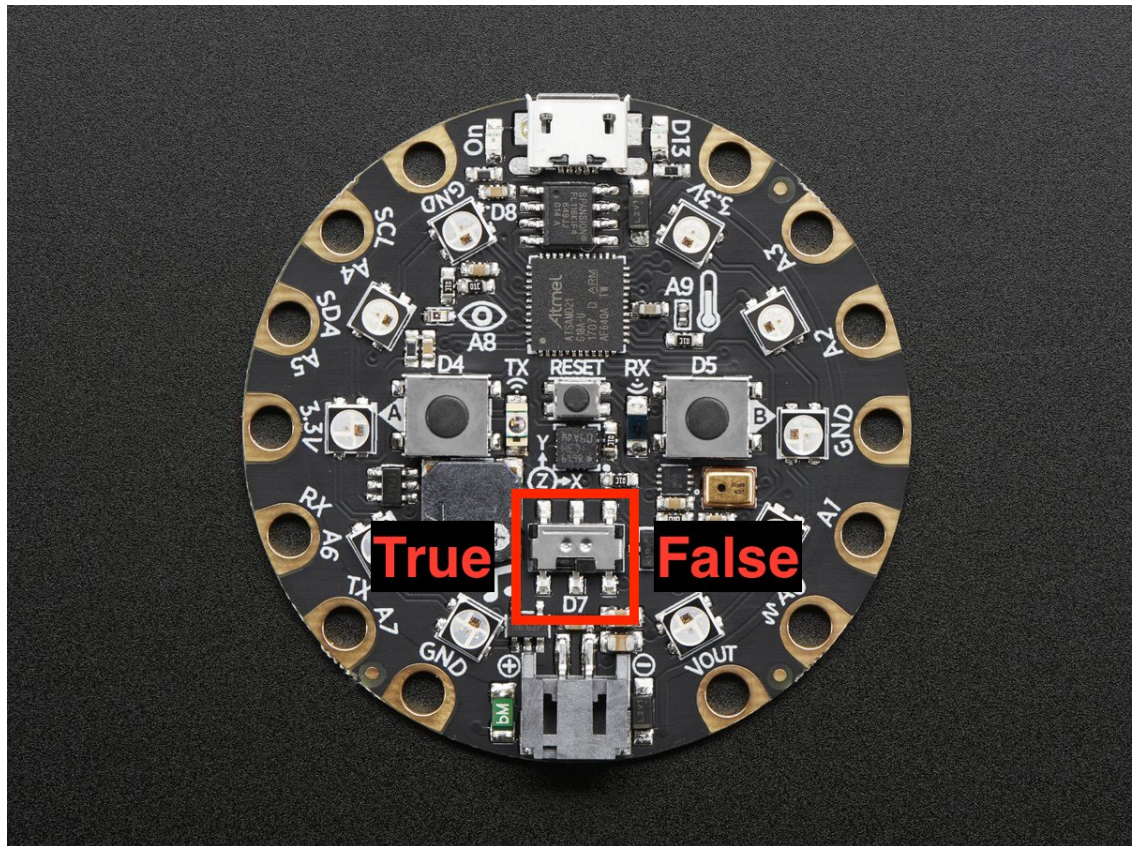
from adafruit_circuitplayground import cp

while True:
    if cp.button_a:
        cp.start_tone(262)
    elif cp.button_b:
        cp.start_tone(294)
    else:
        cp.stop_tone()

```

switch

True when the switch is to the left next to the music notes. False when it is to the right towards the ear.



To use with the Circuit Playground Express or Bluefruit:

```

from adafruit_circuitplayground import cp
import time

while True:
    print("Slide switch:", cp.switch)
    time.sleep(0.1)

```

tapped

True once after detecting a tap. Requires `cp.detect_taps`.



Tap the Circuit Playground once for a single-tap, or quickly tap twice for a double-tap.

To use with Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

cp.detect_taps = 1

while True:
    if cp.tapped:
        print("Single tap detected!")
```

To use single and double tap together, you must have a delay between them. It will not function properly without it. This example uses both by counting a specified number of each type of tap before moving on in the code.

```
from adafruit_circuitplayground import cp

# Set to check for single-taps.
cp.detect_taps = 1
tap_count = 0

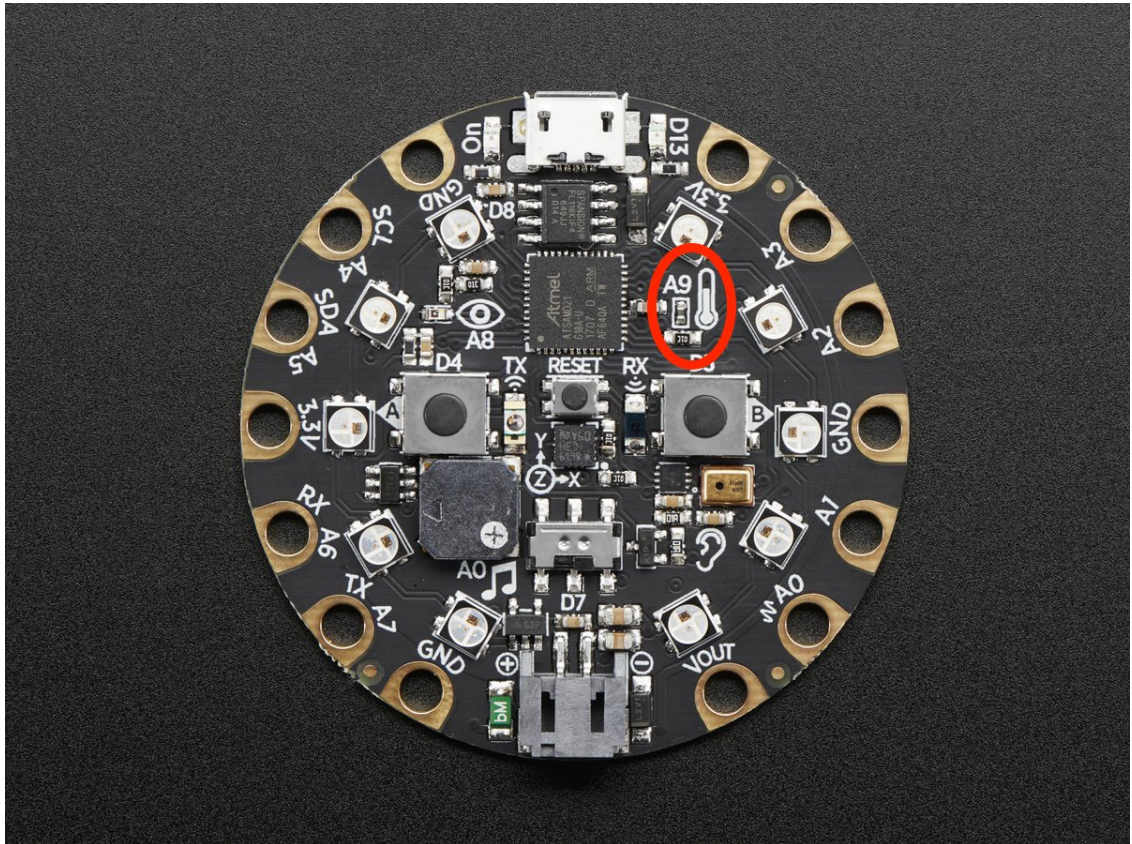
# We're looking for 2 single-taps before moving on.
while tap_count < 2:
    if cp.tapped:
        tap_count += 1
print("Reached 2 single-taps!")

# Now switch to checking for double-taps
tap_count = 0
cp.detect_taps = 2

# We're looking for 2 double-taps before moving on.
while tap_count < 2:
    if cp.tapped:
        tap_count += 1
print("Reached 2 double-taps!")
print("Done.")
```

`temperature`

The temperature in Celsius.



Converting this to Fahrenheit is easy!

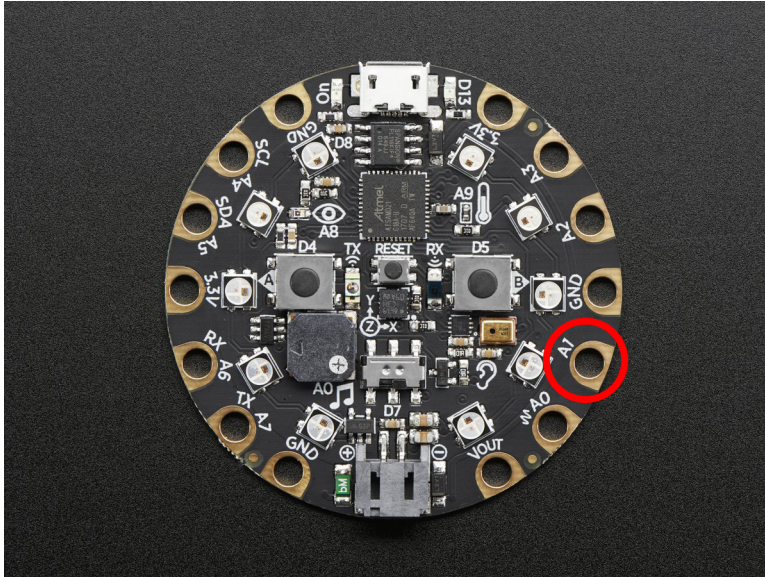
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp
import time

while True:
    temperature_c = cp.temperature
    temperature_f = temperature_c * 1.8 + 32
    print("Temperature celsius:", temperature_c)
    print("Temperature fahrenheit:", temperature_f)
    time.sleep(1)
```

`touch_A1`

Detect touch on capacitive touch pad A1.



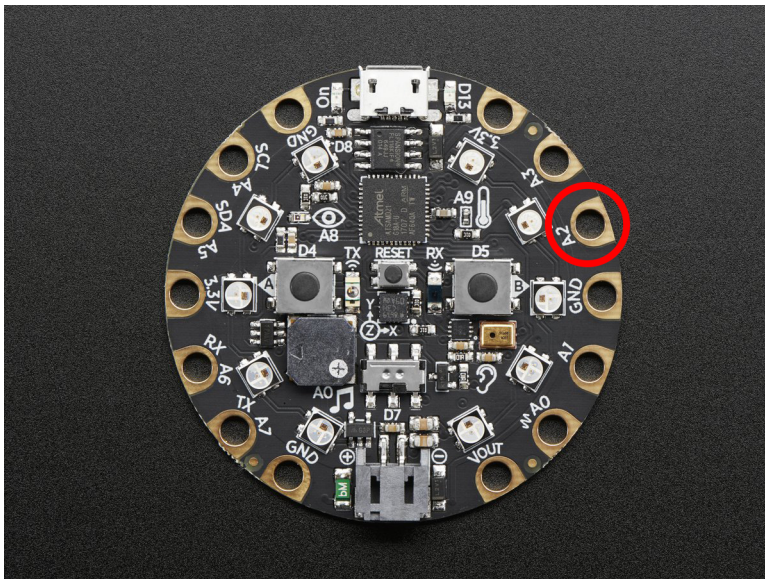
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A1:
        print('Touched pad A1')
```

`touch_A2`

Detect touch on capacitive touch pad A2.



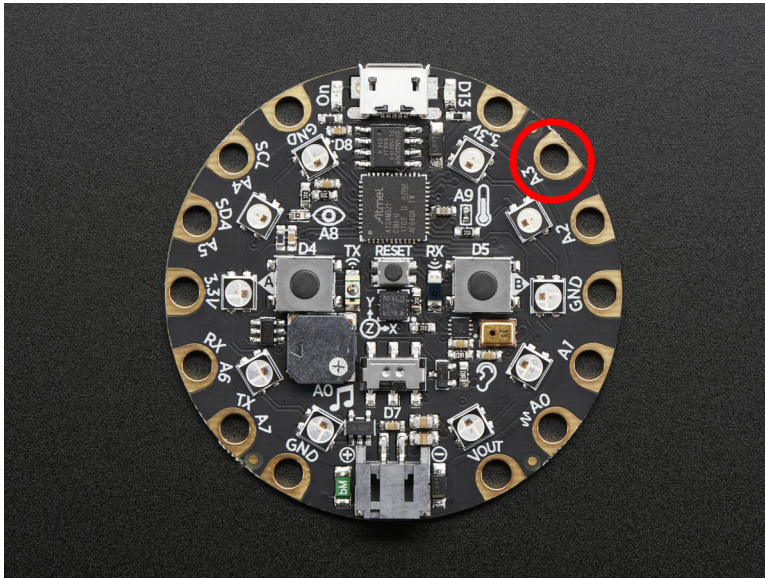
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A2:
        print('Touched pad A2')
```

touch_A3

Detect touch on capacitive touch pad A3.



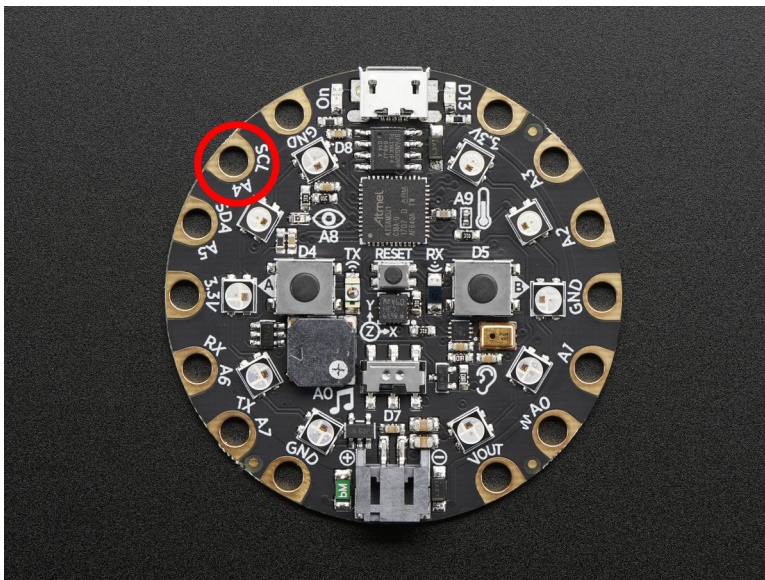
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A3:
        print('Touched pad A3')
```

touch_A4

Detect touch on capacitive touch pad A4.



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp
```

(continues on next page)


```
while True:
    if cp.touch_A4:
        print('Touched pad A4')
```

Detect touch on capacitive touch pad A5.

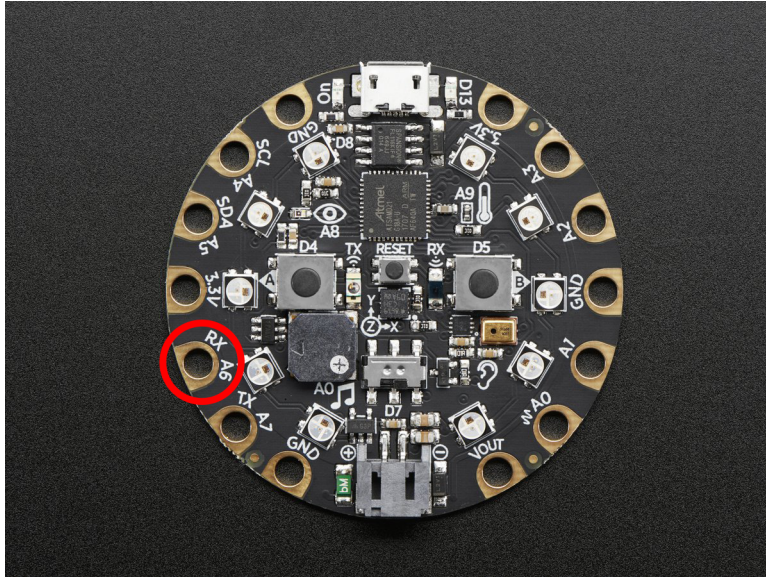
A circular black PCB with various electronic components and gold-plated mounting holes. A red circle highlights a specific pin labeled 'SDA' and '7.5'.

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A5:
        print('Touched pad A5')
```

Detect touch on capacitive touch pad A6.

6.2. adafruit_circuitplayground.circuit_playground_base



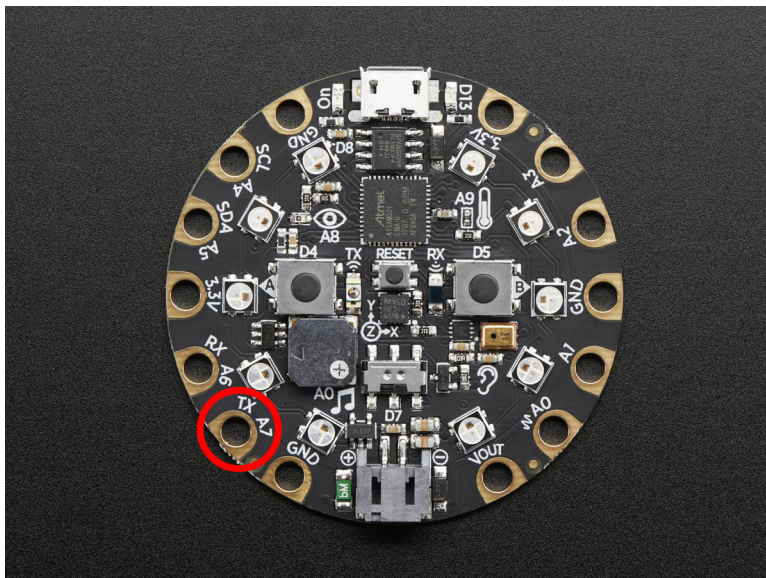
To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A6:
        print('Touched pad A6')
```

touch_TX

Detect touch on capacitive touch pad TX (also known as A7 on the Circuit Playground Express) Note: can be called as touch_A7 on Circuit Playground Express.



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
```

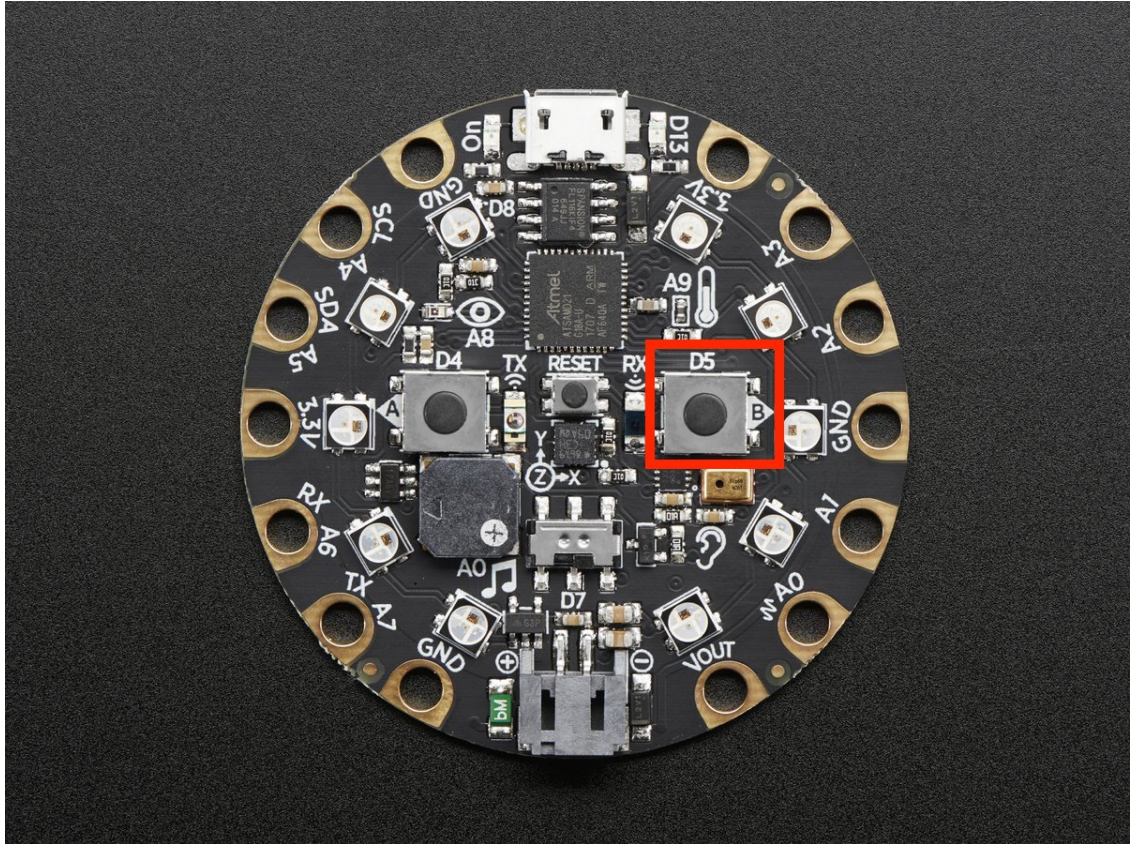
(continues on next page)

(continued from previous page)

```
if cp.touch_A7:
    print('Touched pad A7')
```

were_pressed

Returns a set of the buttons that have been pressed



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    print(cp.were_pressed)
```

class `adafruit_circuitplayground.circuit_playground_base.Photocell` (*pin*)

Simple driver for analog photocell on the Circuit Playground Express and Bluefruit.

light

Light level.

6.3 `adafruit_circuitplayground.bluefruit`

CircuitPython helper for Circuit Playground Bluefruit.

- Author(s): Kattni Rembor

6.3.1 Implementation Notes

Hardware:

- Circuit Playground Bluefruit

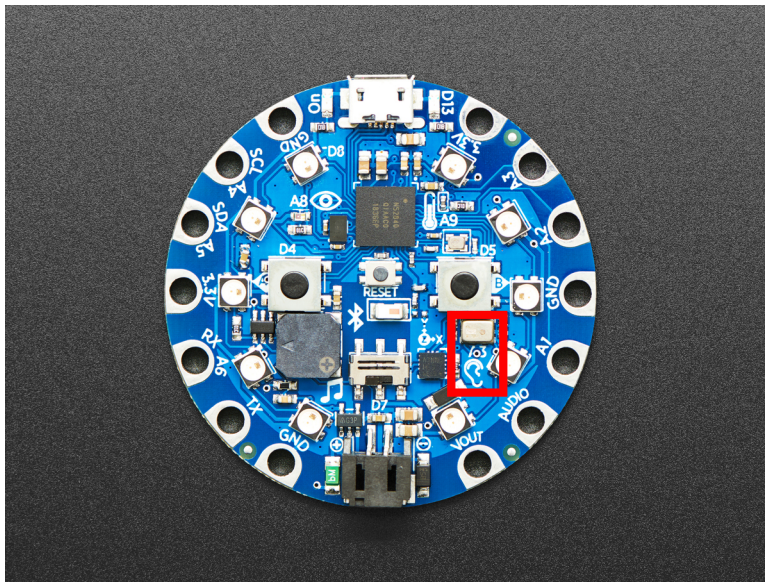
class adafruit_circuitplayground.bluefruit.**Bluefruit**

Represents a single CircuitPlayground Bluefruit.

loud_sound (*sound_threshold=200*)

Utilise a loud sound as an input.

Parameters **sound_threshold** (*int*) – Threshold sound level must exceed to return true (Default: 200)



This example turns the LEDs red each time you make a loud sound. Try clapping or blowing onto the microphone to trigger it.

```
from adafruit_circuitplayground.bluefruit import cpb

while True:
    if cpb.loud_sound():
        cpb.pixels.fill((50, 0, 0))
    else:
        cpb.pixels.fill(0)
```

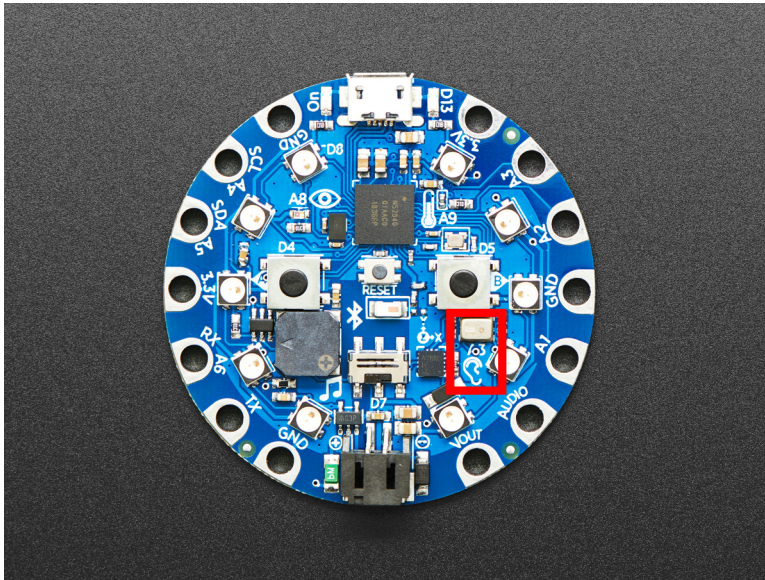
You may find that the code is not responding how you would like. If this is the case, you can change the loud sound threshold to make it more or less responsive. Setting it to a higher number means it will take a louder sound to trigger. Setting it to a lower number will take a quieter sound to trigger. The following example shows the threshold being set to a higher number than the default.

```
from adafruit_circuitplayground.bluefruit import cpb

while True:
    if cpb.loud_sound(sound_threshold=300):
        cpb.pixels.fill((50, 0, 0))
    else:
        cpb.pixels.fill(0)
```

sound_level

Obtain the sound level from the microphone (sound sensor).



This example prints the sound levels. Try clapping or blowing on the microphone to see the levels change.

```
from adafruit_circuitplayground.bluefruit import cpb

while True:
    print(cpb.sound_level)
```

`adafruit_circuitplayground.bluefruit.cpb` = `<adafruit_circuitplayground.bluefruit.Bluefruit`
Object that is automatically created on import.

To use, simply import it from the module:

```
from adafruit_circuitplayground.bluefruit import cpb
```

6.4 `adafruit_circuitplayground.express`

CircuitPython helper for Circuit Playground Express.

Hardware:

- [Circuit Playground Express](#)
- Author(s): Kattni Rembor, Scott Shawcroft

class `adafruit_circuitplayground.express.Express`
Represents a single CircuitPlayground Express. Do not use more than one at a time.

loud_sound

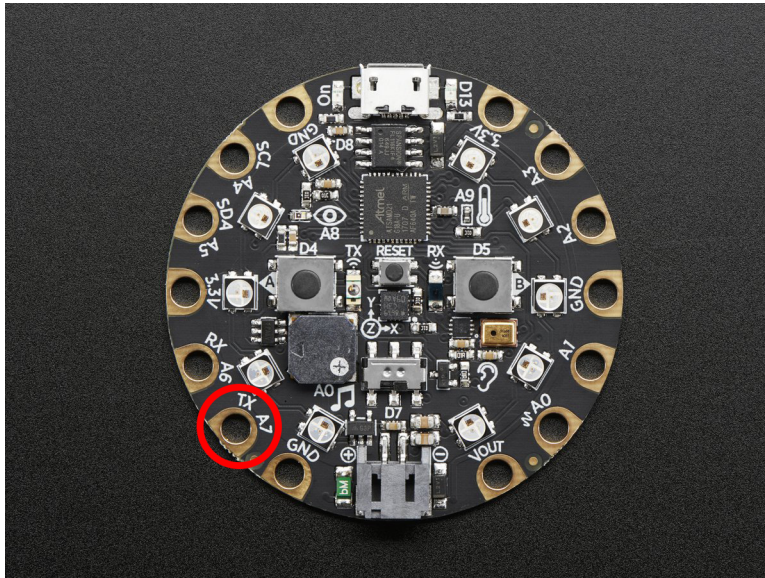
This feature is not supported on Circuit Playground Express.

sound_level

This feature is not supported on Circuit Playground Express.

touch_A7

Detect touch on capacitive touch pad TX (also known as A7 on the Circuit Playground Express) Note: can be called as touch_A7 on Circuit Playground Express.



To use with the Circuit Playground Express or Bluefruit:

```
from adafruit_circuitplayground import cp

while True:
    if cp.touch_A7:
        print('Touched pad A7')
```

`adafruit_circuitplayground.express.cpx` = `<adafruit_circuitplayground.express.Express object>`
Object that is automatically created on import.

To use, simply import it from the module:

```
from adafruit_circuitplayground.express import cpx
```


CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

a

`adafruit_circuitplayground.bluefruit,`

[45](#)

`adafruit_circuitplayground.circuit_playground_base,`

[27](#)

`adafruit_circuitplayground.express,` [47](#)

A

acceleration (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 27

adafruit_circuitplayground.bluefruit (module), 45

adafruit_circuitplayground.circuit_playground_base (module), 27

adafruit_circuitplayground.express (module), 47

adjust_touch_threshold() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 28

light (*adafruit_circuitplayground.circuit_playground_base.Photocell* attribute), 45

loud_sound (*adafruit_circuitplayground.express.Express* attribute), 47

loud_sound() (*adafruit_circuitplayground.bluefruit.Bluefruit* method), 46

P

Photocell (class in *adafruit_circuitplayground.circuit_playground_base*), 45

pixels (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 32

play_file() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 33

play_tone() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 34

B

Bluefruit (class in *adafruit_circuitplayground.bluefruit*), 46

button_a (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 28

button_b (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 29

R

red_led (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 35

C

CircuitPlaygroundBase (class in *adafruit_circuitplayground.circuit_playground_base*), 27

cpb (in module *adafruit_circuitplayground.bluefruit*), 47

cpx (in module *adafruit_circuitplayground.express*), 48

S

shake() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 35

sound_level (*adafruit_circuitplayground.bluefruit.Bluefruit* attribute), 46

sound_level (*adafruit_circuitplayground.express.Express* attribute), 47

start_tone() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 36

stop_tone() (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* method), 37

switch (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 38

D

detect_taps (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 30

E

Express (class in *adafruit_circuitplayground.express*), 47

L

light (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 31

T

tapped (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase* attribute), 38

`temperature` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [39](#)

`touch_A1` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [40](#)

`touch_A2` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [41](#)

`touch_A3` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [41](#)

`touch_A4` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [42](#)

`touch_A5` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [43](#)

`touch_A6` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [43](#)

`touch_A7` (*adafruit_circuitplayground.express.Express*
attribute), [47](#)

`touch_TX` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [44](#)

W

`were_pressed` (*adafruit_circuitplayground.circuit_playground_base.CircuitPlaygroundBase*
attribute), [45](#)