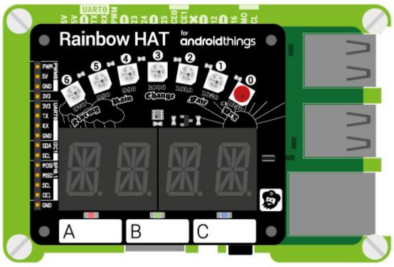
Pimoroni Rainbow Hat

The Pimoroni Rainbow Hat is a nice little hat that packs a lot of different functions into it. We are going to play around with the different things you can do with this hat and the Python programming language.



Install and enable libraries and interfaces

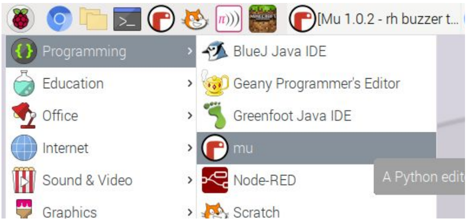
You do not need to carry out this section at the Jam as we have already completed this for you, but if you were to use one at home you would need to follow these steps.

1. Install the Python 3 Rainbow Hat library (if it is not there already) by typing the following into a terminal window:
   ```
   curl https://get.pimoroni.com/rainbowhat | bash
   ```
2. When prompted if you want to continue, press “y” and Enter
3. When asked if you wish to perform a full install, press “y” and Enter

Writing our Python

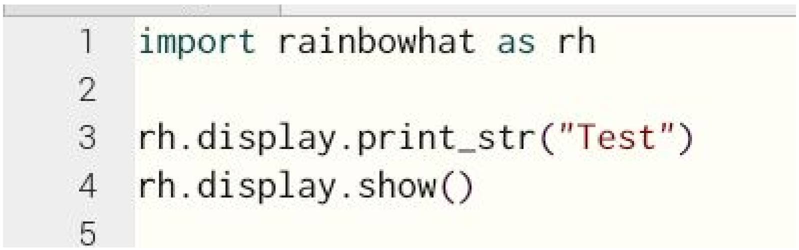
Will will be using the Mu Python IDE. Open it by going to the “**Raspberry Pi menu > Programming > mu**”. *If you do not have mu available on your Raspberry Pi, you can install by typing the following into a terminal window:*

```
sudo apt install mu-editor
```



Displaying Text

Think of a four letter word (not a rude one!) and get it to display on the Rainbow Hat… Now type the following code into a new file in the mu editor (make sure that the mode is “Python” - the mode is indicated in the bottom right hand corner of mu):



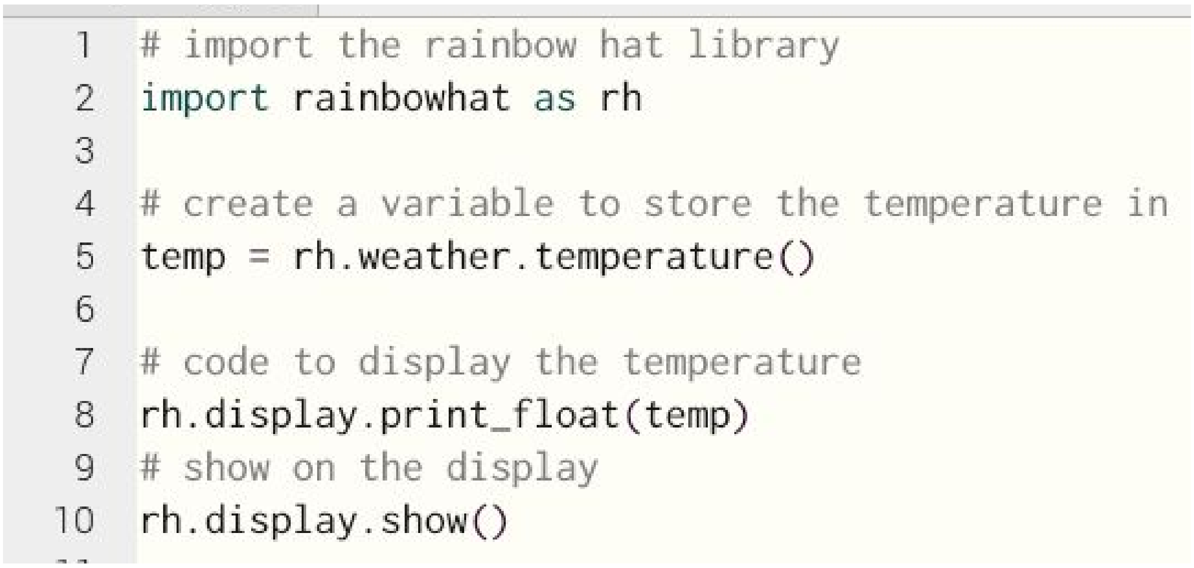
```
import rainbowhat as rh

rh.display.print_str("Test")
rh.display.show()

```

Save and run the file. You should see your message displayed. Now try these extensions:

1. Does it work if you use a word bigger than 4 letters?
2. Try lowercase and uppercase letters
3. Can you code two short words to display one after each other?

Temperature Reading

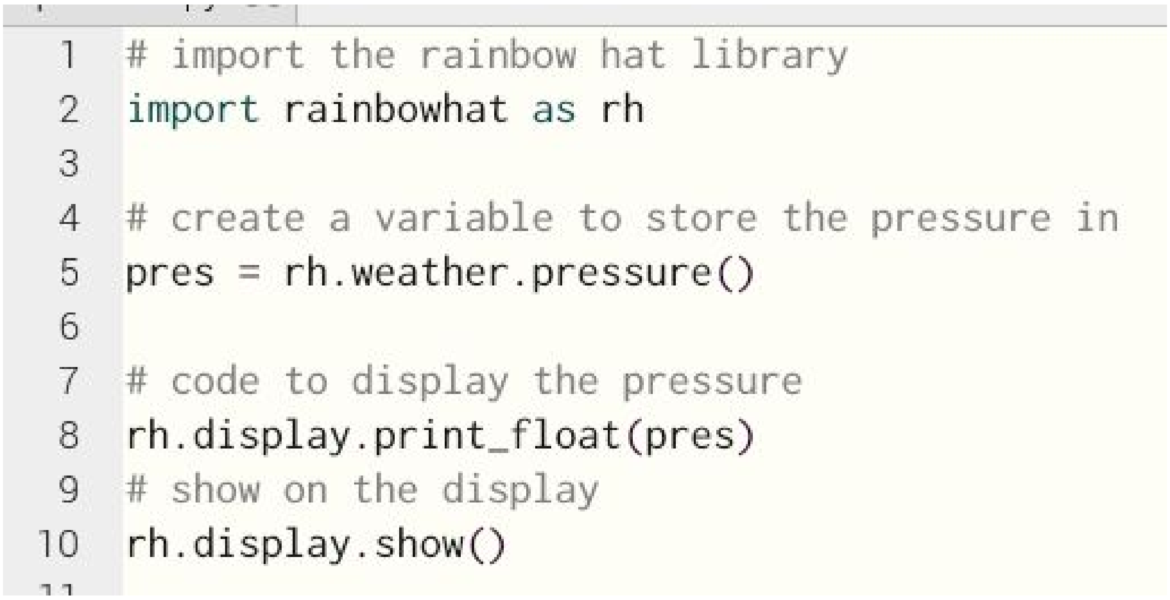
Use the temperature sensor to find out how warm it is in here (or isn't!). Type the following code into a new file in the mu editor and then save and run it to display the temperature on the Rainbow Hat:



```
# import the rainbow hat library
import rainbowhat as rh

# create a variable to store the temperature in
temp = rh.weather.temperature()

# code to display the temperature
rh.display.print_float(temp)
# show on the display
rh.display.show()
```

Pressure Reading

Take a pressure reading with the Rainbow Hat… Type the following code into a new file in the mu editor and then save and run it it to display the current atmospheric pressure of the Rainbow Hat:



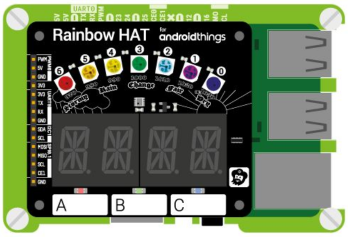
```
# import the rainbow hat library
import rainbowhat as rh

# create a variable to store the pressure in
pres = rh.weather.pressure()

# code to display the pressure
rh.display.print_float(pres)
# show on the display
rh.display.show()
```

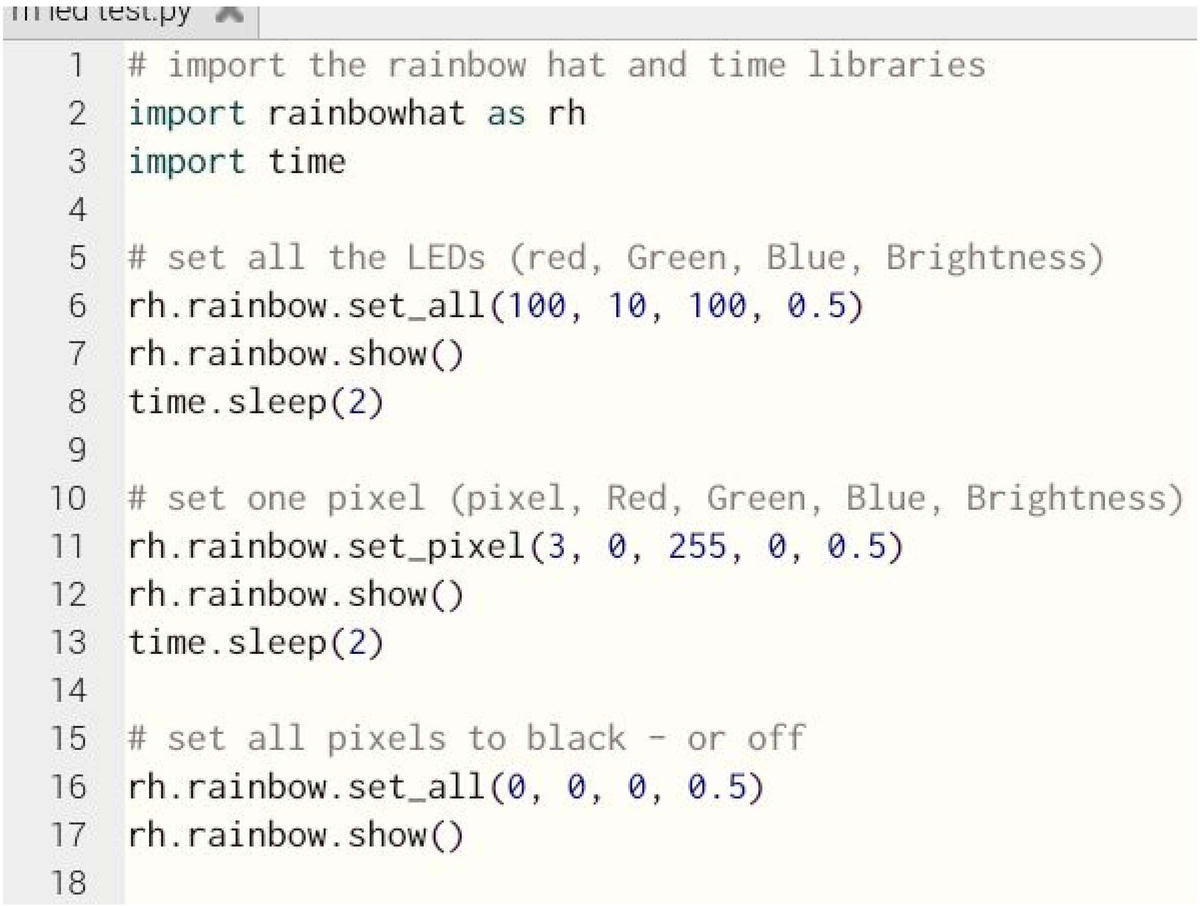
Extension Task

Can you combine what you have done in the previous two examples to create a program that displays the temperature followed by the pressure?

Light Up the LEDs

We will now control the 7 RGB LEDs on the Rainbow Hat. Type the following code into a new file in mu and then save and run it:





```
# import the rainbow hat and time libraries
import rainbowhat as rh
import time

# set all the LEDs (red, Green, Blue, Brightness)
rh.rainbow.set_all(100, 10, 100, 0.5)
rh.rainbow.show()
time.sleep(2)

# set one pixel (pixel, Red, Green, Blue, Brightness)
rh.rainbow.set_pixel(3, 0, 255, 0, 0.5)
rh.rainbow.show()
time.sleep(2)

# set all pixels to black - or off
rh.rainbow.set_all(0, 0, 0, 0.5)
rh.rainbow.show()

```

Extension Tasks

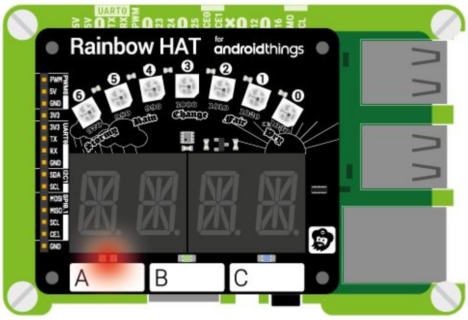
1. Can you turn on LEDs 1, 3, 4 and 6?
2. Turn on the LEDs but individually set the colour of each one
3. Create an LED chase. Set each individual LED so that it flashes once after the previous LED

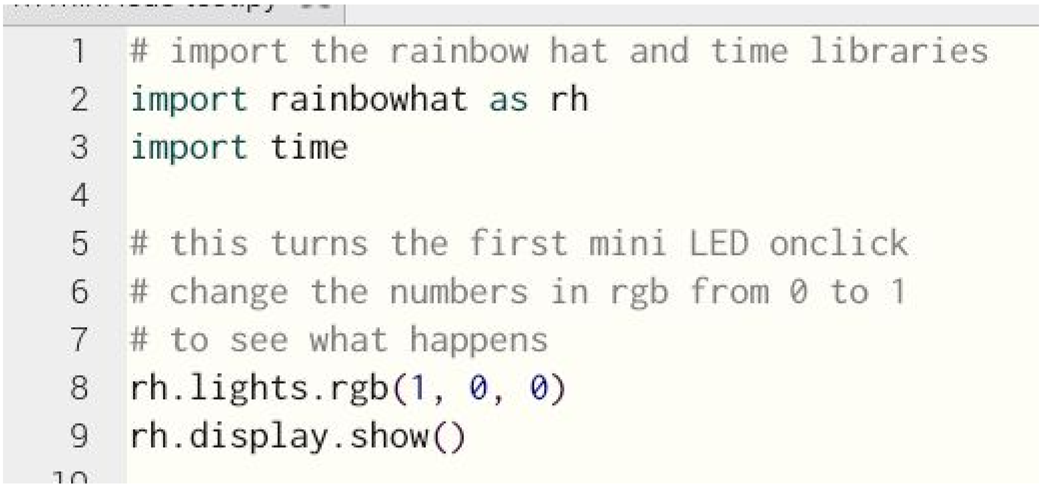
Hints and Debugging

1. Ensure your code is written as above. Python is case sensitive!
2. Each colour value can be between 0 and 255
3. Brightness can be between 0 and 1

Mini LEDs

We will program the mini LED indicators on and off and create a short light sequence. Copy the following code into a new file in mu and then save and run it:





```
# import the rainbow hat and time libraries
import rainbowhat as rh
import time

# this turns the first mini LED onclick
# change the numbers in rgb from 0 to 1
# to see what happens
rh.lights.rgb(1, 0, 0)
rh.display.show()
```

Extension Tasks

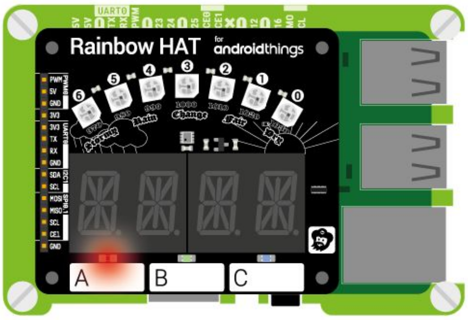
1. Turn on the green mini LED
2. Turn on all the mini LEDs
3. Use time.sleep(2) to flash the red, green and blue mini LEDs on and off every two seconds

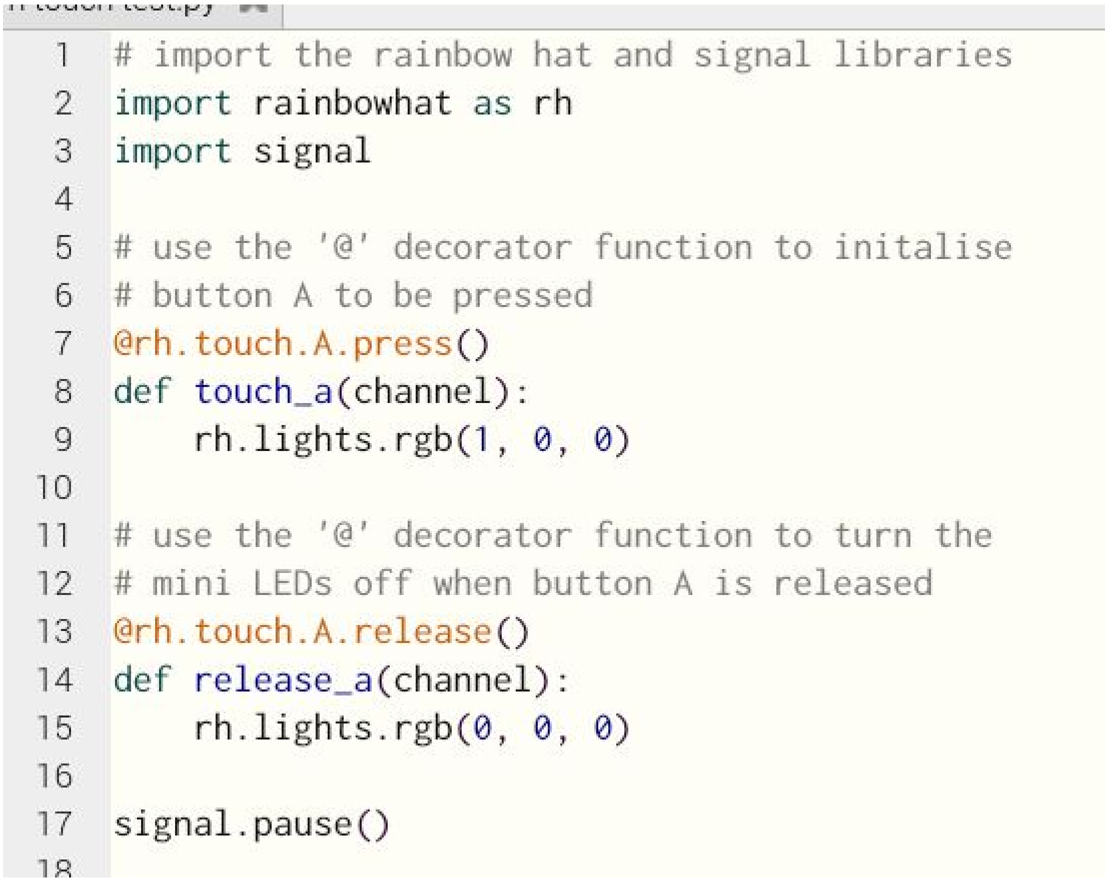
Hints and Debugging

1. The value “1” indicates that the LED is on, the value “0” means that the LED is off
2. The colour of mini LEDs are set, you cannot change the colour.
3. To use the time.sleep code you will need to add the “import time” line to the top of your programme

Touch Buttons

We will now create code that enables the touch buttons to turn the mini LEDs on or off. Copy the following code into a new file in mu and then save and run it:





```
# import the rainbow hat and signal libraries
import rainbowhat as rh
import signal

# use the '@' decorator function to initalise
# button A to be pressed
@rh.touch.A.press()
def touch_a(channel):
    rh.lights.rgb(1, 0, 0)

# use the '@' decorator function to turn the
# mini LEDs off when button A is released
@rh.touch.A.release()
def release_a(channel):
    rh.lights.rgb(0, 0, 0)

signal.pause()
```

Extension Tasks

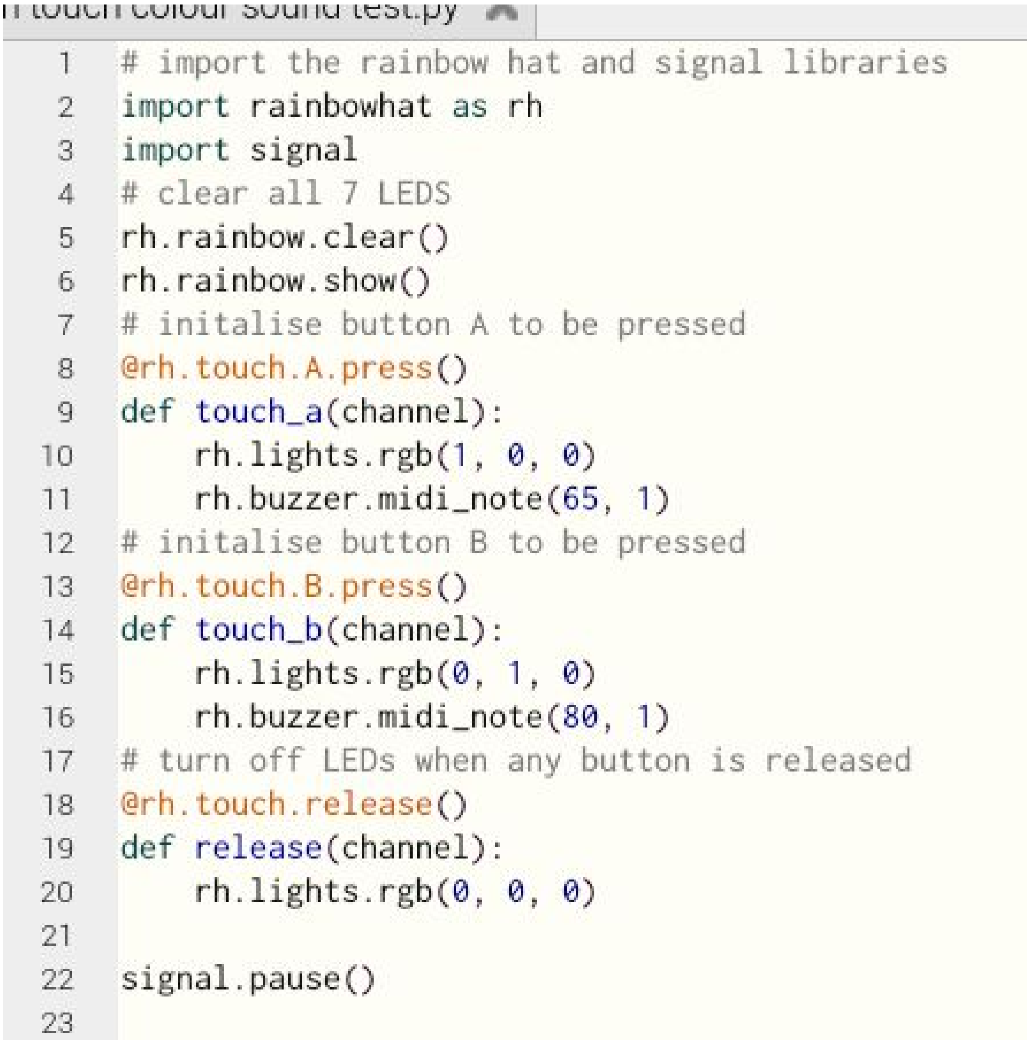
1. Set Button B to turn the green mini LED on and off
2. Write code that enables each button; A, B or C, to turn on its corresponding LED; red, green or blue.
3. Write code that controls the 7 multi-colour LEDs using the touch buttons

Hints and Debugging

1. Ensure that you are setting the correct button letter when using the “@rh.touch.C.press” code. This controls button C
2. Note that the button letter is written in uppercase

Musical Touch Colour

We will now write a program that assigns a colour and sound to each touch button. Copy the following code into a new file in mu and then save and run it:



```
# import the rainbow hat and signal libraries
import rainbowhat as rh
import signal
# clear all 7 LEDS
rh.rainbow.clear()
rh.rainbow.show()
# initalise button A to be pressed
@rh.touch.A.press()
def touch_a(channel):
    rh.lights.rgb(1, 0, 0)
    rh.buzzer.midi_note(65, 1)
# initalise button B to be pressed
@rh.touch.B.press()
def touch_b(channel):
    rh.lights.rgb(0, 1, 0)
    rh.buzzer.midi_note(80, 1)
# turn off LEDs when any button is released
@rh.touch.release()
def release(channel):
    rh.lights.rgb(0, 0, 0)

signal.pause()

```

Extension Tasks

1. Add button C to play a different note when touched
2. Change the pitch of the notes played
3. Turn all of the 7 multi-colour LEDs a particular colour when a note is played
4. Can you create a three note piano with flashing LEDs?

Hints and Debugging

1. Take note of which button controls which sound
2. Ensure that the values are within the permitted ranges: colour; 0 to 255 and brightness 0 to 1
3. Check out this site to see what different notes you can play with the “midi_note” value: https://computermusicresource.com/midikeys.html