MIT AI2 204
IoT with MIT App Inventor

Fundamental

Xincheng Tang

GPIO music box

In this project, you will build a button-controlled "music box" that plays different sounds when different buttons are pressed.

https://www.youtube.com/watch?v=2izvSzQWYak&feature=emb_title

What you will learn

- Play sounds in Python with `pygame`
- Use the Python `gpiozero` library to connect button presses to function calls
- Use the dictionary data structure in Python

GPIO music box

What you will need

Hardware

- A Raspberry Pi computer
- A breadboard
- Four (4) tactile switches (to make buttons)
- Five (5) pin-to-socket jumper leads
- Four (4) pin-to-pin jumper leads
- Speakers or headphones

GPIO music box
Set up your project

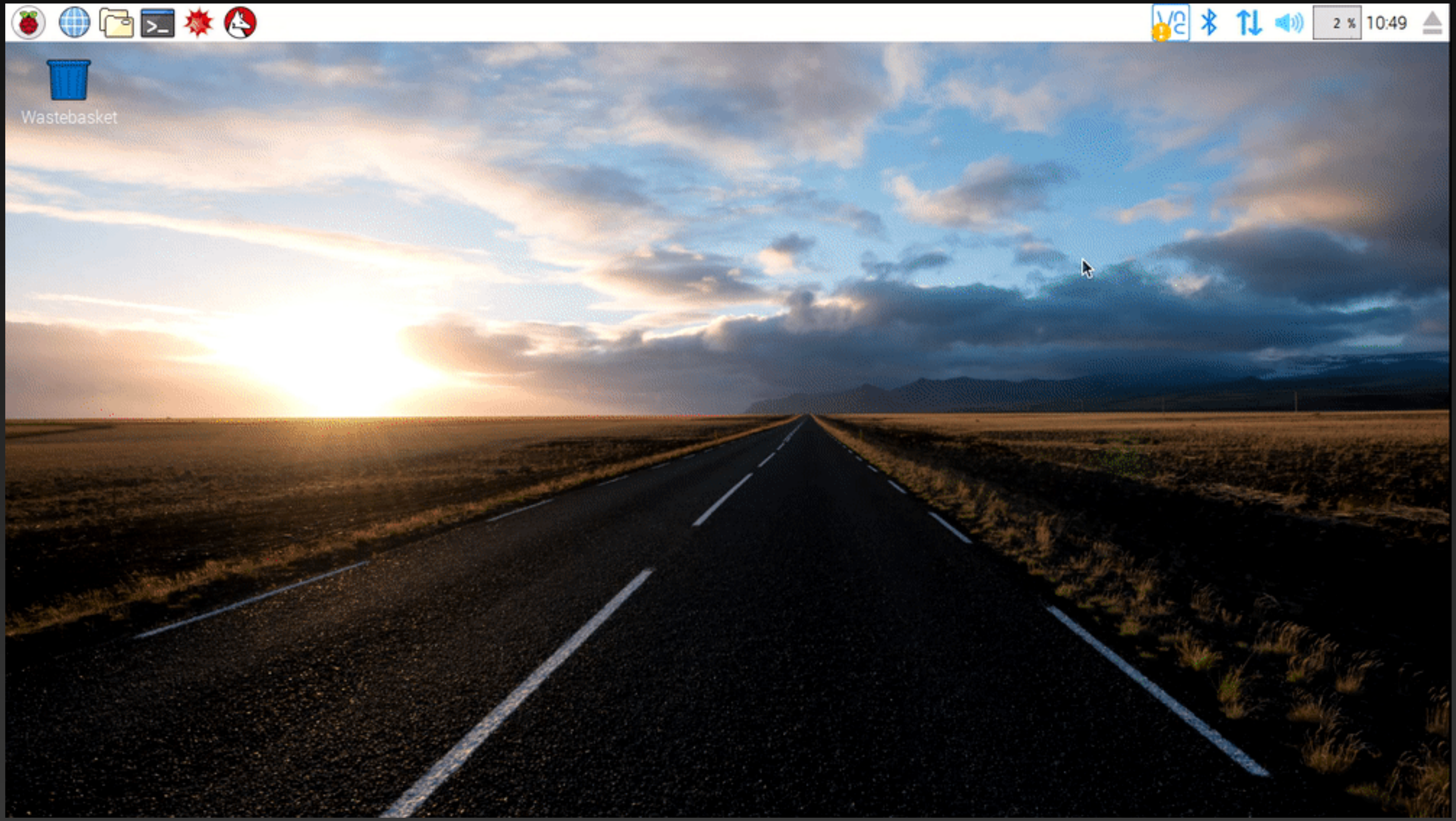
You will need some sample sounds for this project. There are lots of sound files on Raspbian, but it can be a bit difficult to play them using Python. However, you can convert the sound files to a different file format that you can use in Python more easily.

First, in your `home` directory, create a directory called `gpio-music-box`. You will use the new directory to store all your files for the project.

What method you will use to create your directories in Raspberry Pi?

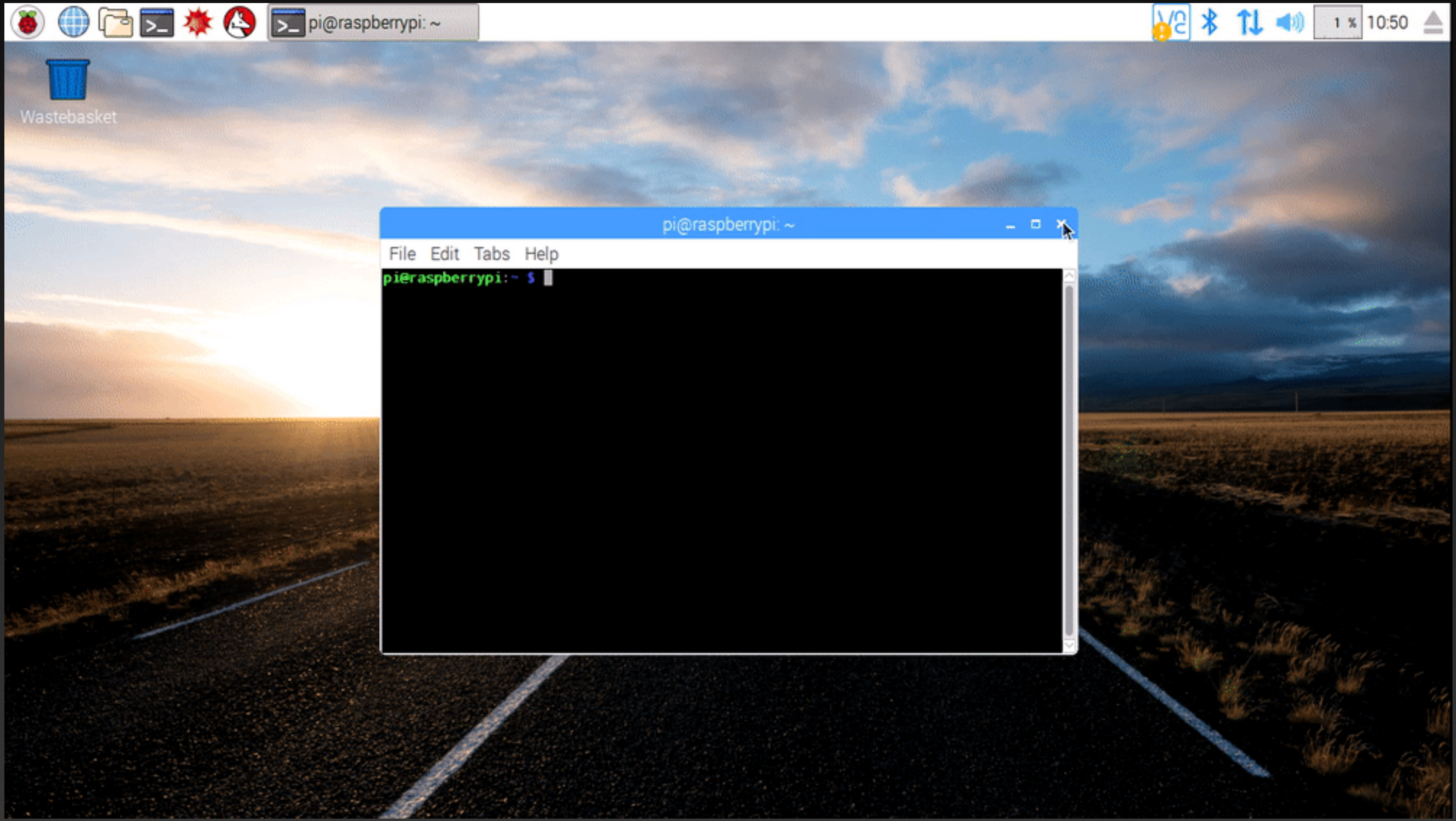
GPIO music box
Set up your project

Method 1 - Using the GUI



GPIO music box
Set up your project

Method 1 - Using the Terminal



```
pi@raspberrypi:~ $
```

GPIO music box
Set up your project

Use the same method as before to create a new directory called `samples` in your `gpio-music-box` directory.

There are lots of sample sounds stored in `/usr/share/sonic-pi/samples`. In the next step, you will copy these sounds into the `gpio-music-box/samples` directory.

Type the following lines to copy all the files from one directory to the other:

```
cp /usr/share/sonic-pi/samples/* ~/gpio-music-box/samples/.
```

GPIO music box
Copy the sample sounds

Use the same method as before to create a new directory called `samples` in your `gpio-music-box` directory.

There are lots of sample sounds stored in `/usr/share/sonic-pi/samples`. In the next step, you will copy these sounds into the `gpio-music-box/samples` directory.

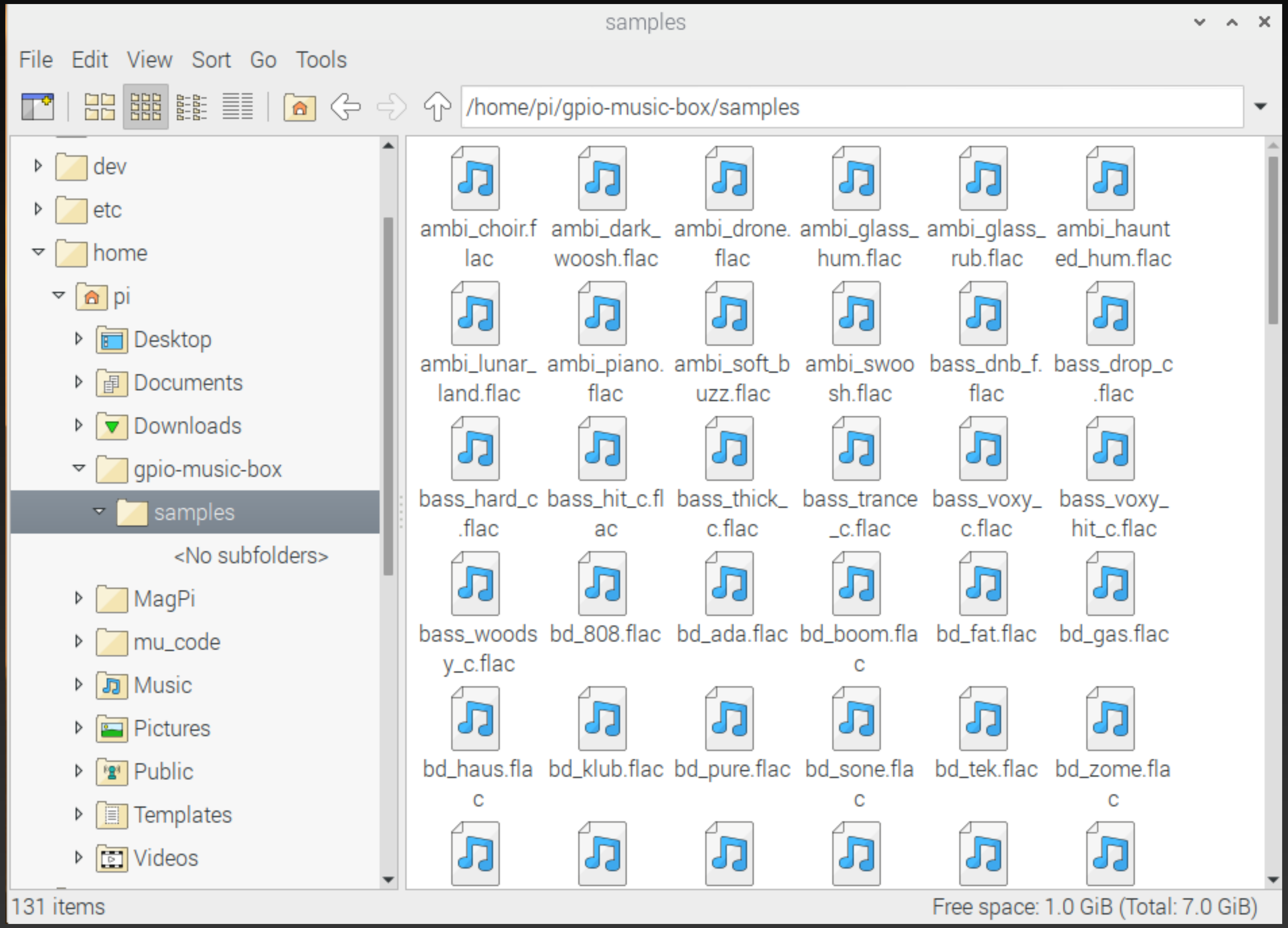
Type the following lines to copy all the files from one directory to the other:

```
cp /usr/share/sonic-pi/samples/* ~/gpio-music-box/samples/.
```

When you have done that, you should be able to see all the `.flac` sound files in the `samples` directory.

GPIO music box
Copy the sample sounds

When you have done that, you should be able to see all the `.flac` sound files in the `samples` directory.

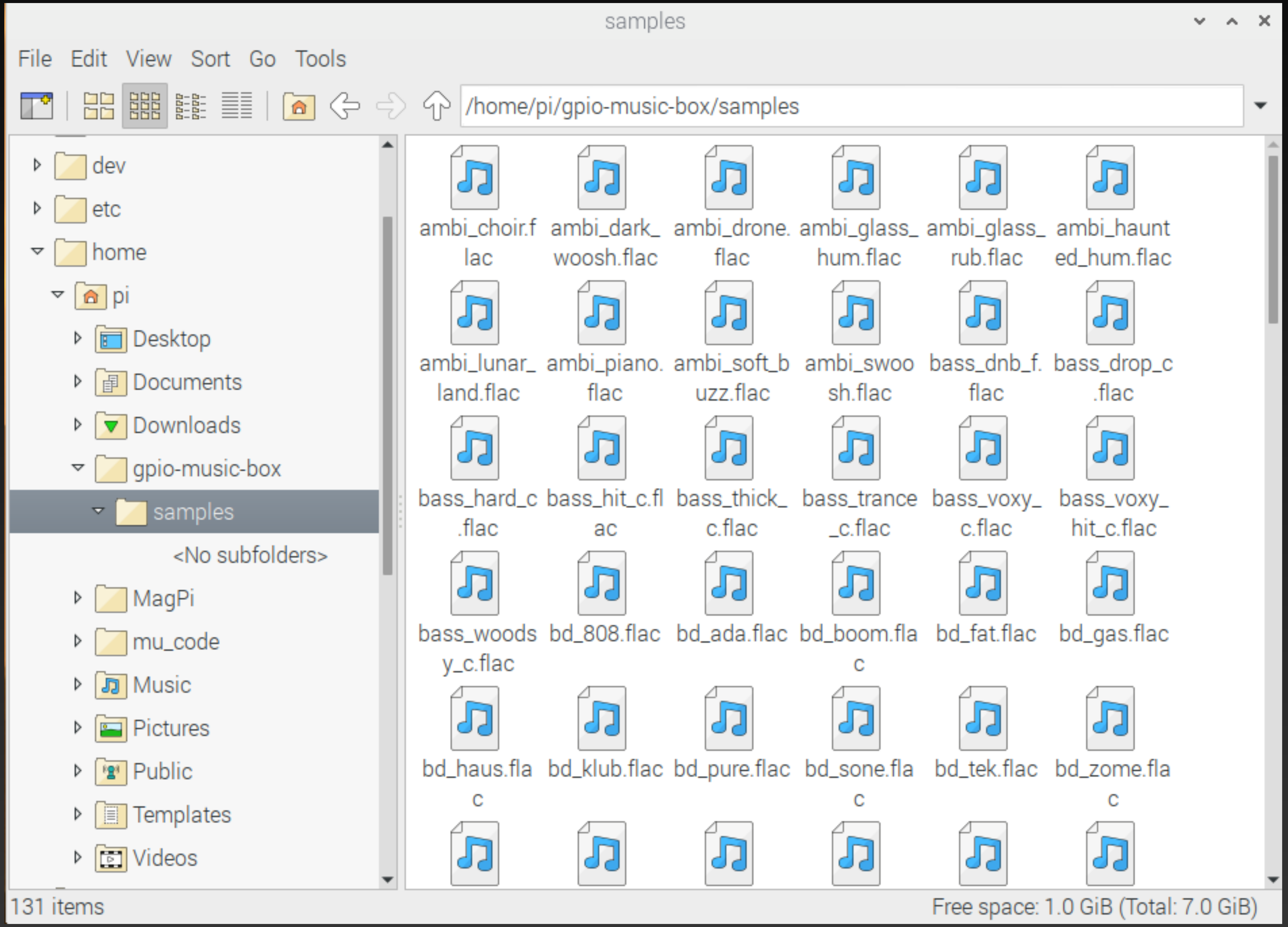


GPIO music box
Covert the sound files

To play the sound files using Python, you need to convert the files from `.flac` files to `.wav` files.

In the terminal, change into your `samples` directory.

```
cd ~/gpio-music-box/samples
```



GPIO music box
Covert the sound files

In your terminal, type the following commands. This will convert all the `.flac` files to `.wav` files, then delete the old files.

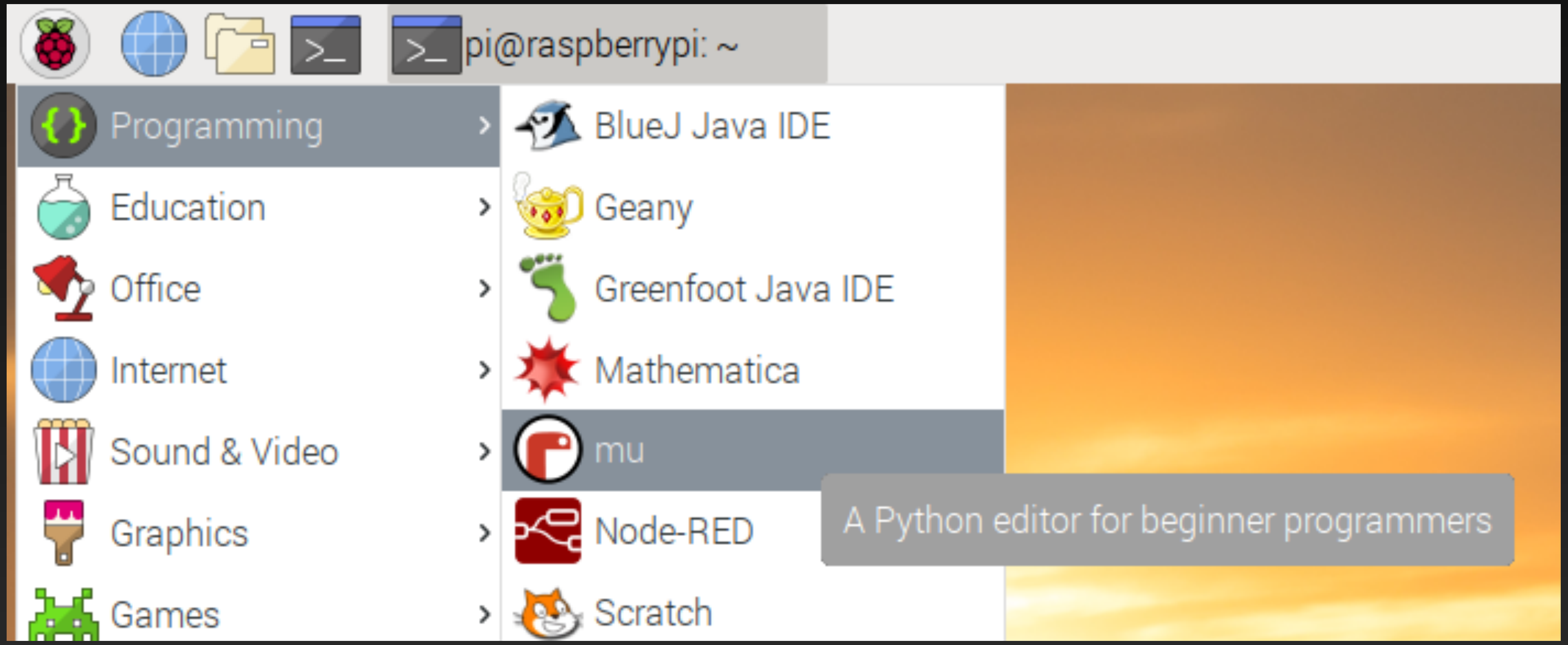
```
for f in *.flac; do ffmpeg -i "$f" "${f%.flac}.wav"; done
```

```
rm *.flac
```

GPIO music box
Play sounds

Next, you will start to write your Python code. You can use any text editor or IDE to do this — Mu is always a good choice.

To start to create the instruments of your music box, you need to test whether Python can play some of the samples that you have copied.



GPIO music box
Play sounds

First, import and initialise the `pygame` module for playing sound files.

```
import pygame
```

```
pygame.init()
```

Save this file in your `gpio-music-box` directory.

Choose four sound files that you want to use for your project, for example:

```
drum_tom_mid_hard.wav
drum_cymbal_hard.wav
drum_snare_hard.wav
drum_cowbell.wav
```

GPIO music box
Play sounds

Then, create a Python object that links to one of these sound files. Give the file its own unique name. For example:

```
drum = pygame.mixer.Sound(“/home/pi/gpio-music-box/samples/drum_tom_mid_hard.wav")
```

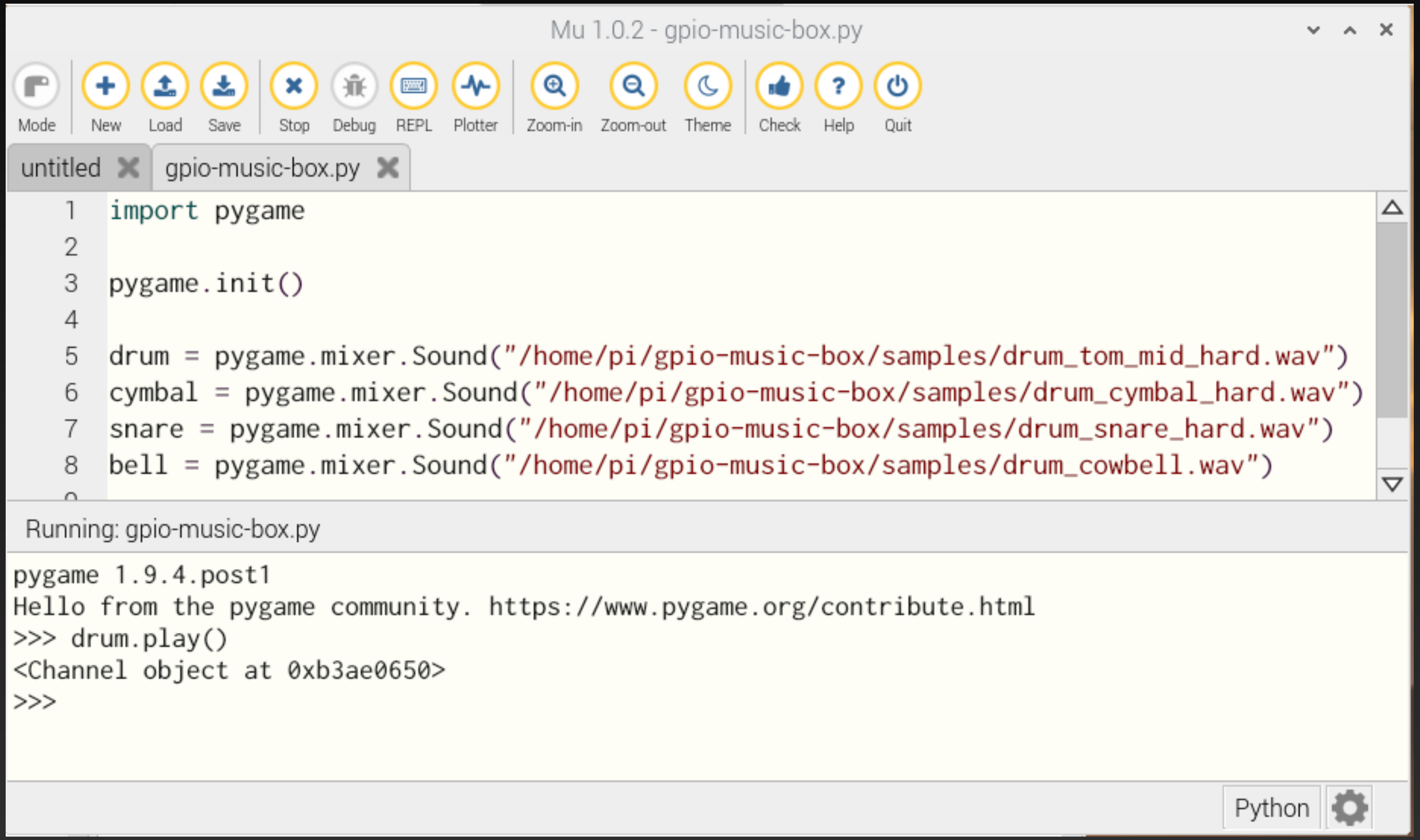
Create named objects for your remaining three sounds.

```
cymbal = pygame.mixer.Sound(“/home/pi/gpio-music-box/samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound(“/home/pi/gpio-music-box/samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound(“/home/pi/gpio-music-box/samples/drum_cowbell_hard.wav")
```

Save and run your code. Then, in the shell at the bottom of the Mu editor, use `.play()` commands to play the sounds.

GPIO music box
Play sounds

If you don't hear any sound, check that your speakers or headphones are working and that the volume is turned up.



```
import pygame

pygame.init()

drum = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_tom_mid_hard.wav")
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-box/samples/drum_cowbell.wav")
```

```
pygame 1.9.4.post1
Hello from the pygame community. https://www.pygame.org/contribute.html
>>> drum.play()
<Channel object at 0xb3ae0650>
>>>
```

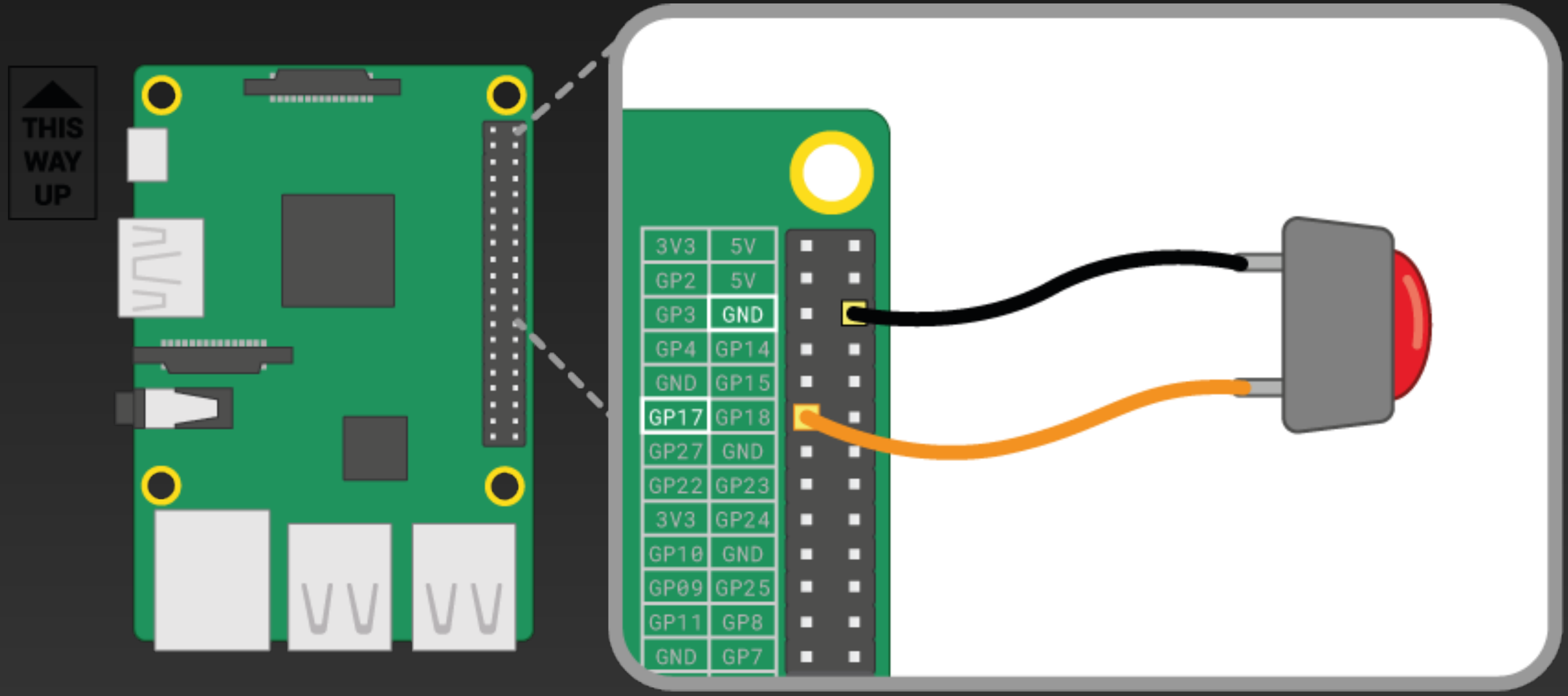
GPIO music box
Connect your buttons

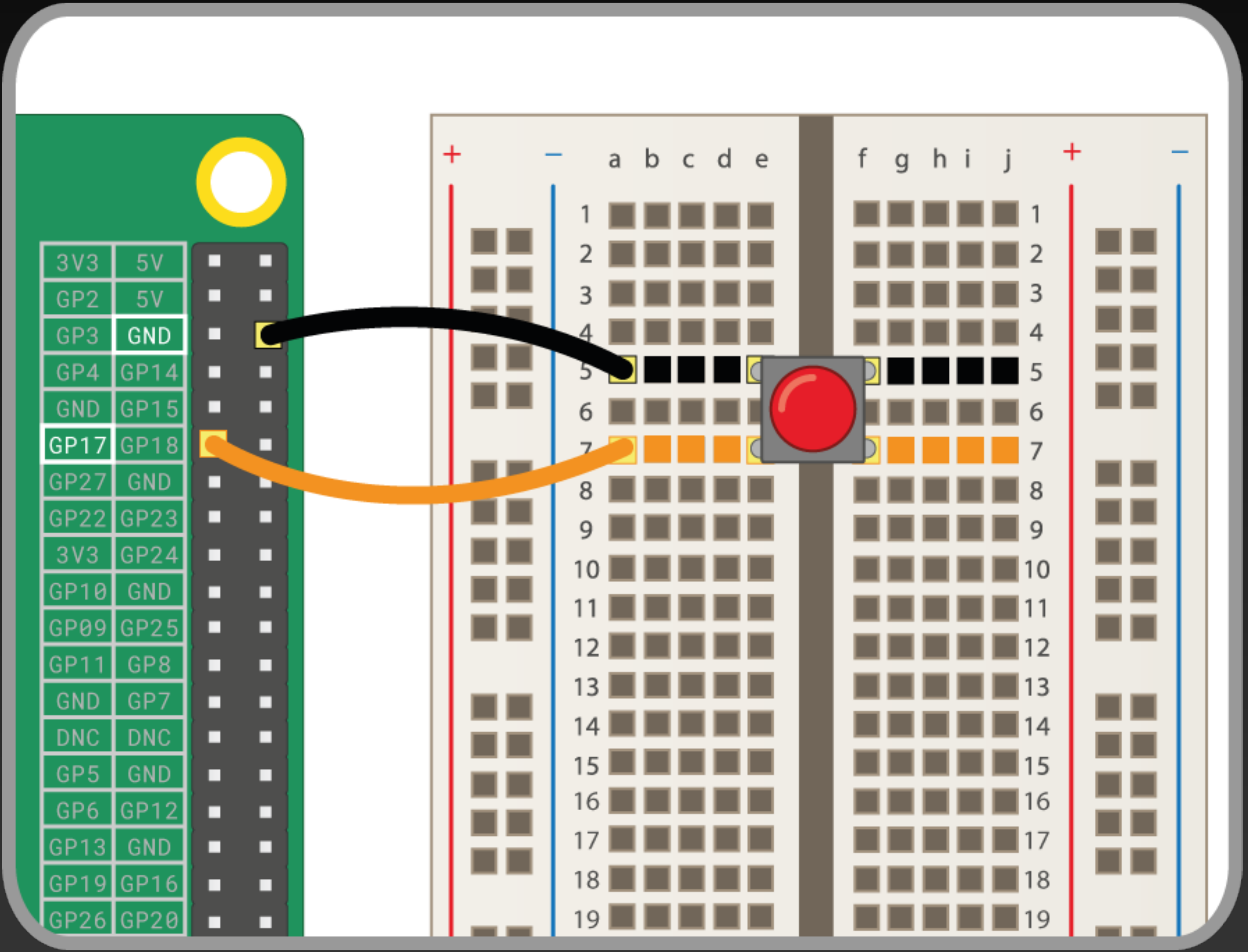
A button is one of the simplest input components you can wire to a Raspberry Pi. It’s a non-polarised component, which means you can place it in a circuit either way round and it will work.

There are various types of buttons - they can for example have two or four legs. The two-leg versions are mostly used with flying wire to connect to the control device. Buttons with four legs are generally mounted on a PCB or a breadboard.

GPIO music box
Connect your buttons

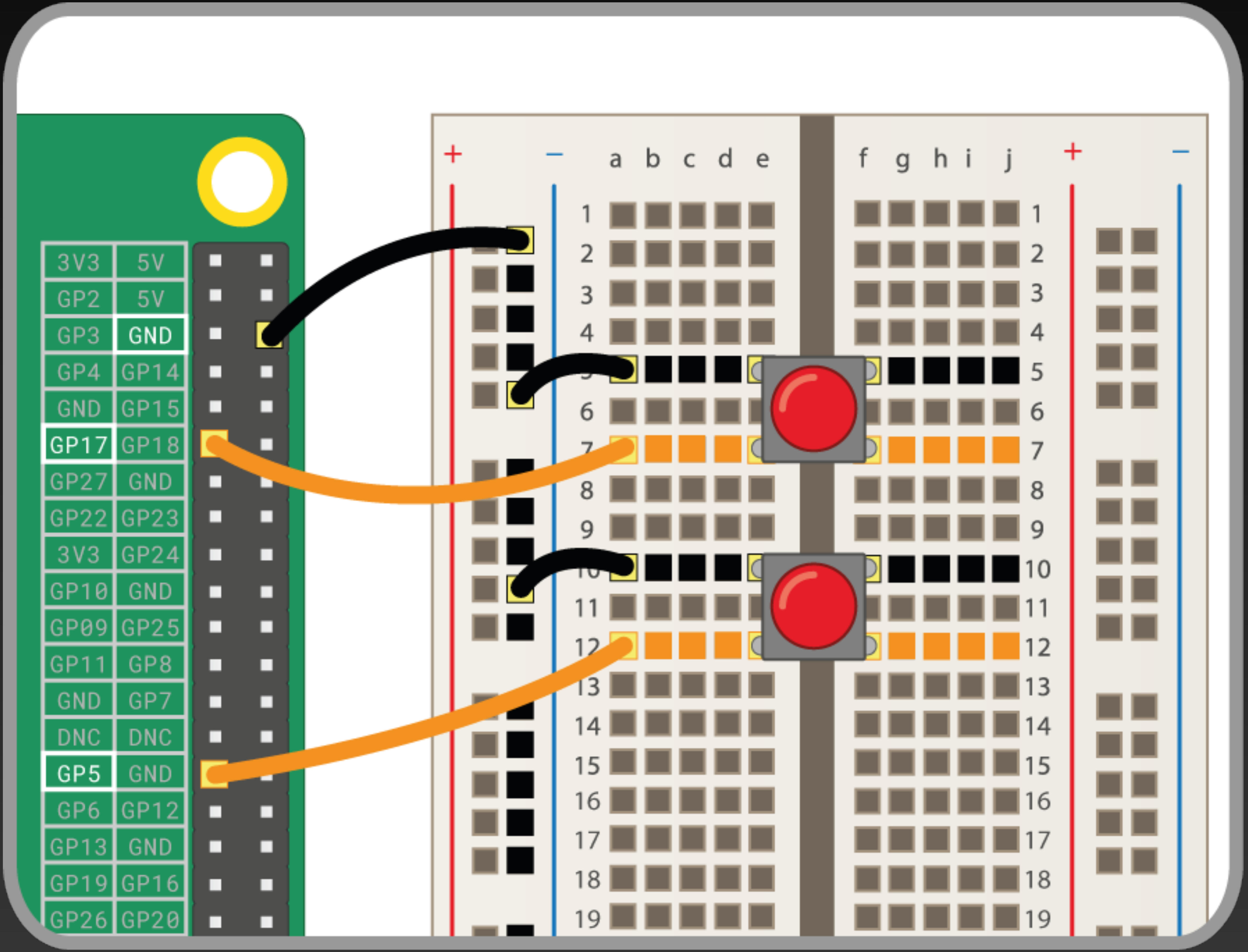
The diagrams below shows how to wire a two-leg or four-leg button to a Raspberry Pi. In both cases, GPIO 17 is the input pin





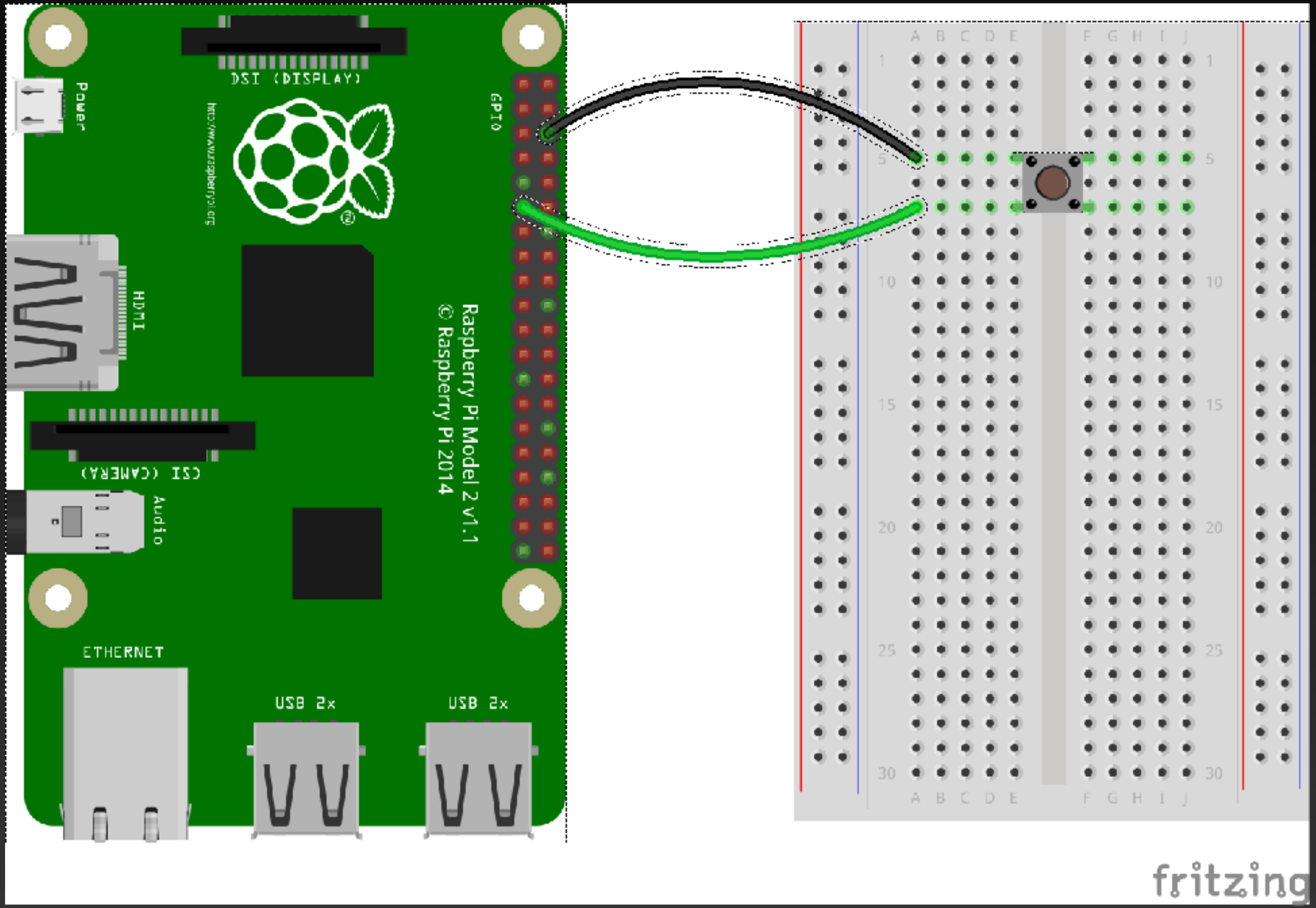
GPIO music box
Connect your buttons

If you are using multiple buttons, then it is often best to use a common ground to avoid connecting too many jumper leads to GND pins. You can wire the negative rail on the breadboard to a single ground pin, which allows all the buttons to use the same ground rail.



GPIO music box
Test the button

```
from gpiozero import Button
btn = Button(17)

def hello():
    print(‘Hello')

btn.when_pressed = hello
```

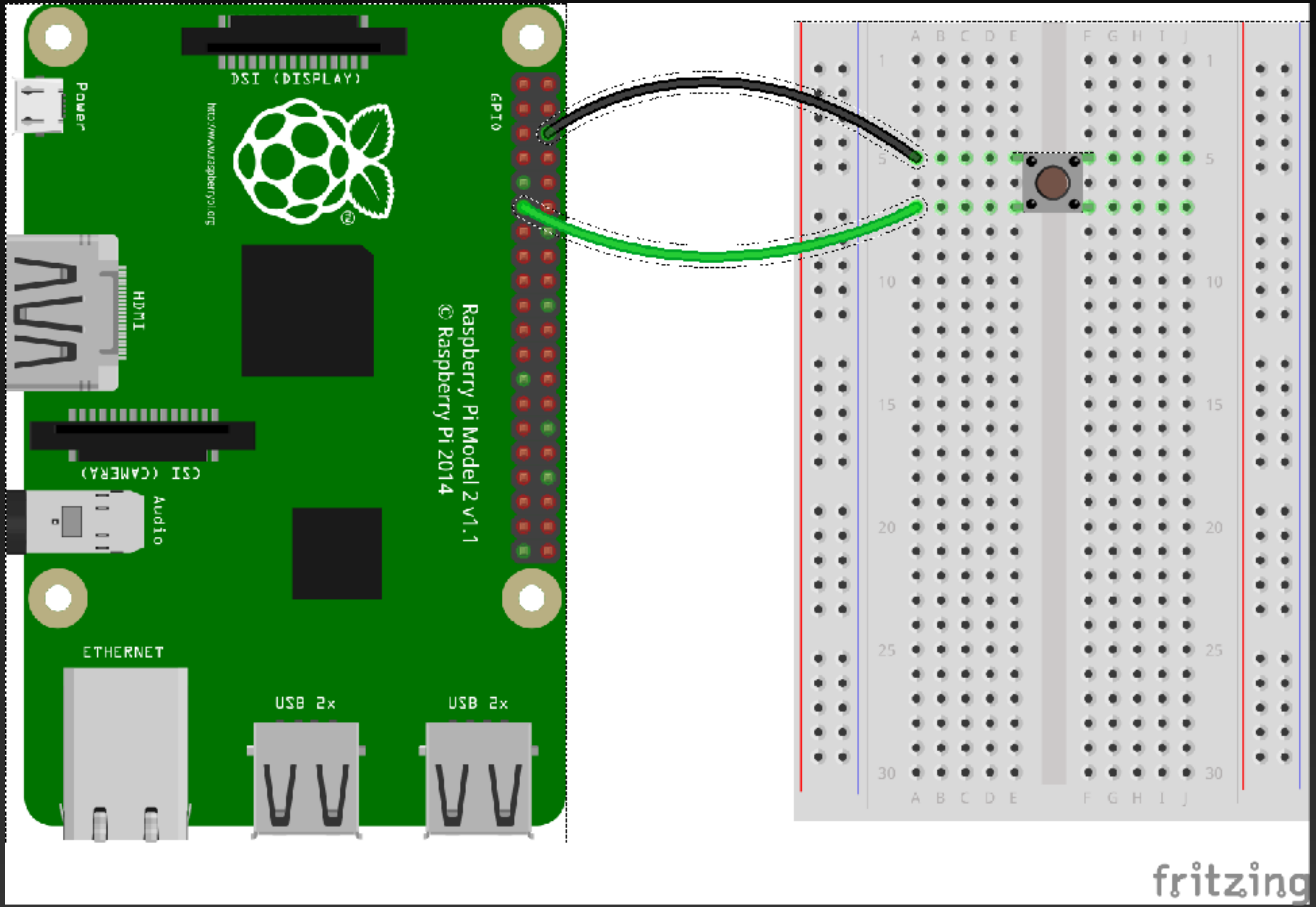


GPIO music box
Test the button

```
import pygame
from gpiozero import Button

pygame.init()

drum = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_tom_mid_hard.wav")
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-box/
samples/drum_cowbell.wav")

btn_drum = Button(17)
```

To play the sound when the button is pressed, just add this line of code to the bottom of your file:

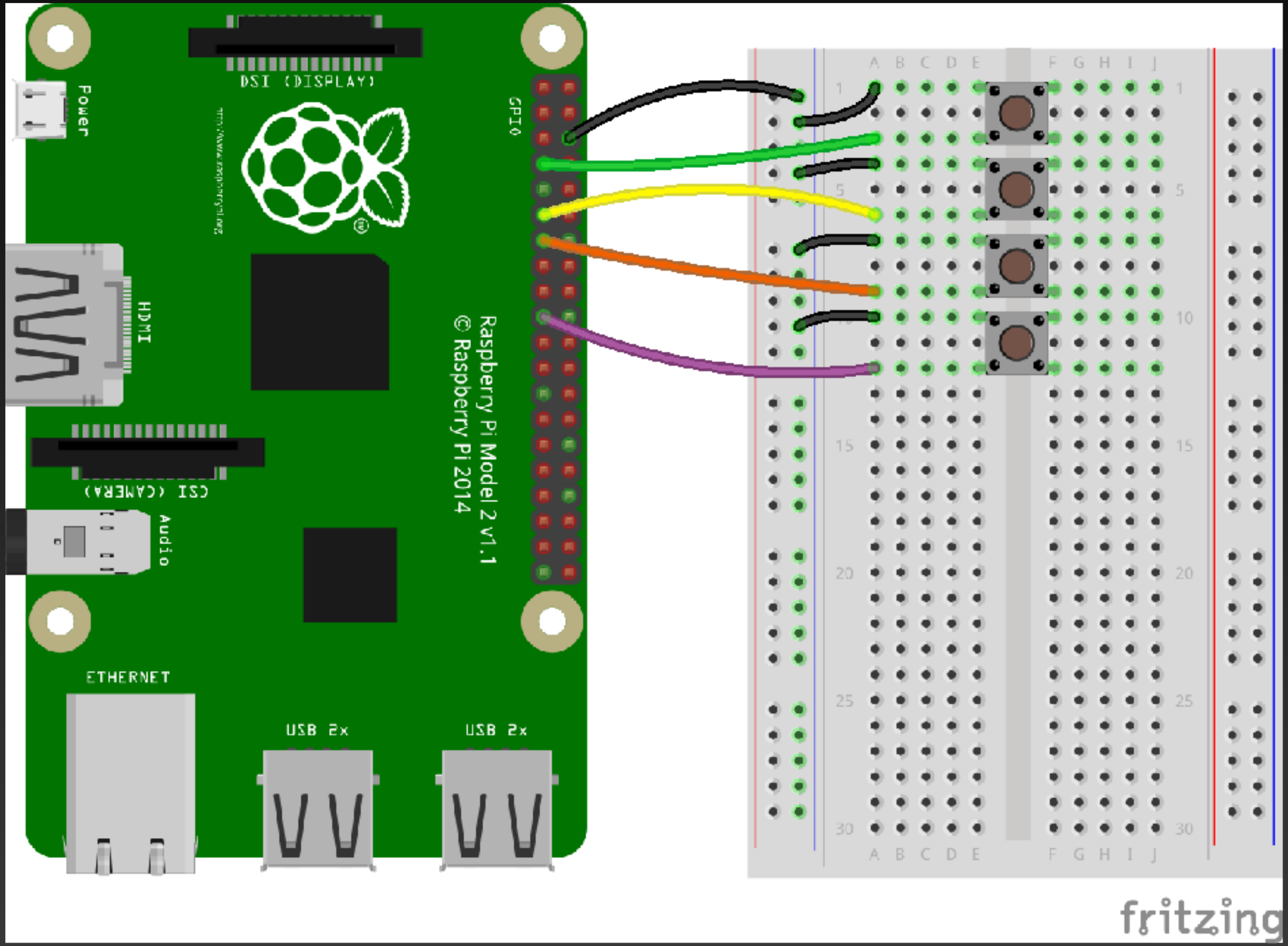
```
btn_drum.when_pressed = drum.play
```



GPIO music box
Connect your buttons

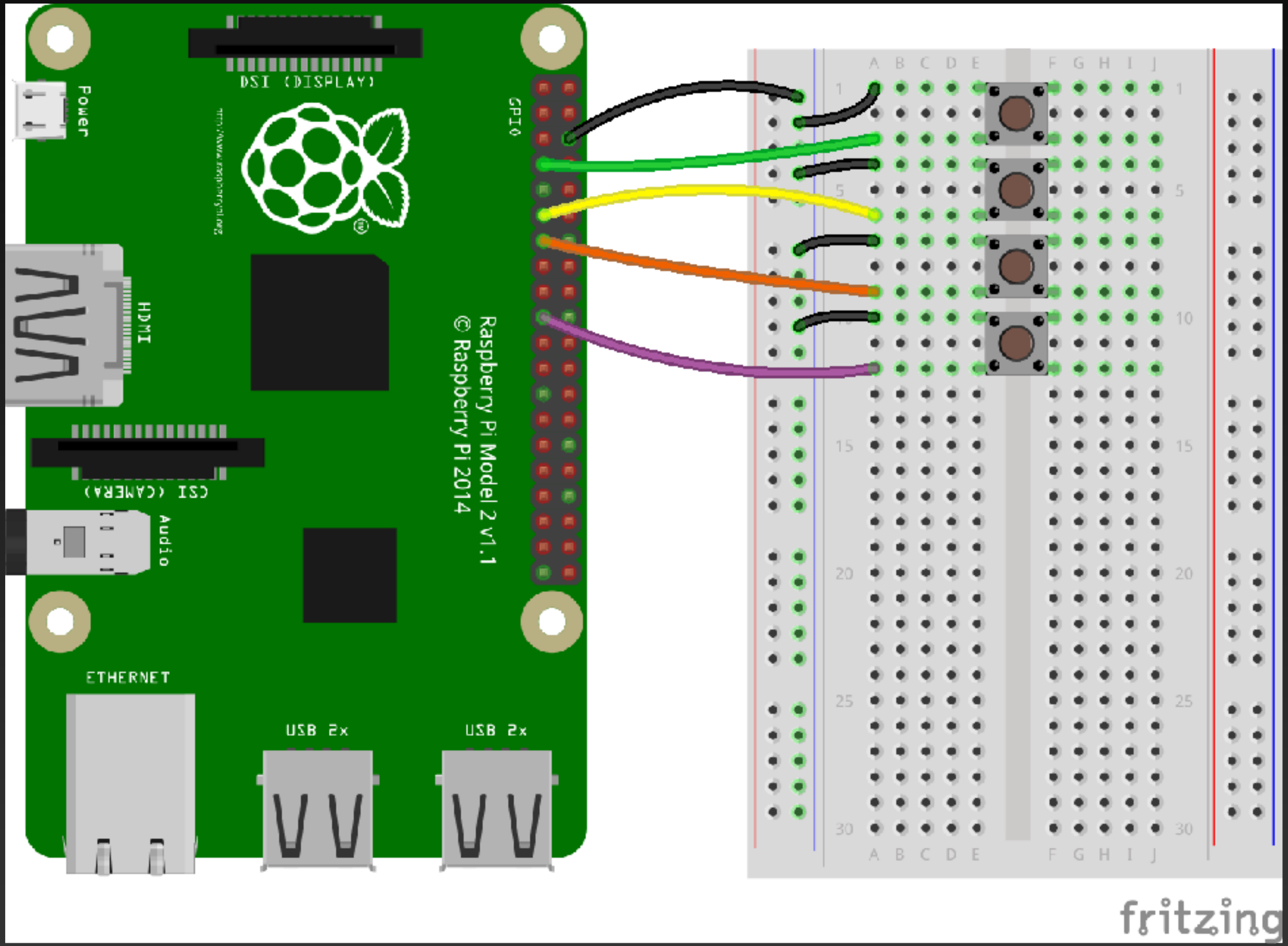
Place the four buttons into your breadboard.

Wire each button to a different numbered GPIO pin. You can choose any pins you like, but you will need to remember the numbers.



GPIO music box
Connect your buttons

```
import pygame
from gpiozero import Button

pygame.init()

drum = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_tom_mid_hard.wav")
cymbal = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_cymbal_hard.wav")
snare = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_snare_hard.wav")
bell = pygame.mixer.Sound("/home/pi/gpio-music-
box/samples/drum_cowbell.wav")

btn_drum = Button(4)
btn_cymbal = Button(17)
btn_snare= Button(27)
btn_bell = Button(10)

btn_drum.when_pressed = drum.play
btn_cymbal.when_pressed = cymbal.play
btn_snare.when_pressed = snare.play
btn_bell.when_pressed = bell.play
```



GPIO music box
Improve your code

```
import pygame
from gpiozero import Button

pygame.init()

button_sounds = {Button(4): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_tom_mid_hard.wav"),
                 Button(17): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_cymbal_hard.wav"),
                 Button(27): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_snare_hard.wav"),
                 Button(10): pygame.mixer.Sound("/home/pi/gpio-music-box/samples/
drum_cowbell.wav")}

for button, sound in button_sounds.items():
    button.when_pressed = sound.play
```

GPIO music box
Improve your code

```
Homework 1:  finish the class project with 4 buttons

Homework 2:  add led to class project and modify your code to use button to control LED
on/off
```

IoT with MIT App Inventor

Office hour(8:45pm CST)