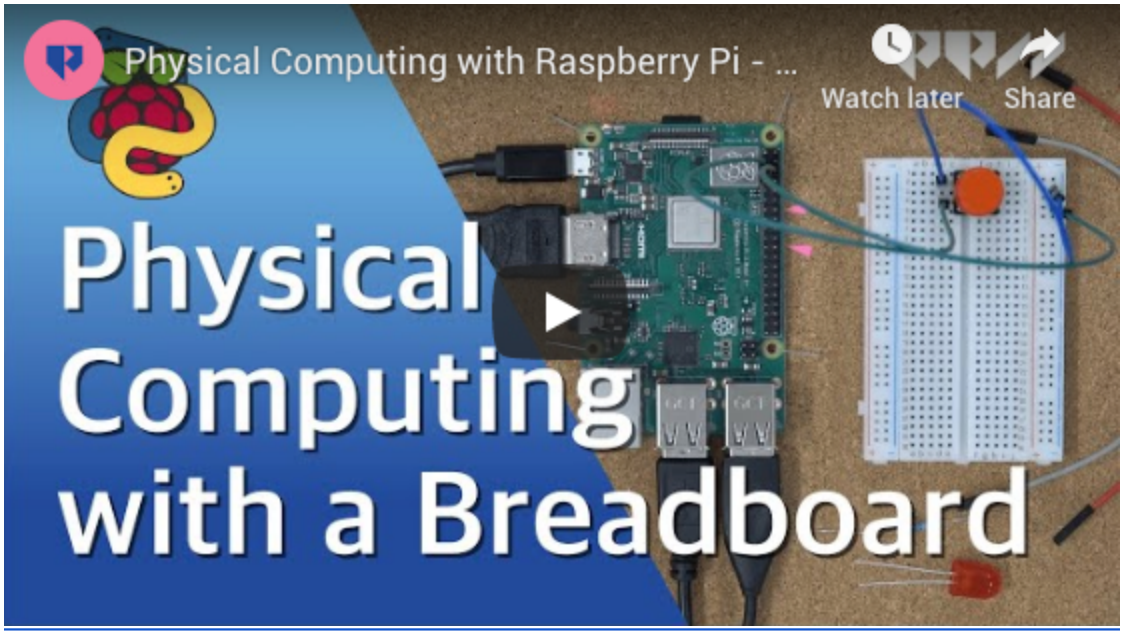
Physical Computing with a Breadboard



Think of all of the pieces of technology that require a button. That is a ton of devices. Adding a button gives you the ability to create so many different projects. We will work with adding the button to our program and then connect it to the LED so you can make the light turn on and off.

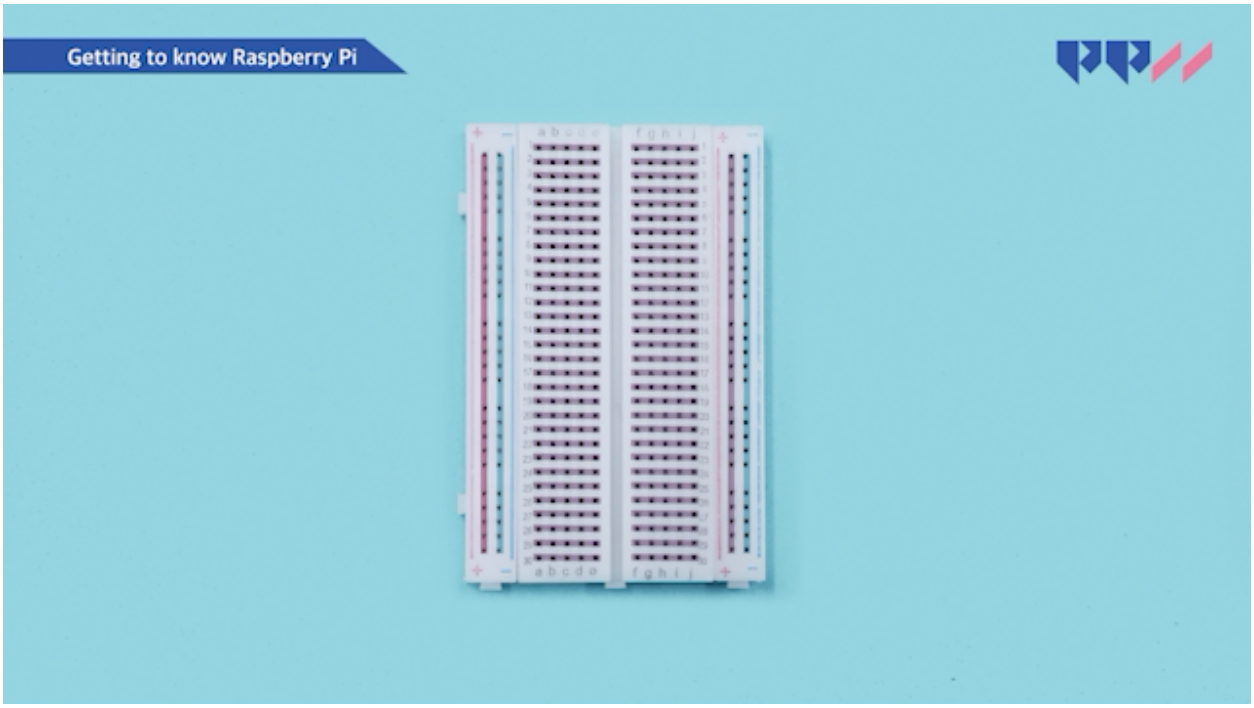
Materials

- Raspberry Pi Model 3 B+ 8GB
- Micro SD Card
- Breadboard
- Micro USB Power Supply
- Mouse
- Jumper Wires
- HDMI Cord
- Keyboard
- LEDs
- HDMI Monitor
- Computer
- Button

Key Concepts

- GPIO
- Time
- Sleep
- Button
- LED

About the breadboard

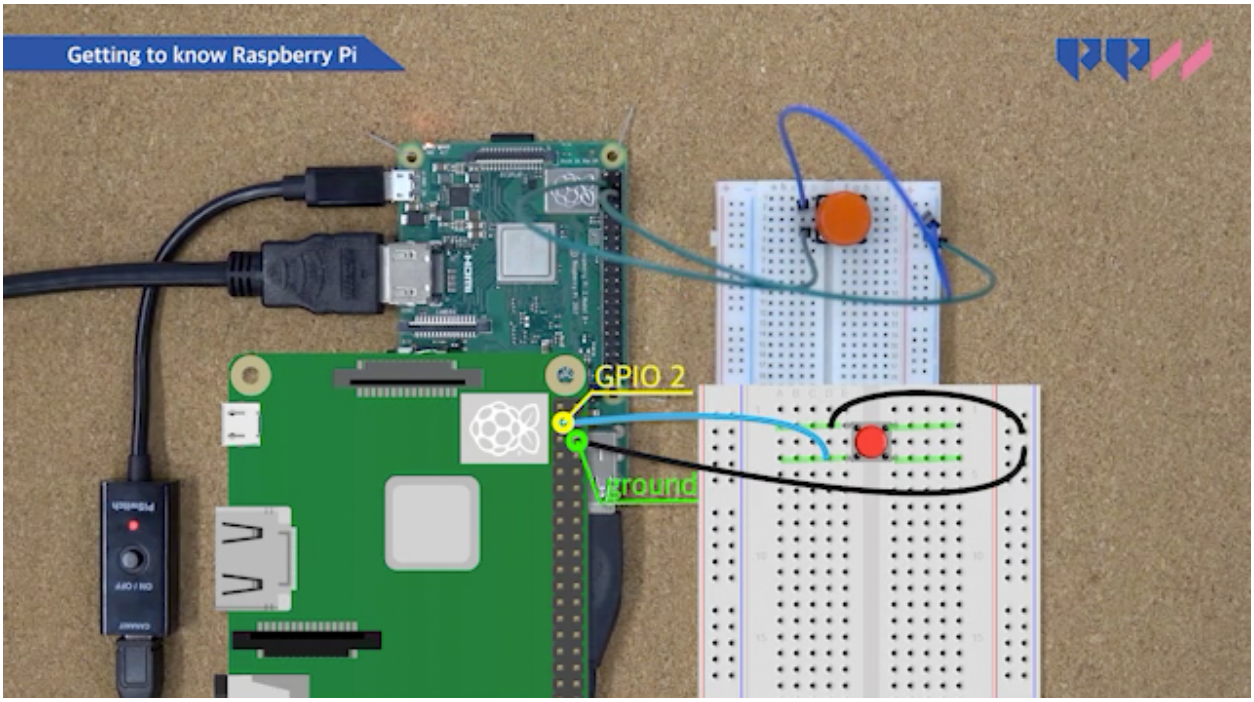


The breadboard is going to make the next projects so much easier by holding all of the components we will be using. Here is a look at what a breadboard looks like and how things are connected.

The columns in the center are all connected and the rails on the outer part of the breadboard are connected as well. Under each row and column are metal strips that act like wires to connect each hole to another. The top and bottom have rails that are designed to make it easier to share power across the entire rail and to create a ground across the entire rail as well. They are usually marked with a + or a -.

Adding components to a breadboard is easy. You line up the metal parts (leads) of the component and place each one into a hole on the breadboard. We are going to set a button on the breadboard using jumper wires. Here is a diagram of what it will look like.

The Hardware



1. First, we are going to set the button in place. We are going to place it on the breadboard across the center divide of the breadboard. Note how it looks in the image.
2. You will need a male/female jumper wire. You will attach the female end to **GPIO 2**pin and the male end will go to the hole right before the button lead.
3. You will take a female/male jumper wire and attach the female end to the ground pin on the Pi. Take the male end and stick the pin into the far rail that is marked as a negative rail.
4. Finally, take a male/male jumper wire and connect one end to the ground rail and then take the other end and place it next to the other lead of the button. Use the image to help guide your wire placement.

You are creating a circuit with the GPIO pin and ground with the button in the center. Let's write some code to use the button.

The Code

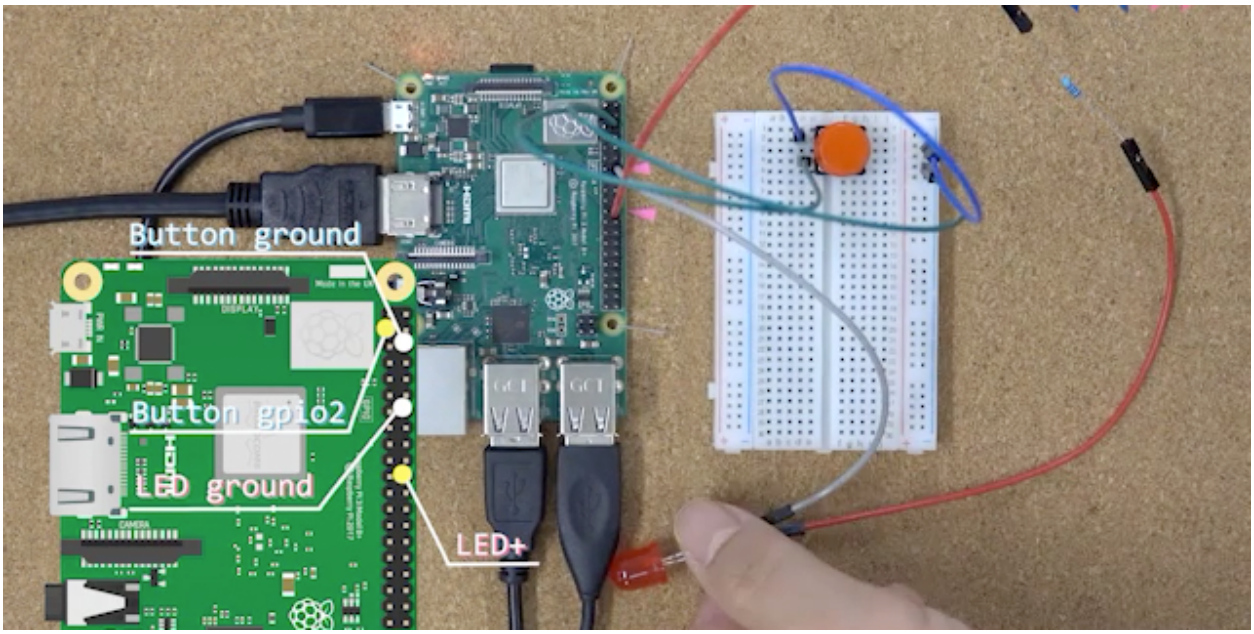
```
from gpiozero import Button

button = Button(2)
button.wait_for_press()
print("Stop pressing my Button!")
```

1. Copy the following code in a new Thonny file.
2. We will start by pulling Button from `gpiozero` and then designate button to `gpio 2`.
3. Next, we will tell the button to “wait for press”
4. Finally, we want something to happen when it is pressed, so we tell it to print.
5. If everything is wired up correctly, the shell should print what you wrote when you press the button.
6. If it does not print, then your connections might be loose. Make sure everything is plugged into where it is supposed to be and try again.

The LED Project -the hardware

Take the LED circuit you created earlier and add it to the button circuit you assembled on the breadboard. You do not need to add it to the breadboard directly, but you can if you'd like.



The LED Project -the code

You need to tweak the code to make the button turn on the led. Here is the new code,

```
from gpiozero import LED
from time import sleep
from gpiozero import Button

button = Button(2)
led = LED(25)
button.wait_for_press()
led.on()
sleep(3)
led.off()
```

Notice that we added time and sleep at the top, connected led to `LED(25)`, added `led.on`and `led.off`, and finally added the sleep command to cause it to blink. Add this to the code and run it. Push the button and watch it blink.

The Challenge

- Can you use a forever loop to make the light continue to blink after you push it?
- Can you make the button turn the light on with one push and then off with another?

  Check the GPIO library for commands to help you solve these problems.