

Part 2

Scratch: LED + Button

Adapted with permission from Raspberry Pi Picademy

Goals

- Why physical computing?
- Build a simple circuit using Raspberry Pi
- Control inputs & outputs = microcontroller
- Create simple or complex circuits





Physical Computing

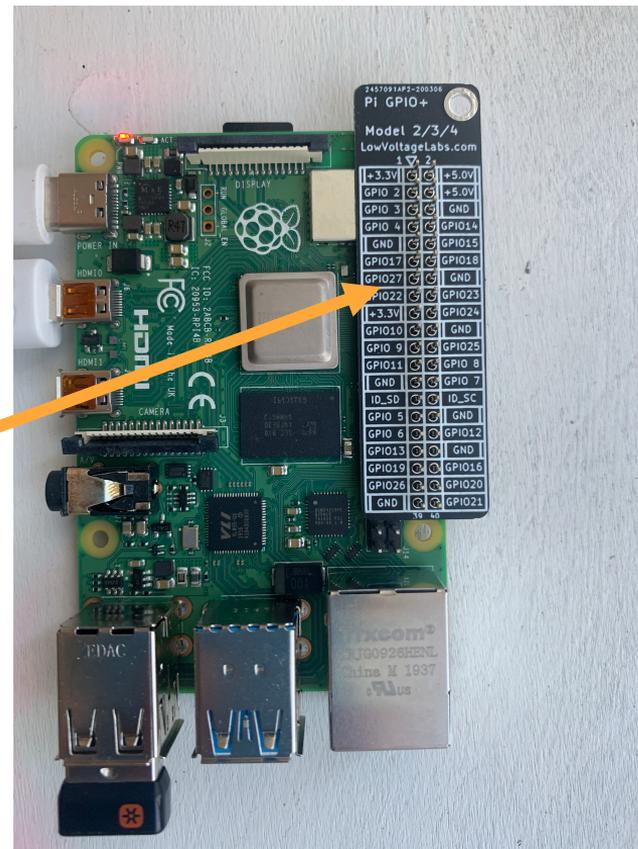
- Bring your code to life
- Connect the digital and physical world
- Engage in hands-on making and creating



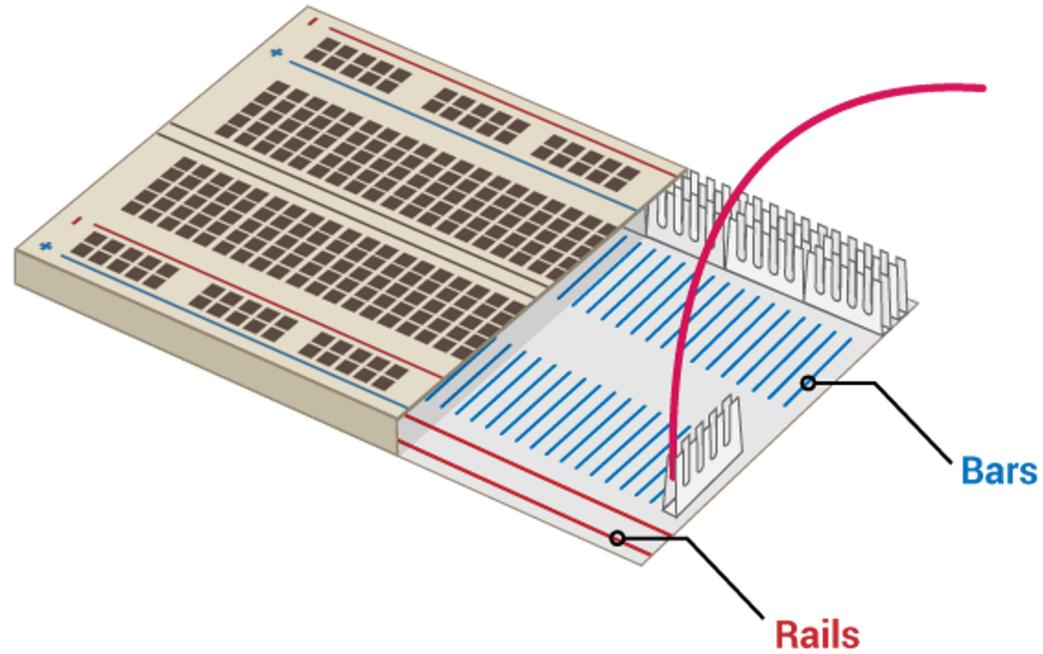


GPIO Reference Board

- Identify GND
- Identify 3.3V and 5.0 V
- Identify GPIO 18
- Identify GPIO 14
- Align and place over the two 20-pin rows on the Pi.

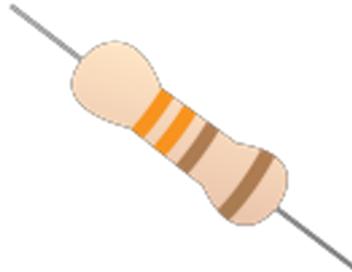
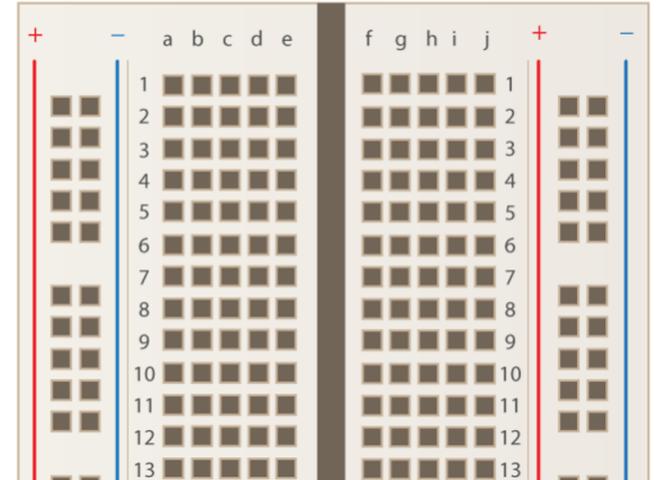


Inside a breadboard



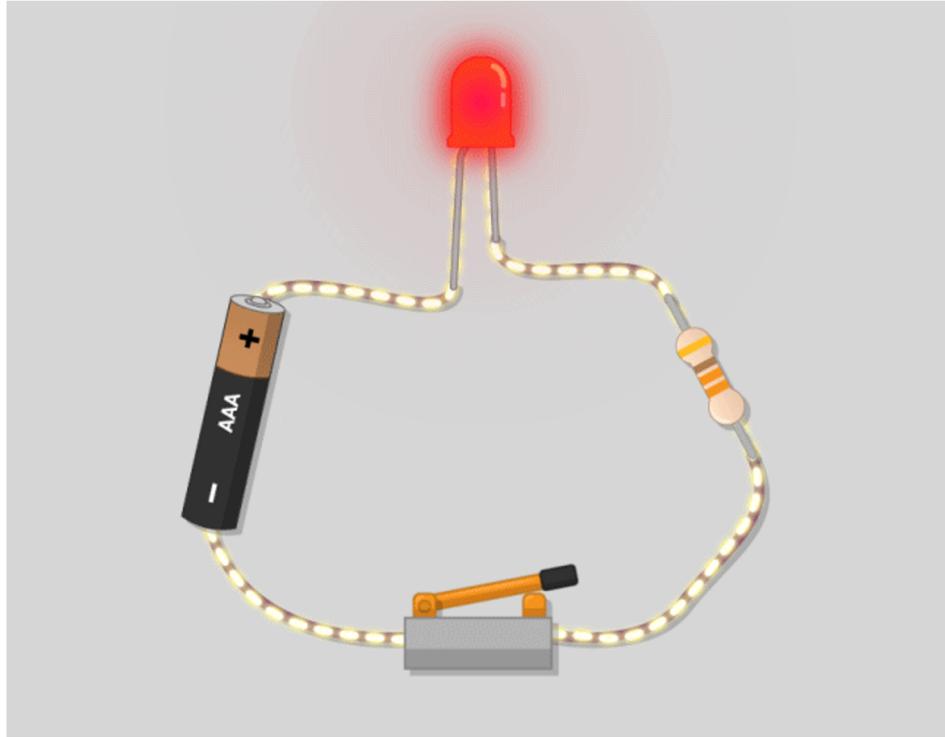
Physical Computing

Enables creators to connect the virtual world with the physical, great for engagement.



Simple Circuit

Your Raspberry Pi can act as the power supply for simple circuits, we can use this test our hardware is working.





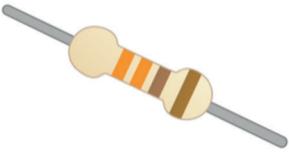
Materials

You will need:

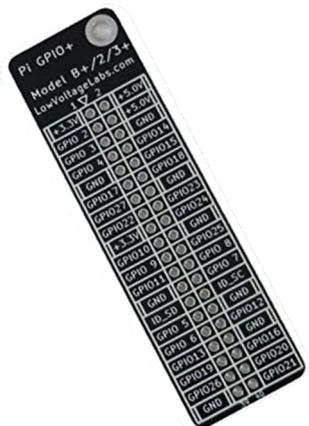
LED



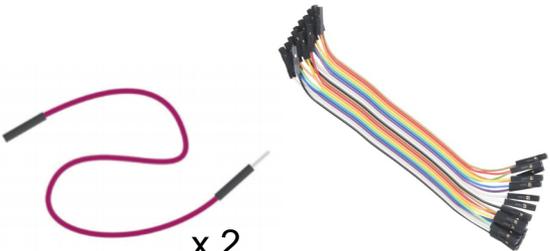
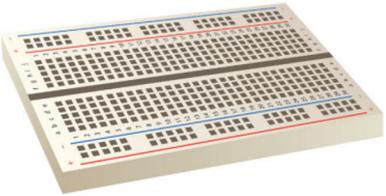
Resistor (200-500 ohms)



GPIO Reference Board



Breadboard

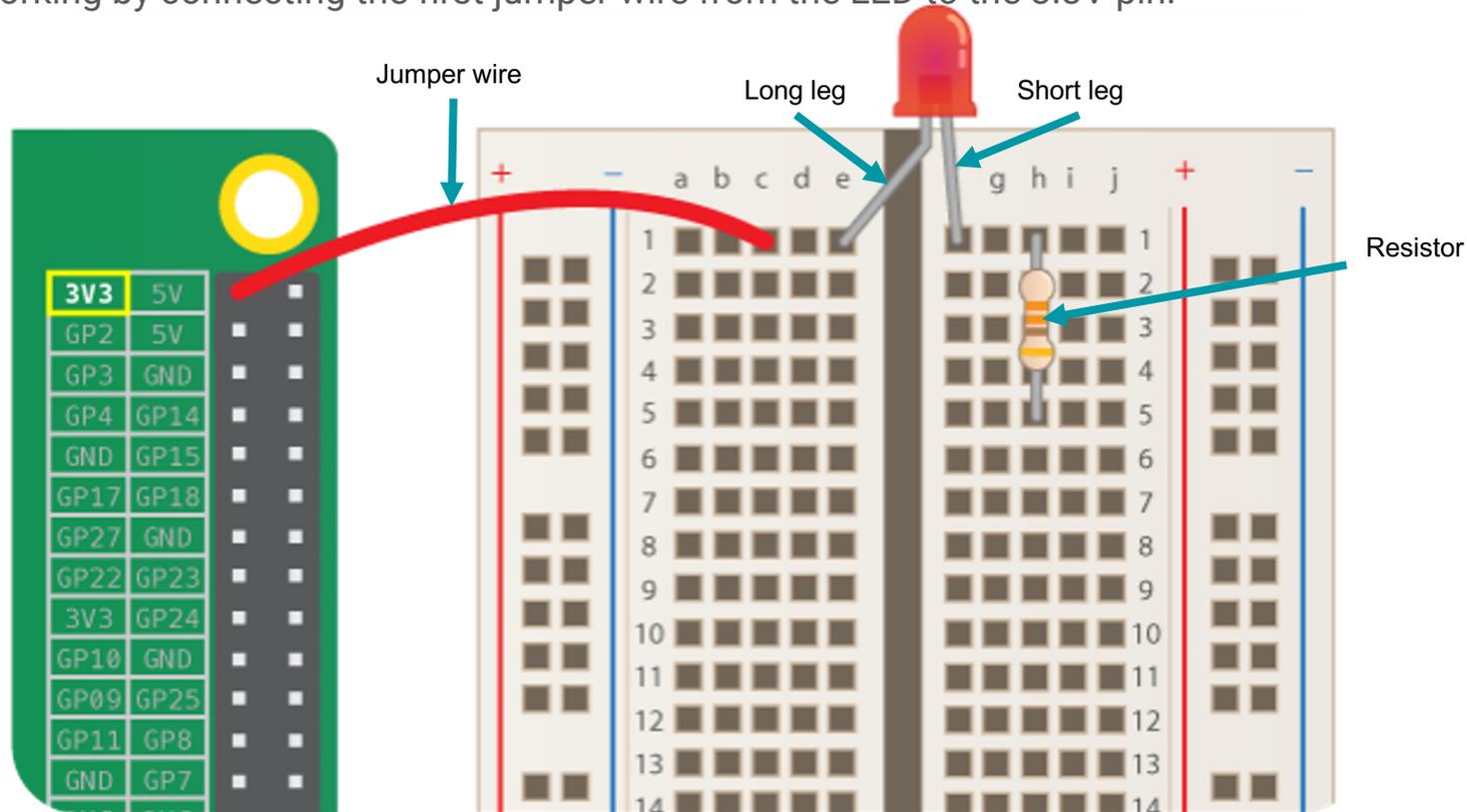


x 2

Male-to-Female Jumper Wires

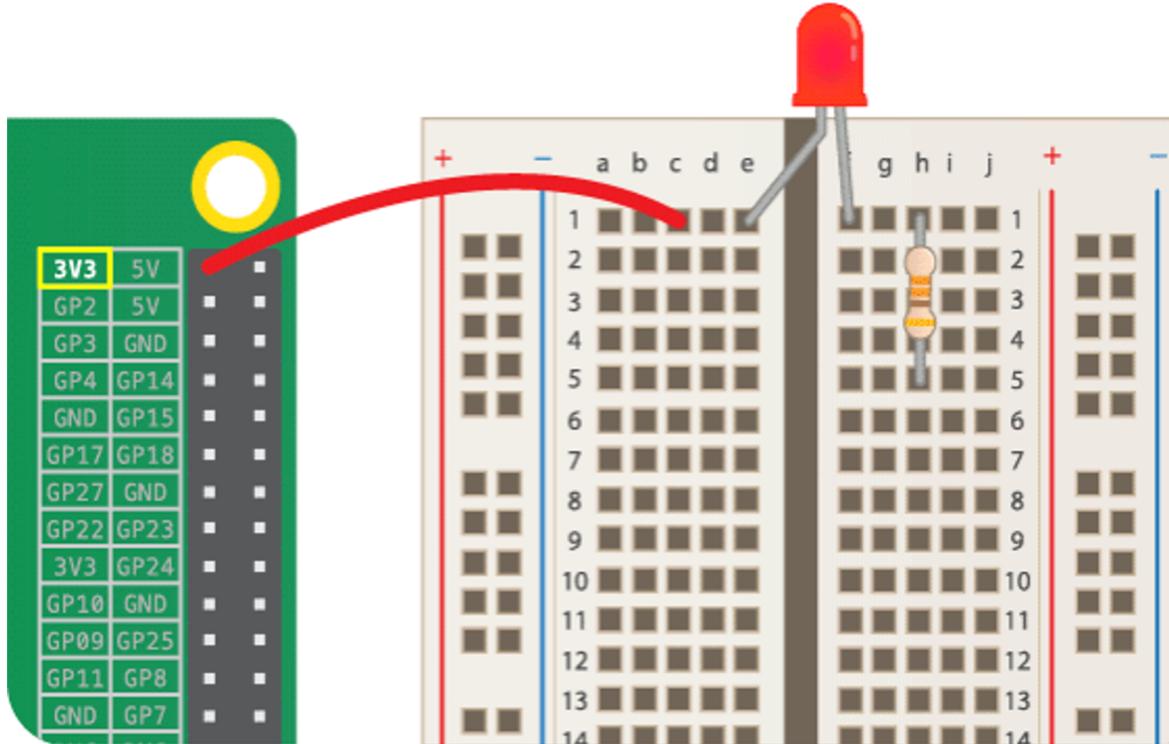
Simple Circuit

Your Raspberry Pi can act as the power supply for simple circuits, we can use this to test if our hardware is working by connecting the first jumper wire from the LED to the 3.3V pin.



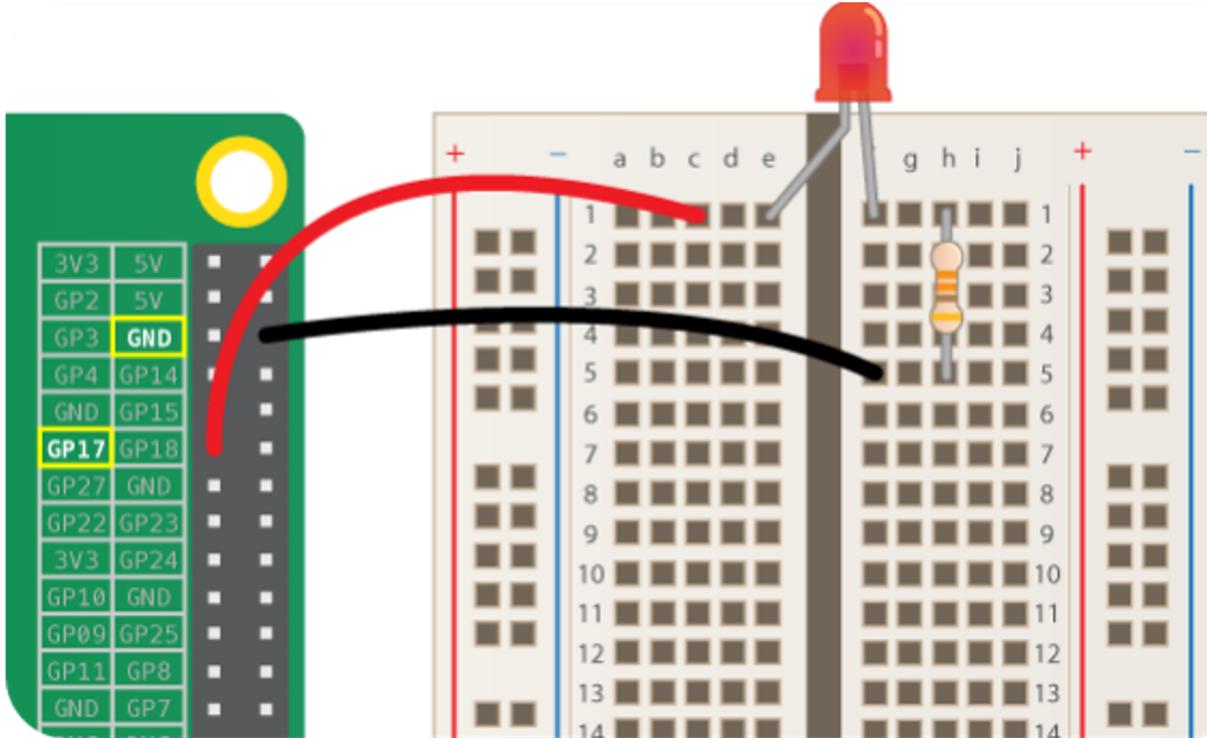
Simple Circuit

Next, connect the other jumper wire from the resistor to any of the ground pins labelled GND. The LED should turn on and stay on. If it doesn't, try checking to see if you've reversed the long and short legs of the LED.



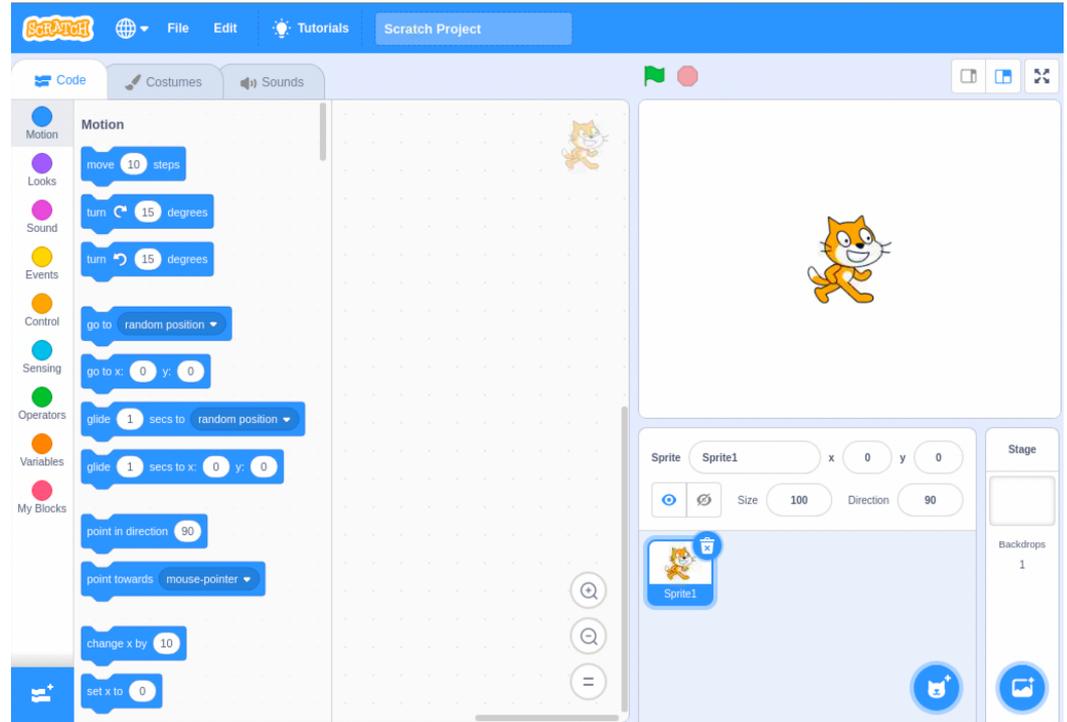
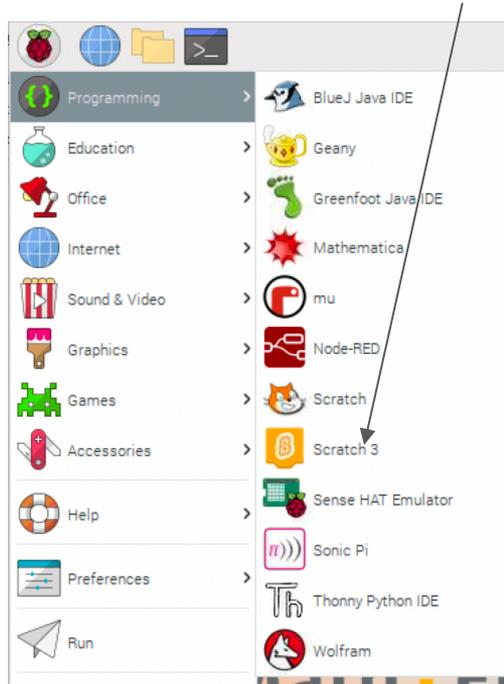
Switching Circuit

Your Raspberry Pi can act as a switch in the circuit, which can be controlled in software. Connect the positive leg of the LED to any other GPIO pin on your Raspberry Pi. In the example below, the LED is now connected to GPIO #17.



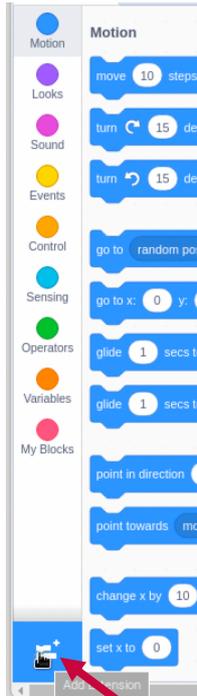
Scratch on the Raspberry Pi

Open Scratch 3.

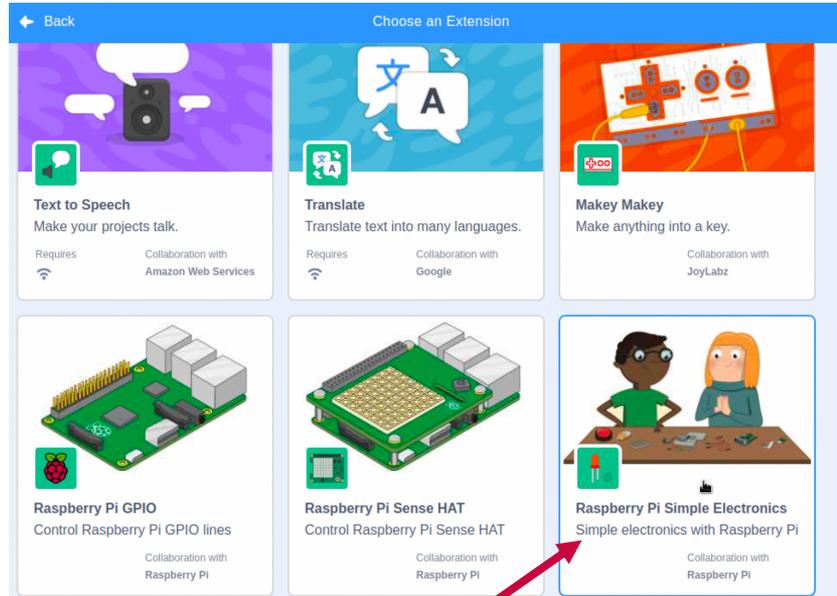


Scratch on the Raspberry Pi

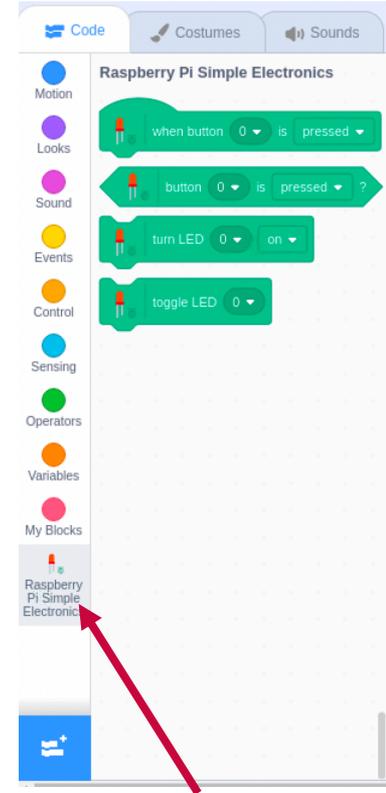
Add Extension & Choose "Simple Electronics":



1



2



Activated!

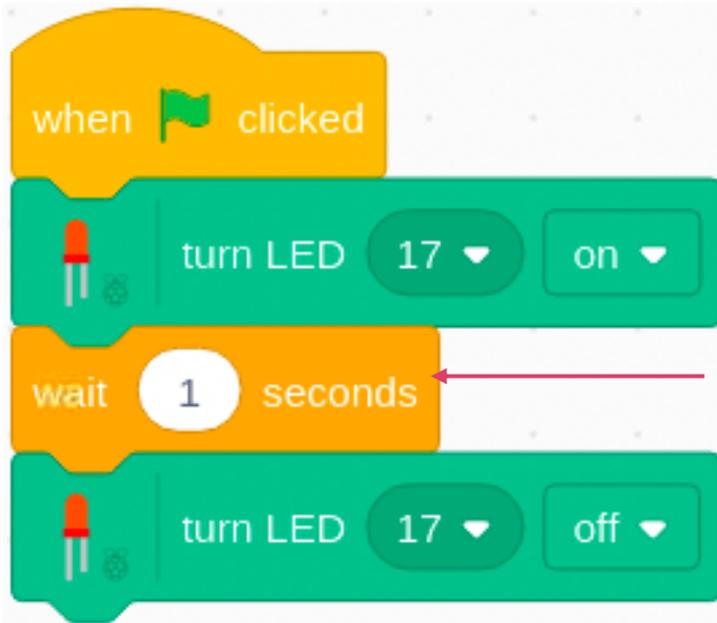


Turn LED on and off- try with these blocks

Starts your code		When green flag clicked
Turn LED on/off		Choose GPIO # of your LED & choose "on" or "off"
Pause		Pauses for the given amount of time



Turn LED on and off - SOLUTION



Why is this "wait" block necessary?

Extra Challenge!

- Can you make a dot (short flash) and dash (long flash) and use to make a distress beacon. S(...) O(---) S(...)

Save your program by clicking on
File → Save to your computer



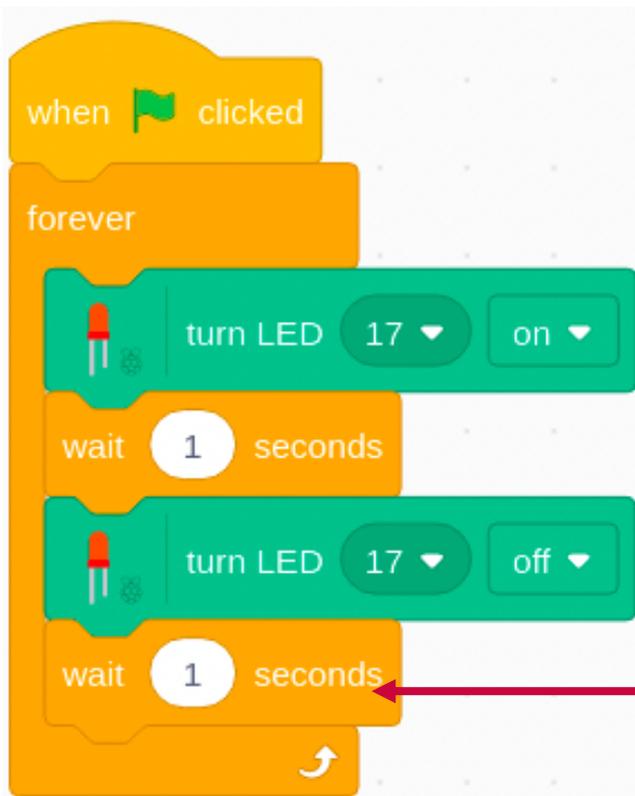
Blink LED - try with these blocks

Save a new version of the previous program, or start a new one by clicking **File → New**

Starts your code		When green flag clicked
Turn LED on		Sets pin 17 to ON = HIGH
Turn LED off		Sets pin 17 to OFF = LOW
Pause		Under 'Control' palette of blocks
Loop forever		



Blink LED - SOLUTION



Extra Challenge!

Can you:

- Flash your LED at different speeds, how fast can you make it flash?

Why is this 'wait' block necessary? What happens if it's not included?

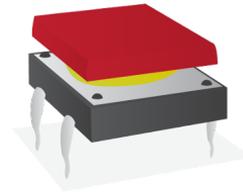
Save your program by clicking on
[File](#) → [Save to your computer](#)



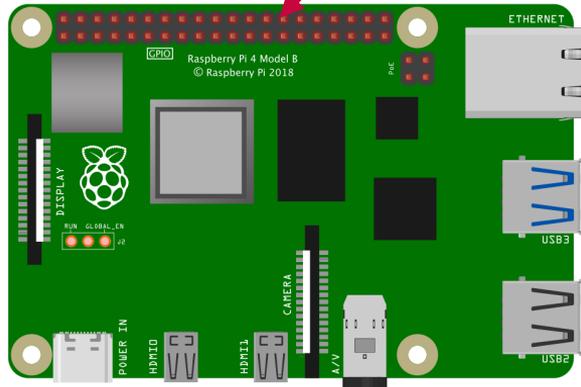
What is GPIO?

G = General P = Purpose I = Input O = Output

The Raspberry Pi is an example of a micro-controller, or a device that can take read signals as an **INPUT** or send signals as an **OUTPUT**. It does this via the 2 rows of pins (40 total) that are called GPIO pins.

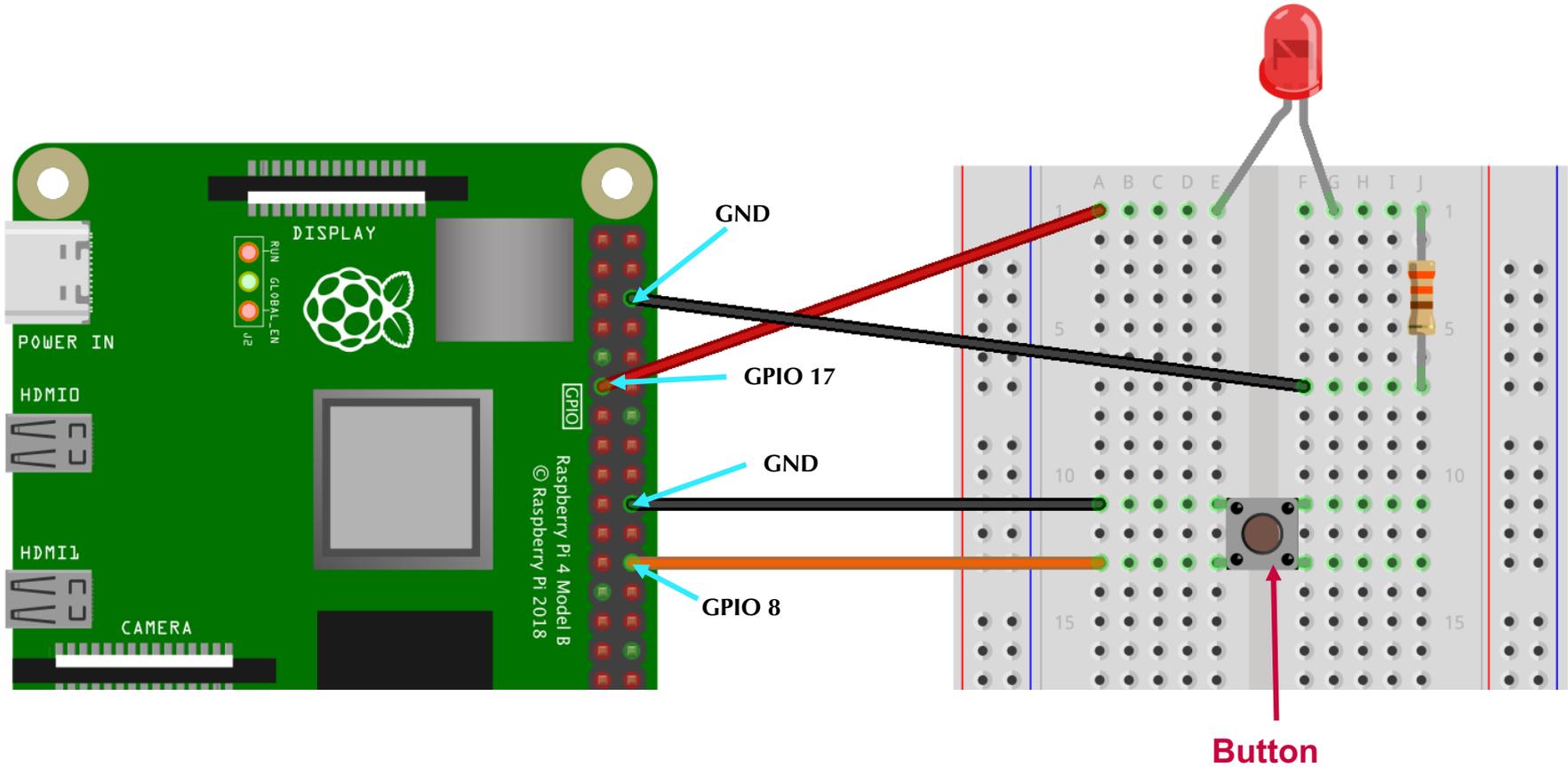


A push button is an example of an **INPUT**. The Pi reads the signal of the button through the GPIO pin to determine if it's being pressed or not.

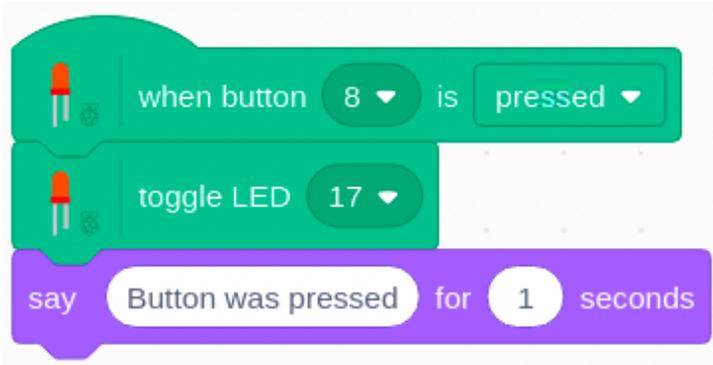


A LED is an example of an **OUTPUT**. The Pi sends a signal to the LED through the GPIO pin to make it turn on and off.

Add a GPIO Button



Let's test the Button first:



This program waits until the button at GPIO8 is pressed. Then it toggles the LED between on and off and makes Scratch the Cat say “Button was pressed” for 1 second.

If the button is working, you can move on to the next challenge. If not, check that the button and LED are connected to the proper GPIO pins.

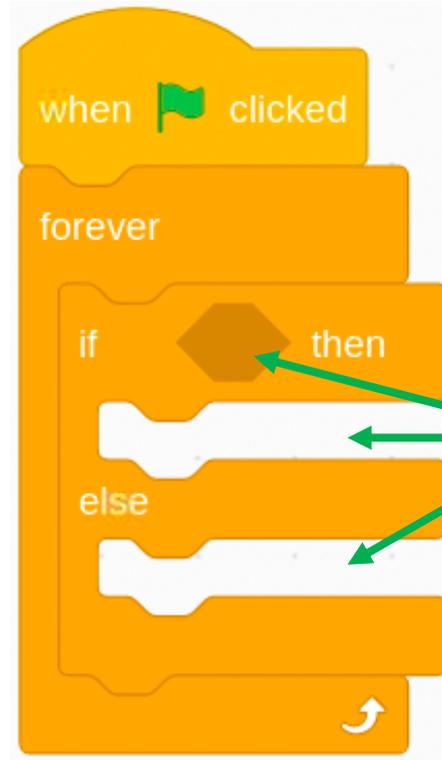


Make the LED react to the button:

CHALLENGE:

Now that we've added the button to our circuit, let's write a program that turns the LED on when the button is pressed and turns it off otherwise.

Here are the starter blocks.
Can you fill in the rest?

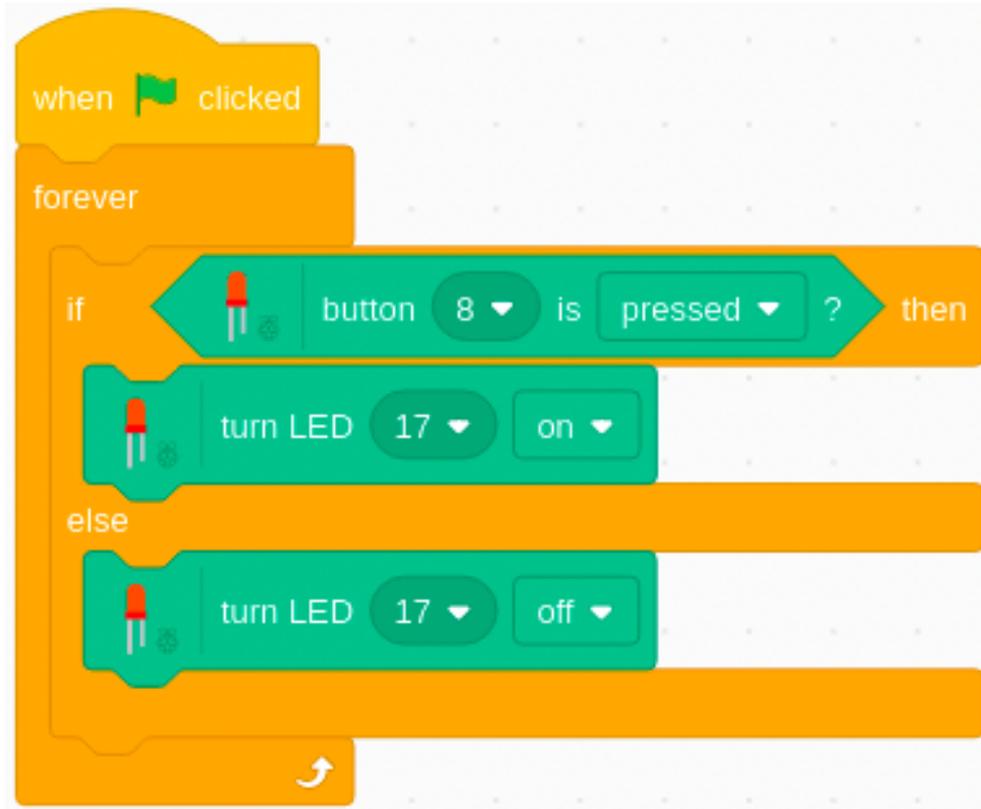


Save a new version of the previous program, or start a new one by clicking File → New

Which blocks go in these spaces?



Make the LED react to the button: SOLUTION

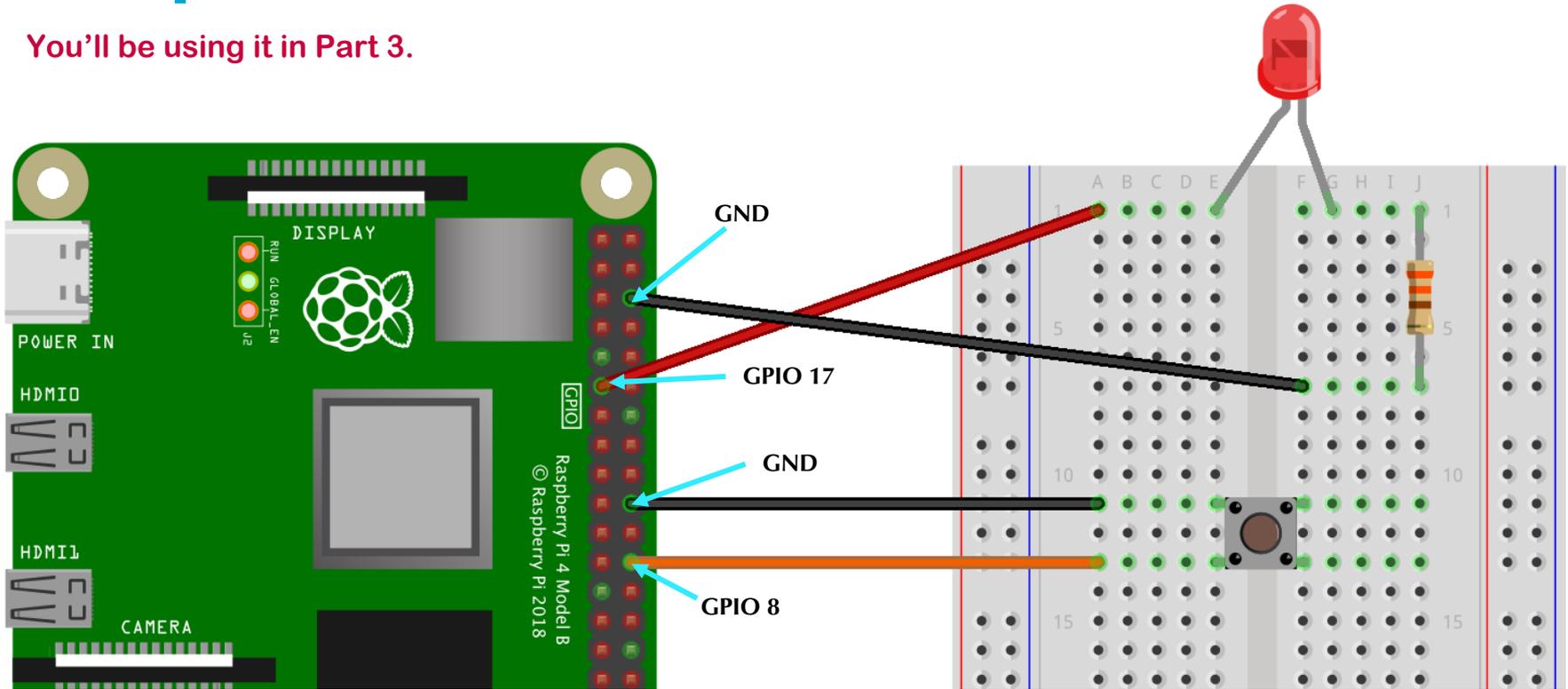


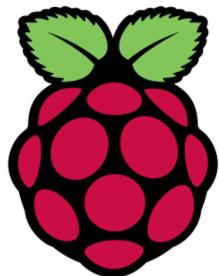
Save your program by clicking on
File → Save to your computer



Keep this circuit:

You'll be using it in Part 3.





END

PART 2