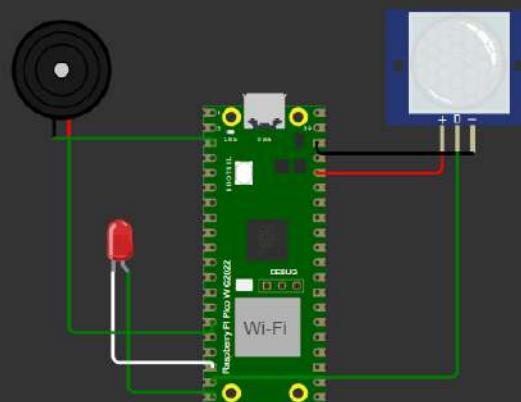


Simulation



⌚ 01:02.576 📺 100%



System reset. Monitoring again...

ALERT: Motion detected!

System reset. Monitoring again...

ALERT: Motion detected!

System reset. Monitoring again...

ALERT: Motion detected!

System reset. Monitoring again...



Simulation

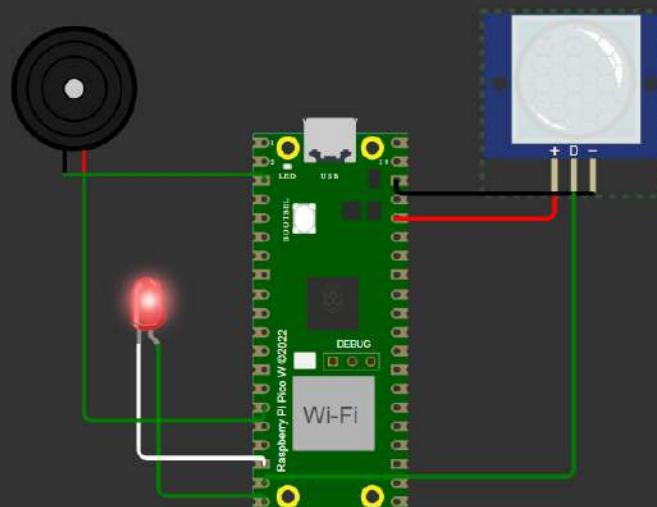


⌚ 01:57.497 🌐 100%

PIR Motion Sensor

Simulate motion

X



```
ALERT: Motion detected!
System reset. Monitoring again...
ALERT: Motion detected!
System reset. Monitoring again...
ALERT: Motion detected!
System reset. Monitoring again...
ALERT: Motion detected!
```



main.c diagram.json

```
1 #include <stdio.h>
2 #include "pico/stdlib.h"
3
4
5 #define LED_PIN 15
6 #define BUTTON_PIN 14
7
8 int main() {
9     stdio_init_all();
10
11     gpio_init(LED_PIN);
12     gpio_set_dir(LED_PIN, GPIO_OUT);
13
14     gpio_init(BUTTON_PIN);
15     gpio_set_dir(BUTTON_PIN, GPIO_IN);
16     gpio_pull_up(BUTTON_PIN);
17
18     while (true) {
19         if (gpio_get(BUTTON_PIN) == 0) {
20             gpio_put(LED_PIN, 1); // LED ON when pressed
21         } else {
22             gpio_put(LED_PIN, 0); // LED OFF when released
23         }
24         sleep_ms(10); // small delay
25     }
26 }
```

Simulation 00:28.221 99%

The screenshot shows a software interface for simulating a Raspberry Pi setup. On the left, there is a code editor with the file 'main.c' containing C code for controlling a LED via a push-button input. On the right, there is a simulation window titled 'Simulation'. The simulation shows a digital circuit. A green wire connects the 'GND' pin of a Raspberry Pi model B+ to the common ground rail. Another green wire connects the '5V' pin of the Raspberry Pi to the positive rail. A black wire connects the 'GPIO 14' pin of the Raspberry Pi to a green rectangular component representing a push-button. A red wire connects the other terminal of the push-button to the common ground rail. A third red wire connects the 'GPIO 15' pin of the Raspberry Pi to a red rectangular component representing an LED. A fourth red wire connects the anode of the LED to the positive rail. The simulation interface includes a toolbar with icons for play, stop, and step, and a status bar at the top right indicating the time (00:28.221) and battery level (99%).

WORKSPACE

main.c

diagram.json

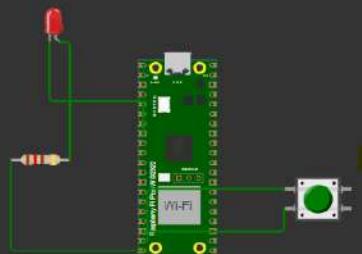
Simulation

01:08.121 100%

```
1 #include <stdio.h>
2 #include "pico/stdlib.h"
3
4
5 #define LED_PIN 15
6 #define BUTTON_PIN 14
7
8 int main() {
9     stdio_init_all();
10
11    gpio_init(LED_PIN);
12    gpio_set_dir(LED_PIN, GPIO_OUT);
13
14    gpio_init(BUTTON_PIN);
15    gpio_set_dir(BUTTON_PIN, GPIO_IN);
16    gpio_pull_up(BUTTON_PIN);
17
18    while (true) {
19        if (gpio_get(BUTTON_PIN) == 0) {
20            gpio_put(LED_PIN, 1); // LED ON when pressed
21        } else {
22            gpio_put(LED_PIN, 0); // LED OFF when released
23        }
24        sleep_ms(10); // small delay
25    }
26 }
```

Diagram showing a simulation of a Raspberry Pi Pico connected to a breadboard. The breadboard has a red LED connected to pin 15 (labeled LED) and a green pushbutton connected to pin 14 (labeled BUTTON). The simulation shows the LED being turned on when the button is pressed and off when it is released.

Simulation



Your reaction time: 610 ms

Get ready...

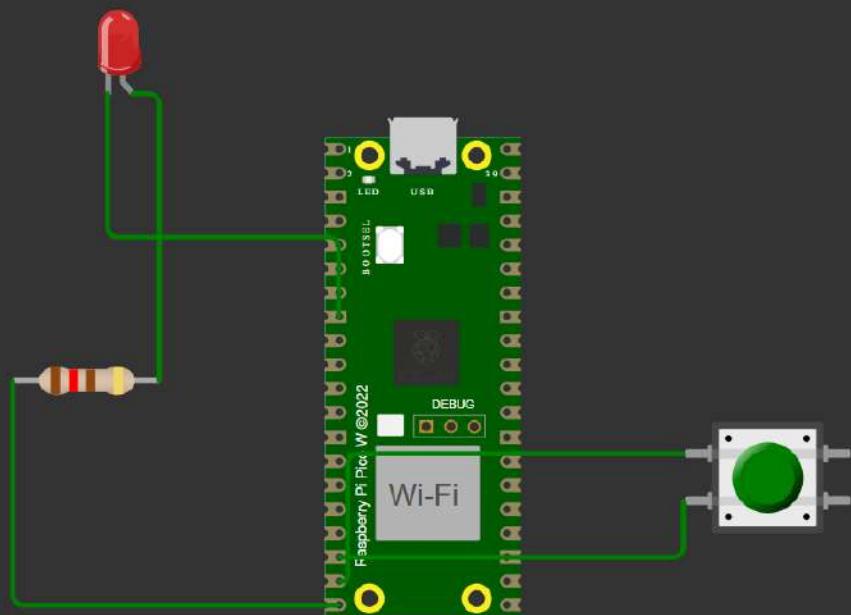
Wait for the LED to turn off!

GO!

Your reaction time: 189 ms

Get ready...

Simulation



Get ready...

Wait for the LED to turn off!

GO!

Your reaction time: 610 ms

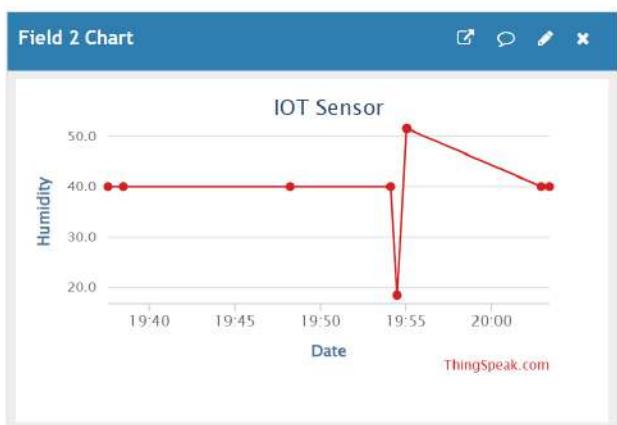
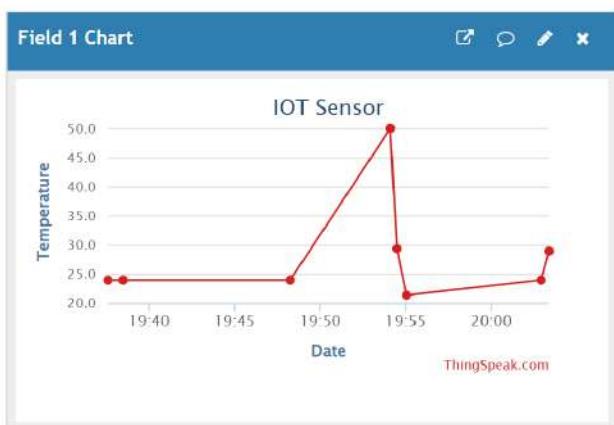


Channel Stats

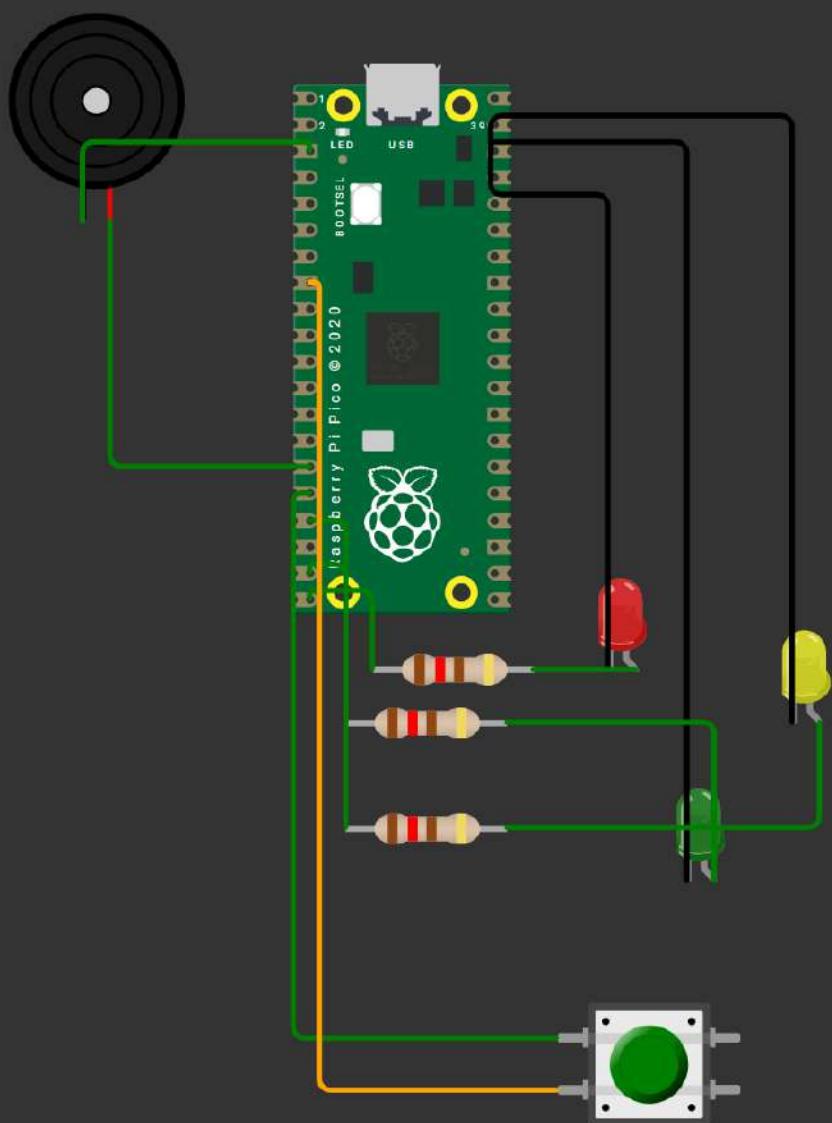
Created: 29 minutes ago

Last entry: less than a minute ago

Entries: 8



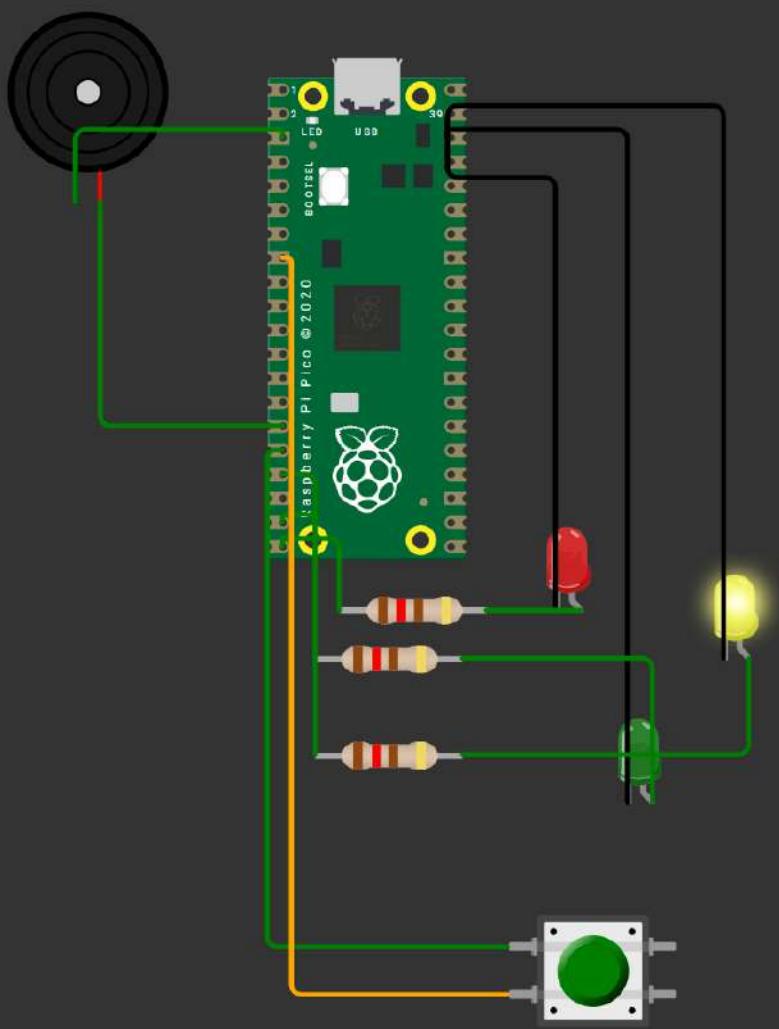
Simulation



Simulation



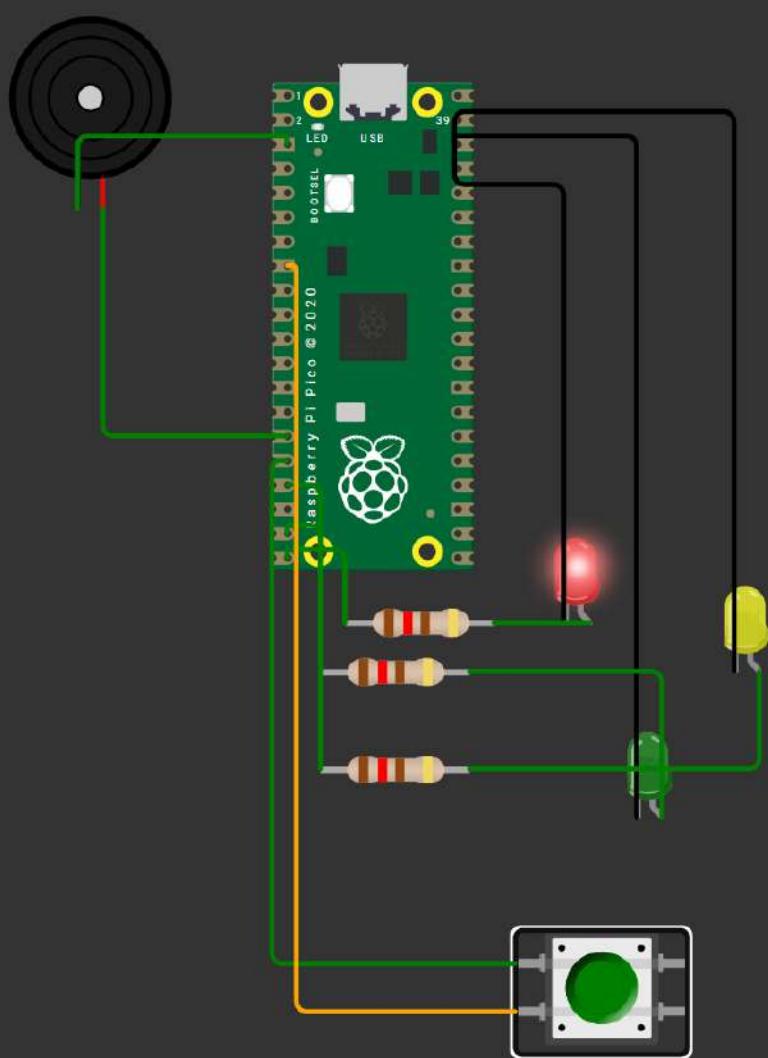
⌚ 00:07.388 🌟 100%



Simulation



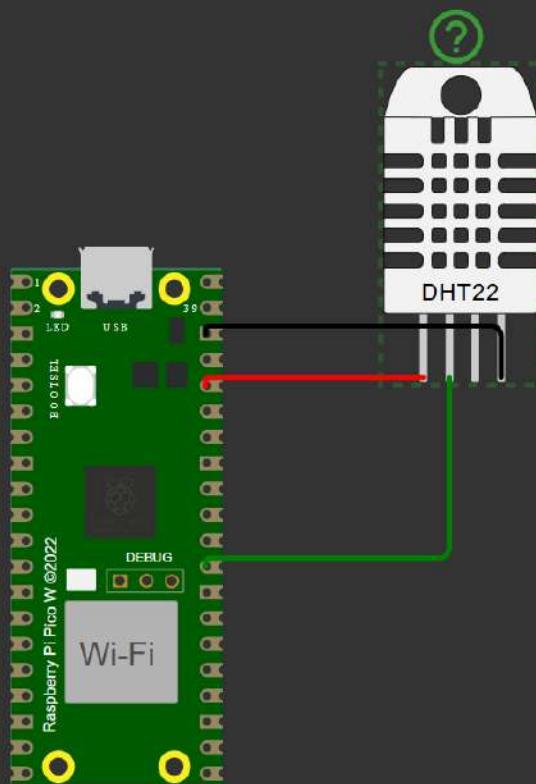
⌚ 00:52.432 🌟 100%



Simulation



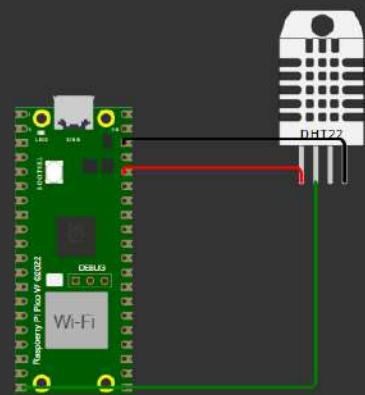
WiFi: Connected to (Public IoT Gateway)



Simulation



⌚ 00:06.322 🌩 99%

 Weather Station (Simulated DHT22)

Temperature: 25.1 °C | Humidity: 50.2 %
Temperature: 25.2 °C | Humidity: 50.4 %
Temperature: 25.3 °C | Humidity: 50.6 %
Temperature: 25.4 °C | Humidity: 50.8 %

