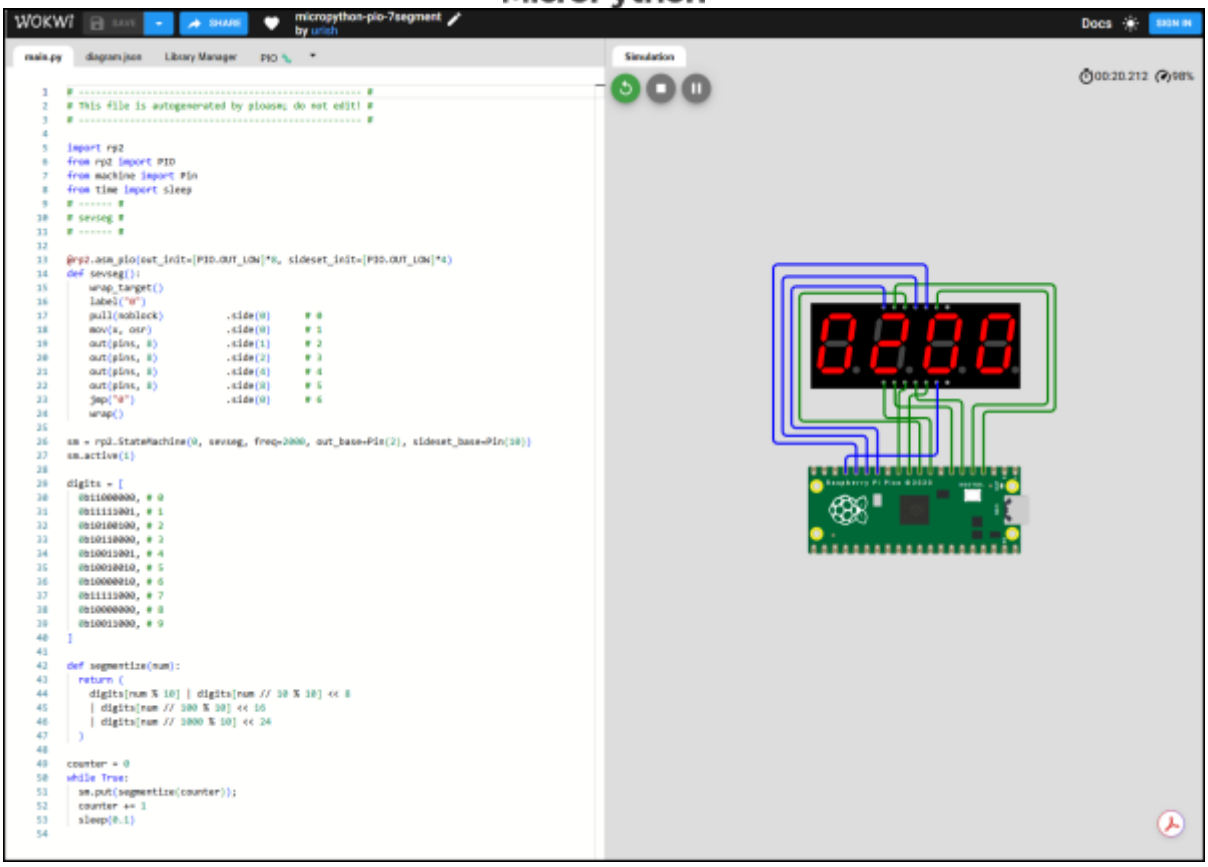
Button Controlled LED

Jul 2026

MicroPython



Introduction

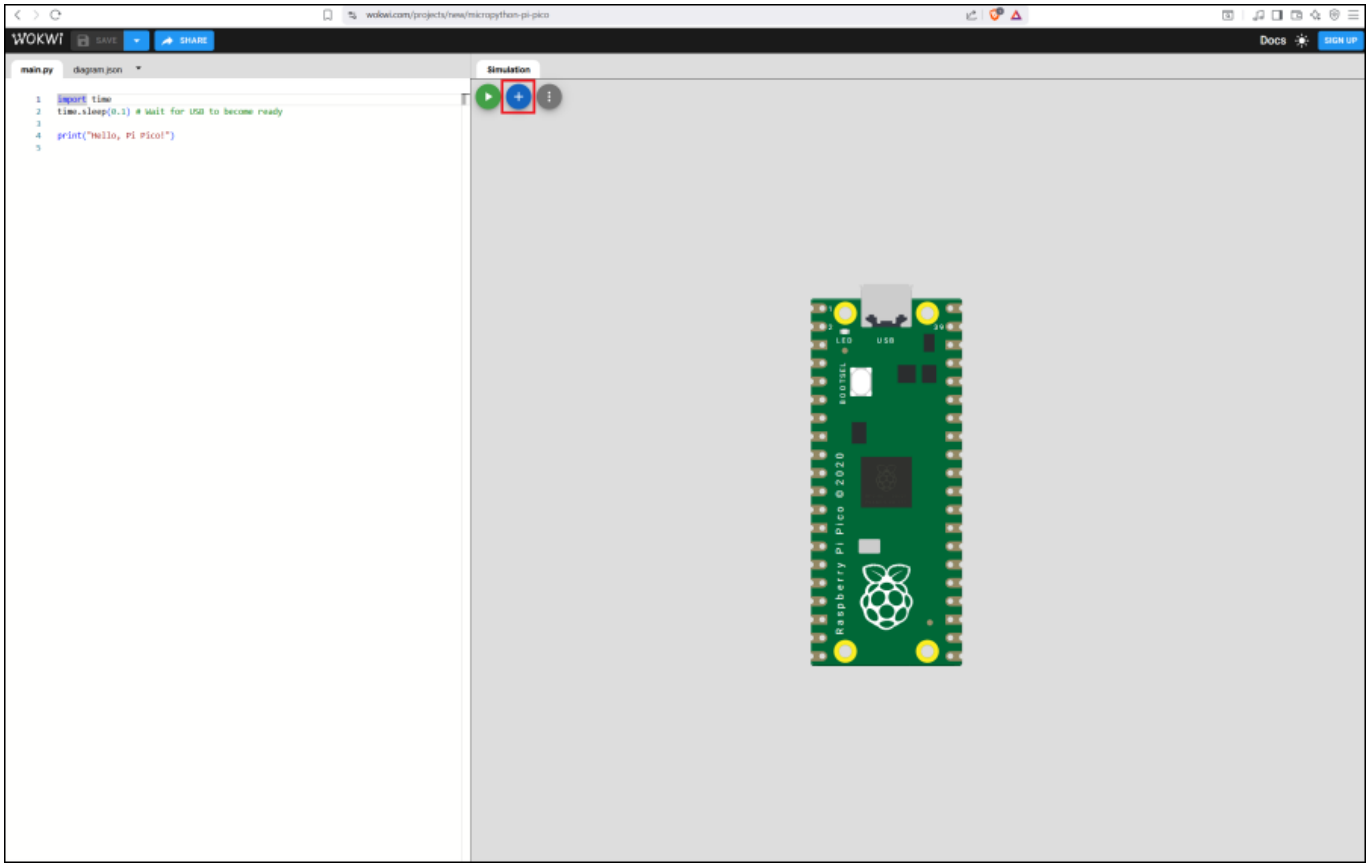
I did not want to lead by talking about a simulator, if you have your Pico, have installed Thonny, have some basic parts, breadboard, LEDs, switches, wires etc, and you want to build something physical, please skip this section and come back to it later.

However, if you have no hardware yet, but want to do some Micro Python work, then you can start here by simulating some electronics, and controlling those electronics with real Micro Python code.

The simulator is called WokWi, and it can be found here. WokWi emulates many devices, including Arduino, ESP32, STM32 and Raspberry Pi Pico. If you want to get directly to the Pico section, then please click the link here.

A Simple LED Button Project

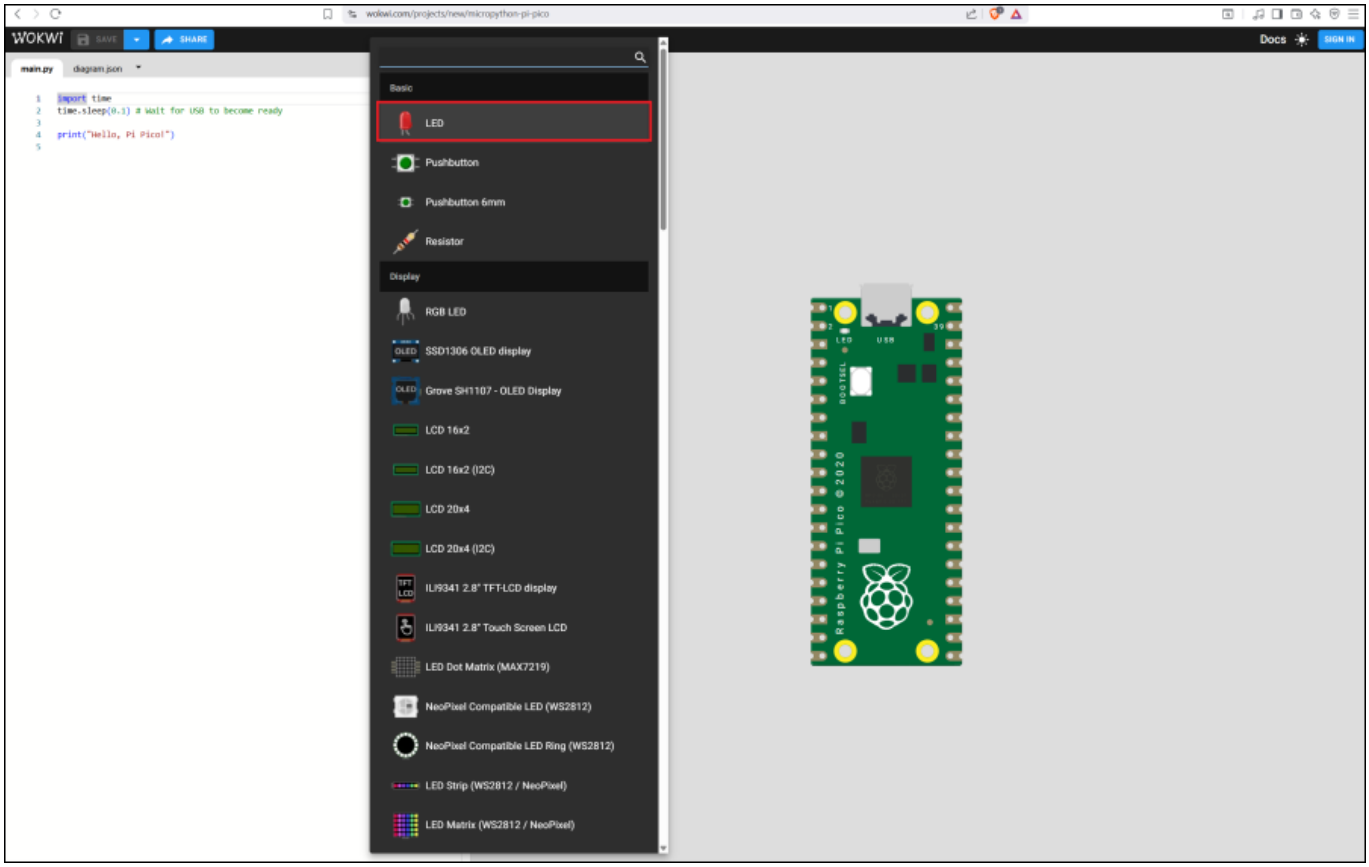
When you first open WokWi, you will be presented with a split page, with your code section on the left and a Raspberry Pi Pico on the right (if you used the direct link).



As can be seen, the Raspberry Pi Pico is already added, there is some simple code, but it does not use any hardware, so it not of any interest to us.

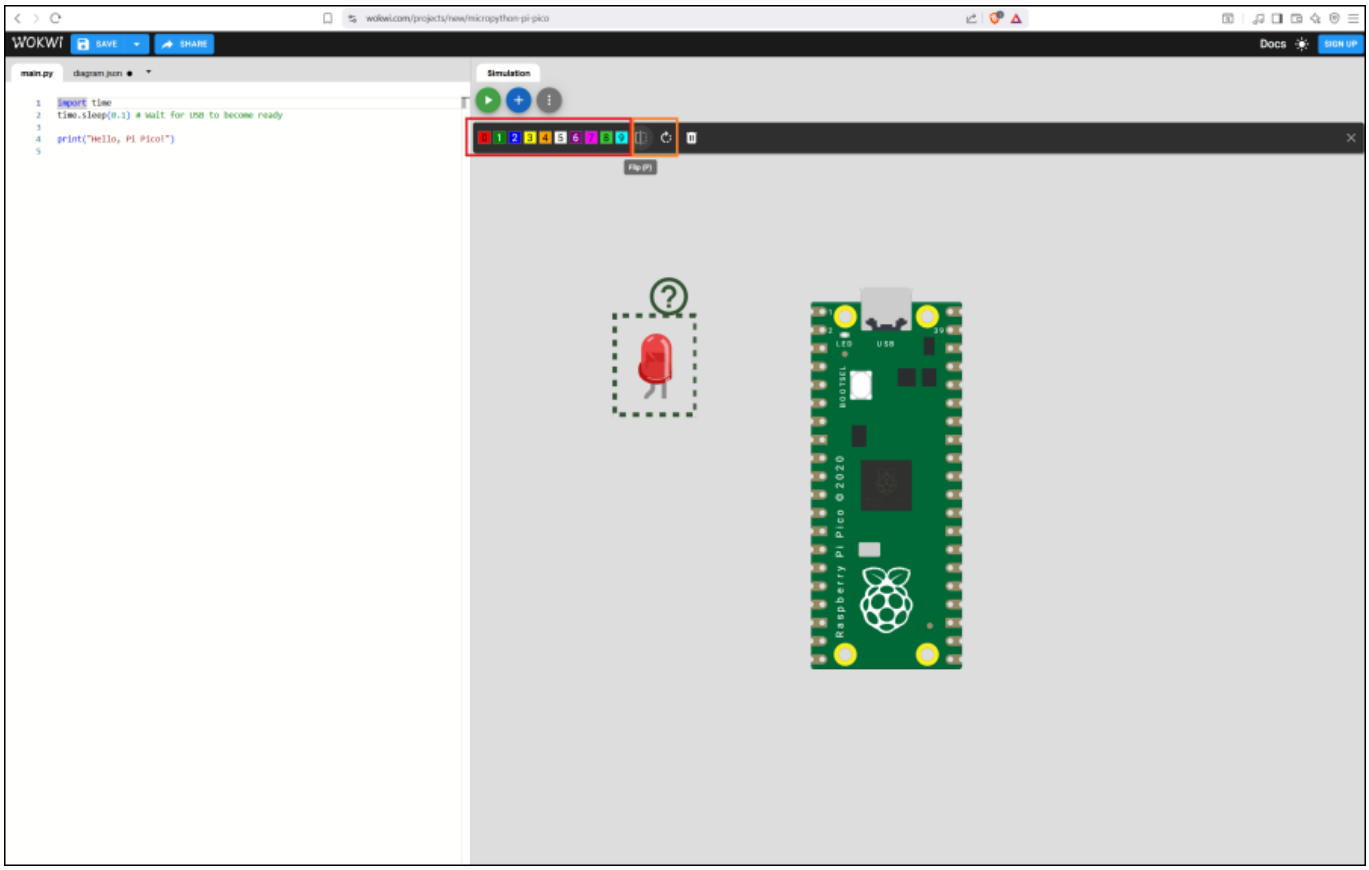
In the top middle of the page, there are three icons, the first is a play button, the second allows you to add components and the third gives some display options.

Click the blue + button.



A drop down will appear where you can select from a list of components, select the LED.

This will place an LED on your active project page.



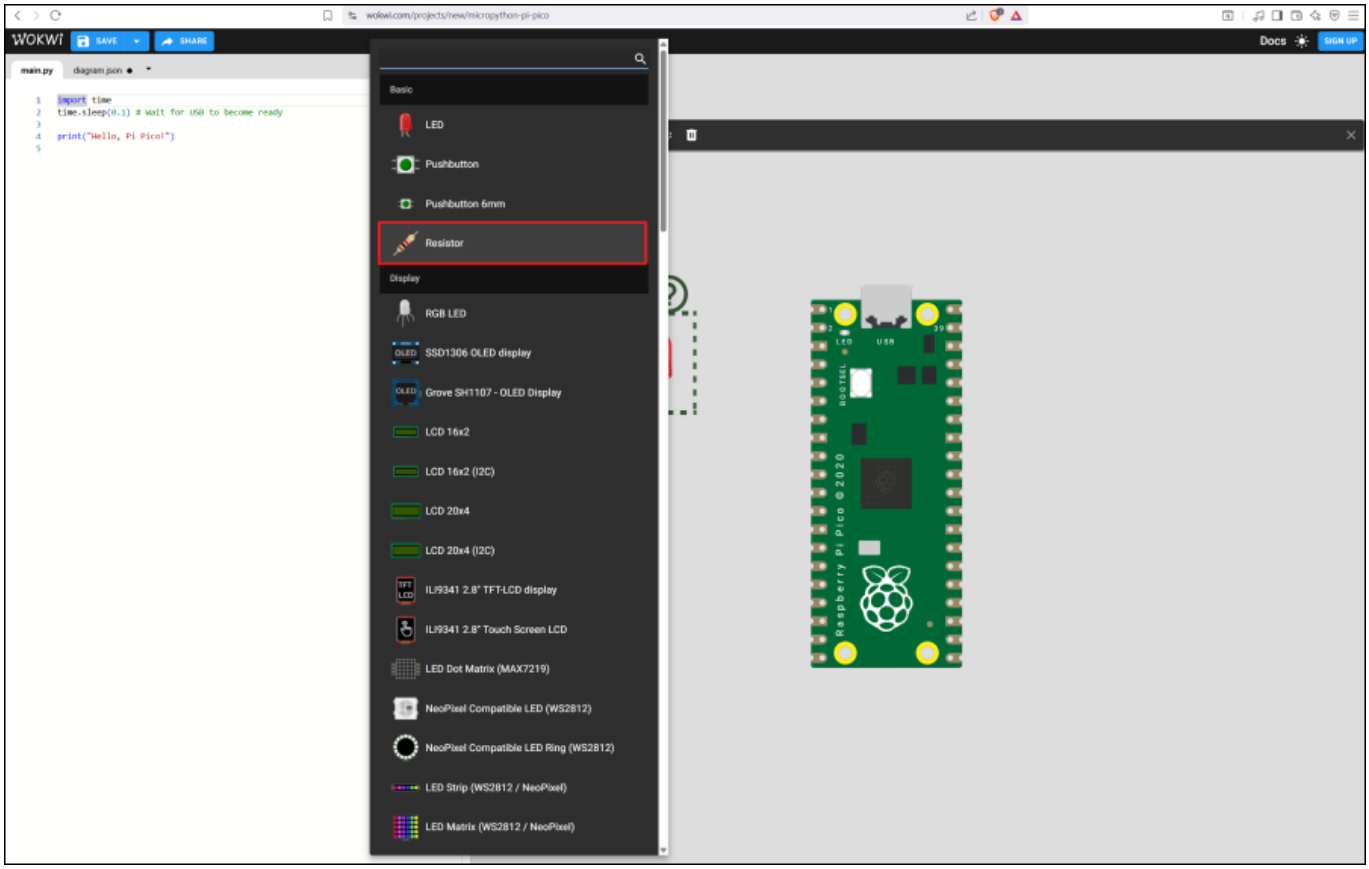
While the LED is selected (has a dotted line around it) you can change some parameters, if you look just below the blue + button, there is now another set of options.

There is a set of coloured squares, this lets you set the colour of your LED, while the colour range is quite nice, I am pretty sure you cannot buy LEDs of these colours!. The next set of icons allow you to rotate and flip the LED image, to make wiring simpler.

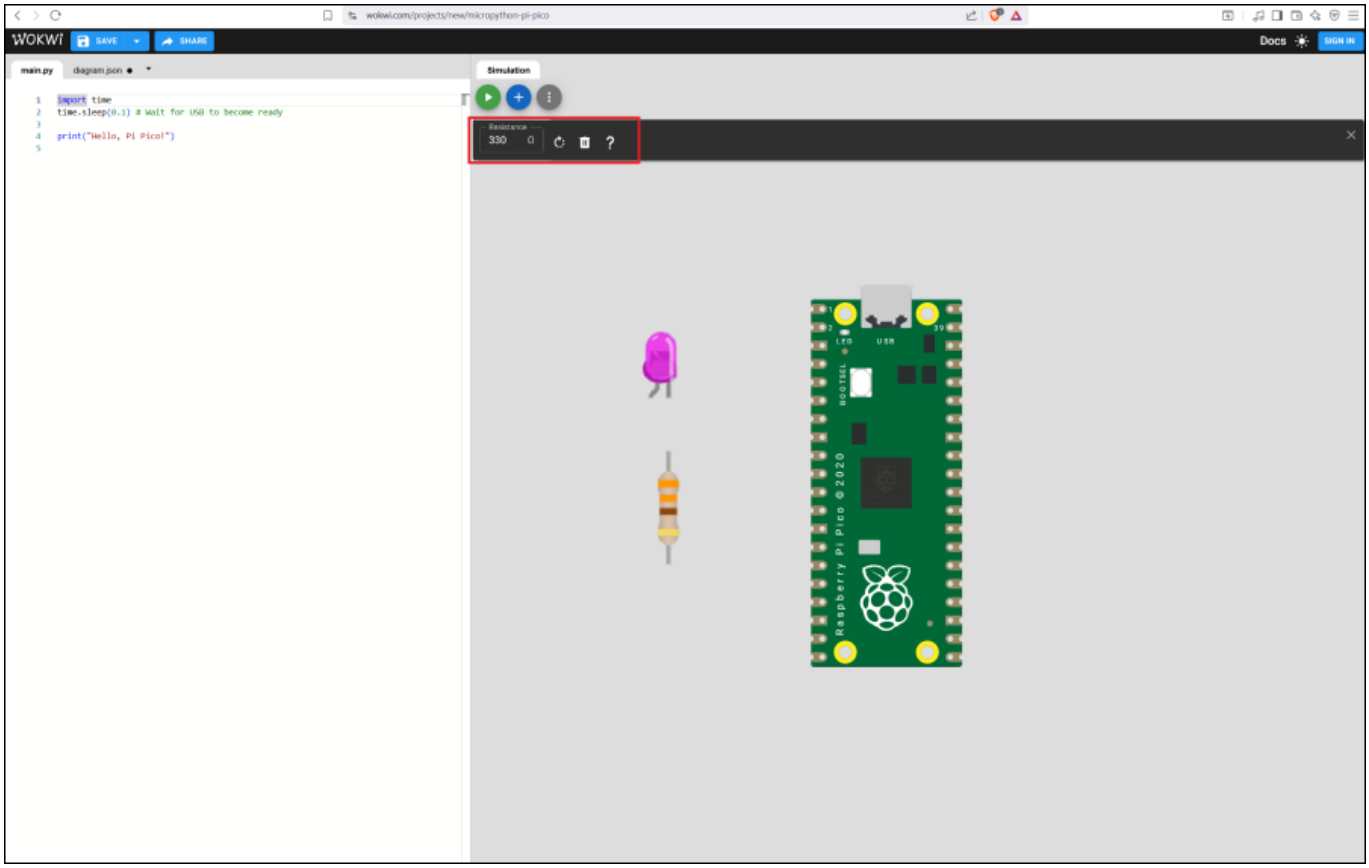
Set your LED the same as mine, with the straight leg (Cathode/negative) on the right and the dog leg (Anode/positive) on the left. You will likely have to flip/rotate the LED to get this view.

Next we are going to add a resistor (a current limiting resistor, so that we don't blow up the LED or damage the Pico).

Click the blue + button again, and select Resistor from the list.



This will add a resistor to our project page.

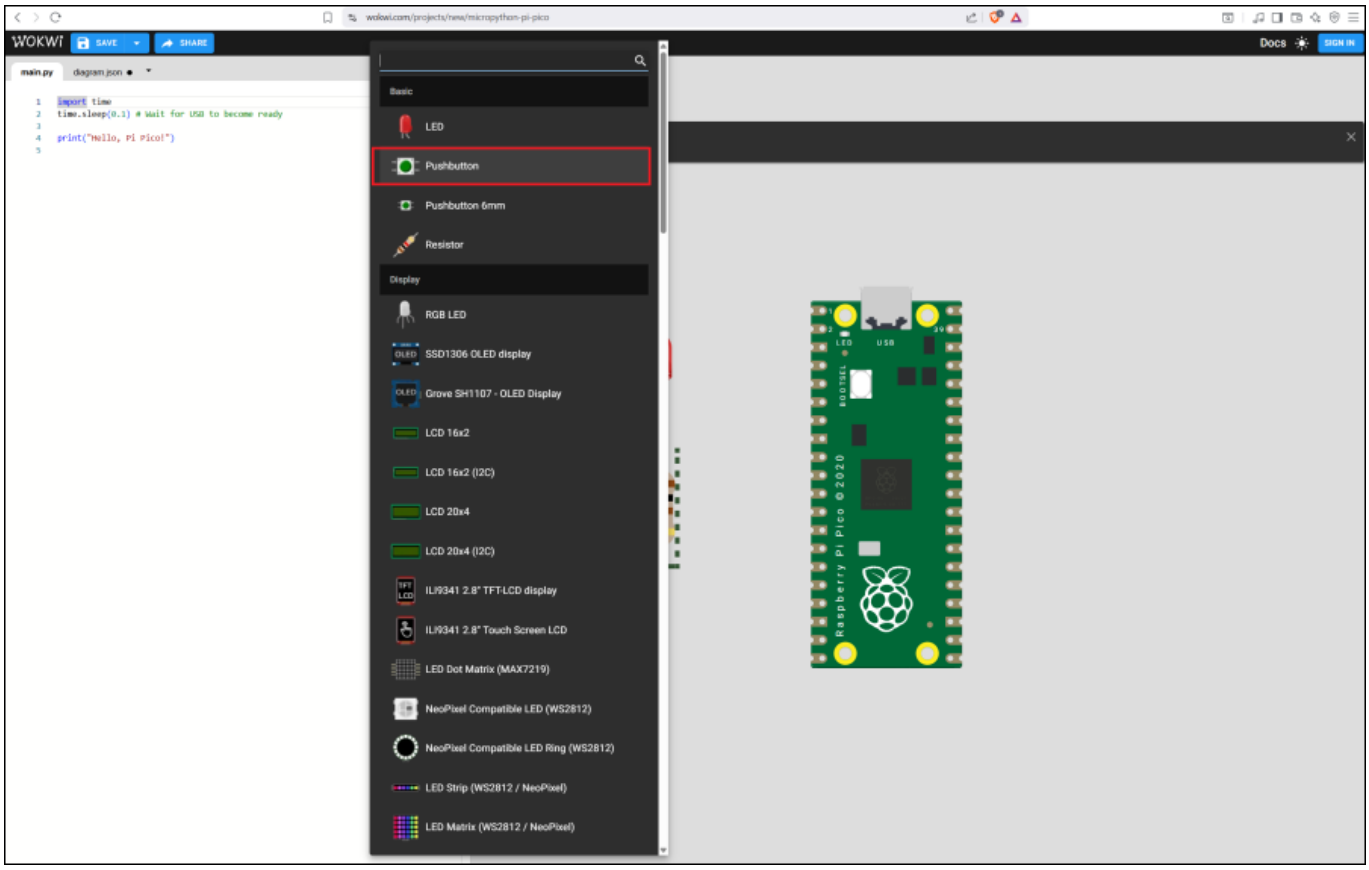


While the resistor is selected, you can change its properties, just under the blue + button. Set the resistor to 330 ohms, and use the rotate symbol to get the correct orientation. The colour code on the resistor will also change to reflect the resistor value.

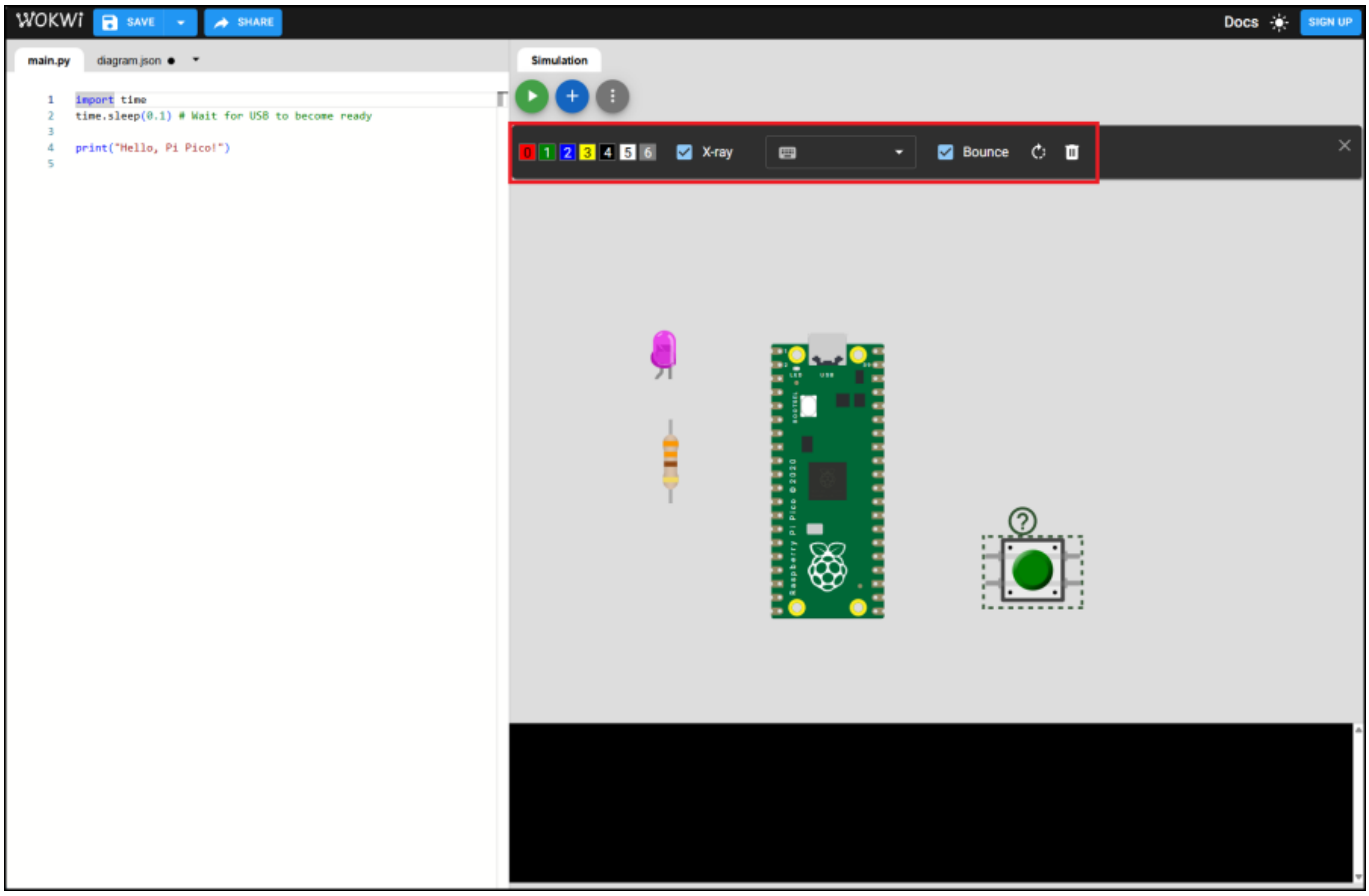
For those of you paying attention, yes, my LED has changed colour, only temporarily though :)

The last component we are going to add, is a push button. Click the blue + button and select Pushbutton. (There are two options,

Pushbutton and Pushbutton 6mm, the 6mm button is visibly smaller).

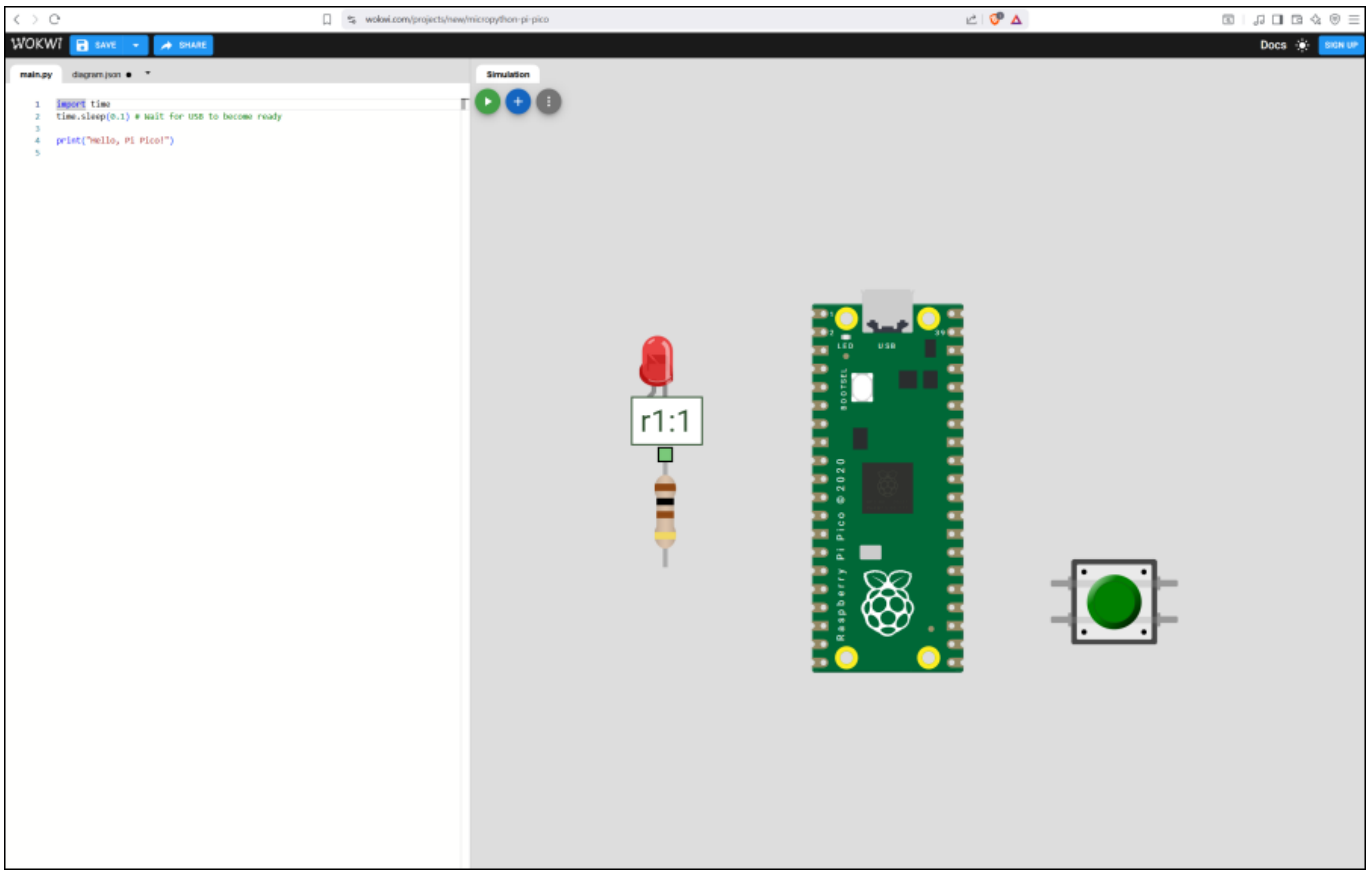


You will see a Pushbutton added to your project.

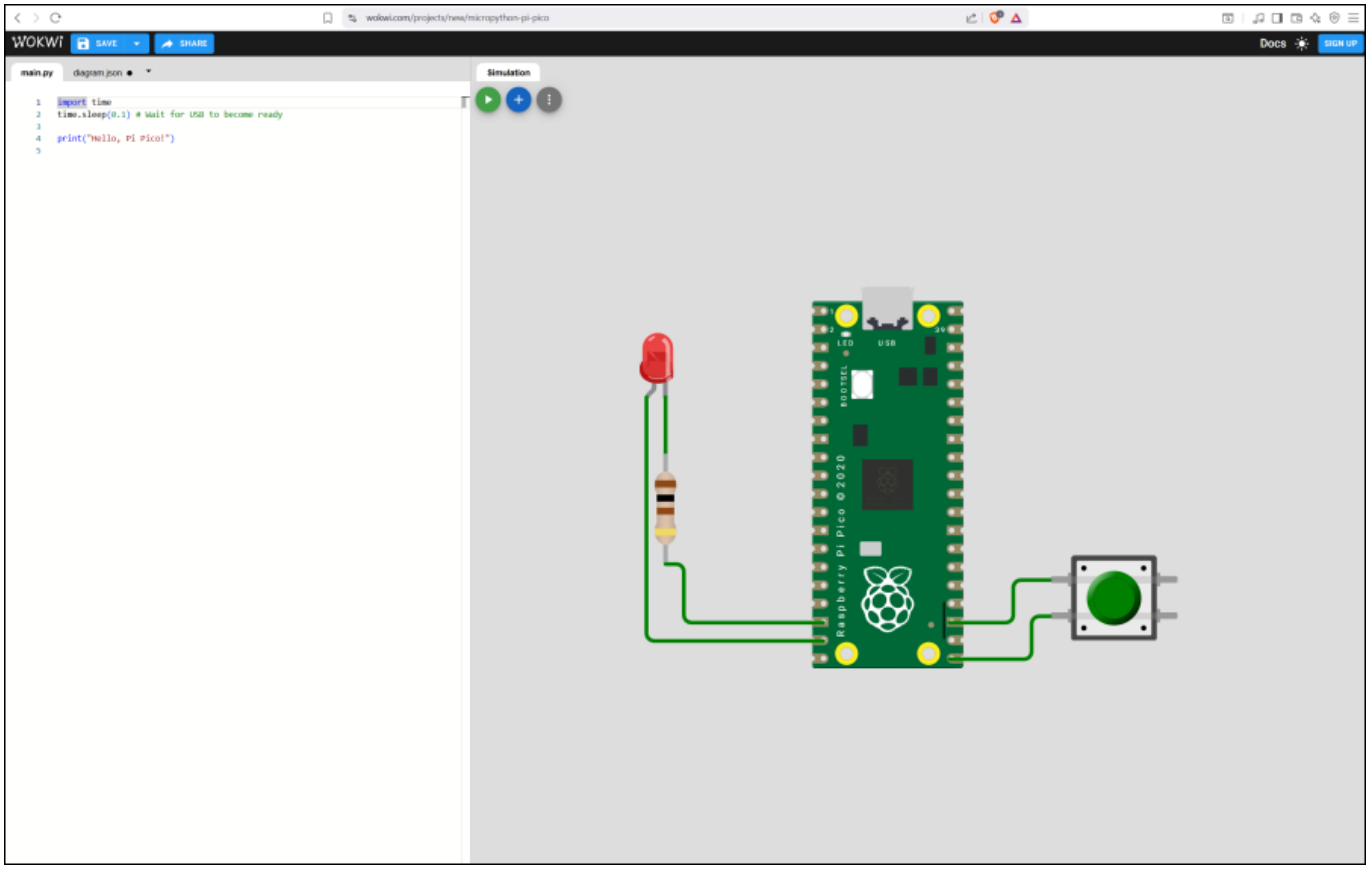


While selected, the Pushbutton has some options, you can change the button colour, X-Ray shows you the button wiring (helpful to use the correct pins) there is a keyboard shortcut, so instead of mouse-clicking the button, you can press a button on your keyboard, and you can rotate the button.

Now we can start to wire up the project, this is very simple, you click on the leg of a component, and drag to the leg of another component or to the Raspberry Pi Pico.



It takes a little practice to get the routing correct, but have a go, you should end up with something identical to the image below.



This is the hardware part complete. What we have done, is connected the following:

- The LED/Resistor between the Raspberry Pi Pico GP14 pin and GND.

- The button between GPIO 16 and GND. (GND = Ground).

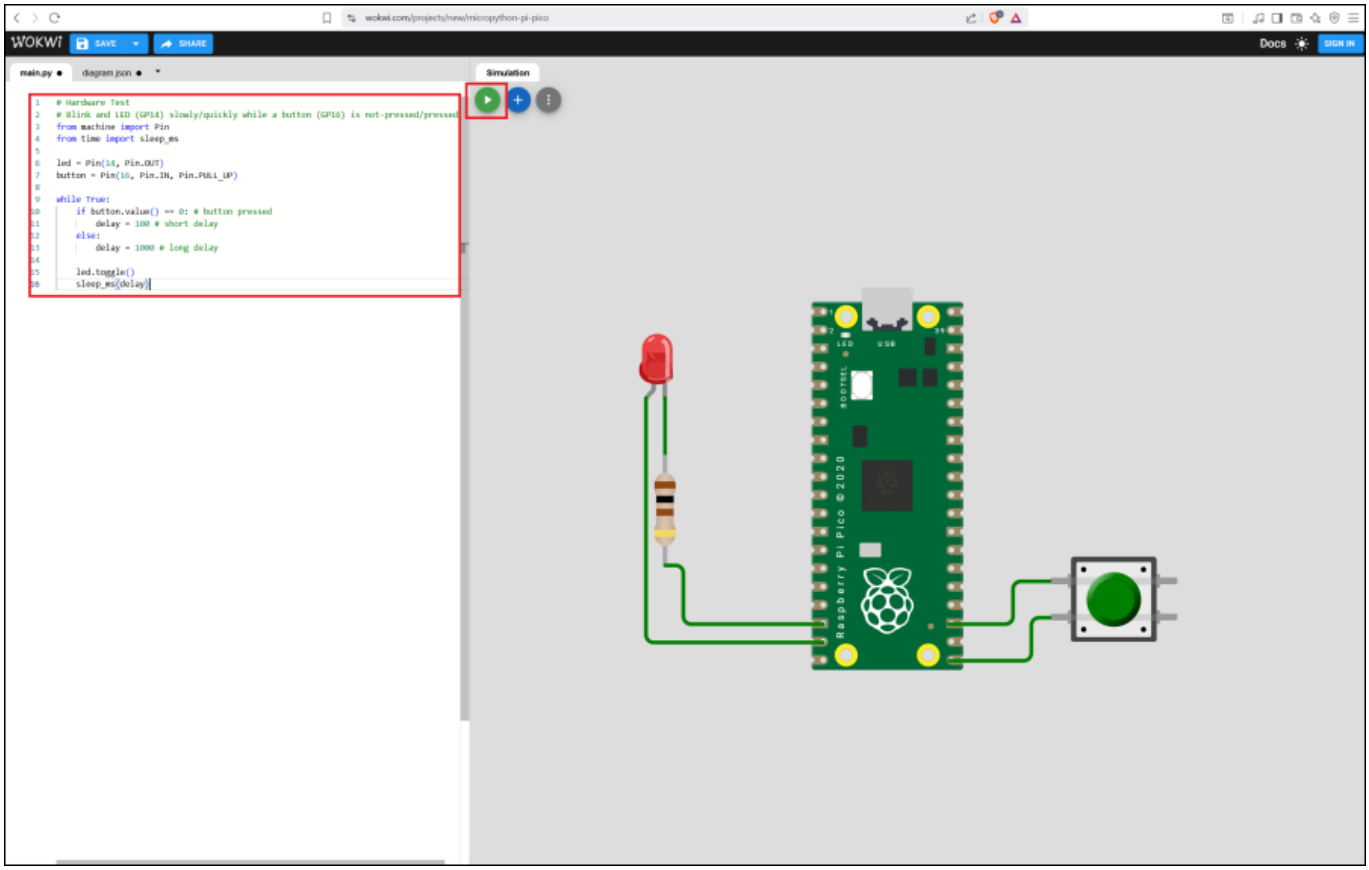
Now, we need to add the Micro Python code.

| download

```
# Hardware Test
# Blink and LED (GP14) slowly/quickly while a button (GP16) is not-pressed/pressed

from machine import Pin          # Import Pin class to control GPIO pins
from time import sleep_ms        # Import millisecond sleep function

led = Pin(14, Pin.OUT)           # Create an output pin on GPIO14 for the LED
button = Pin(16, Pin.IN, Pin.PULL_UP)  # Create an input pin on GPIO16 with internal pull-up
resistor

while True:                      # Infinite loop
    if button.value() == 0:      # If button is pressed (input pulled LOW)
        delay = 100              # Use short delay (fast blink)
    else:                        # Otherwise (button not pressed)
        delay = 1000             # Use long delay (slow blink)

    led.toggle()                 # Flip LED state: ON → OFF or OFF → ON
    sleep_ms(delay)              # Wait for the chosen delay before next toggle
```

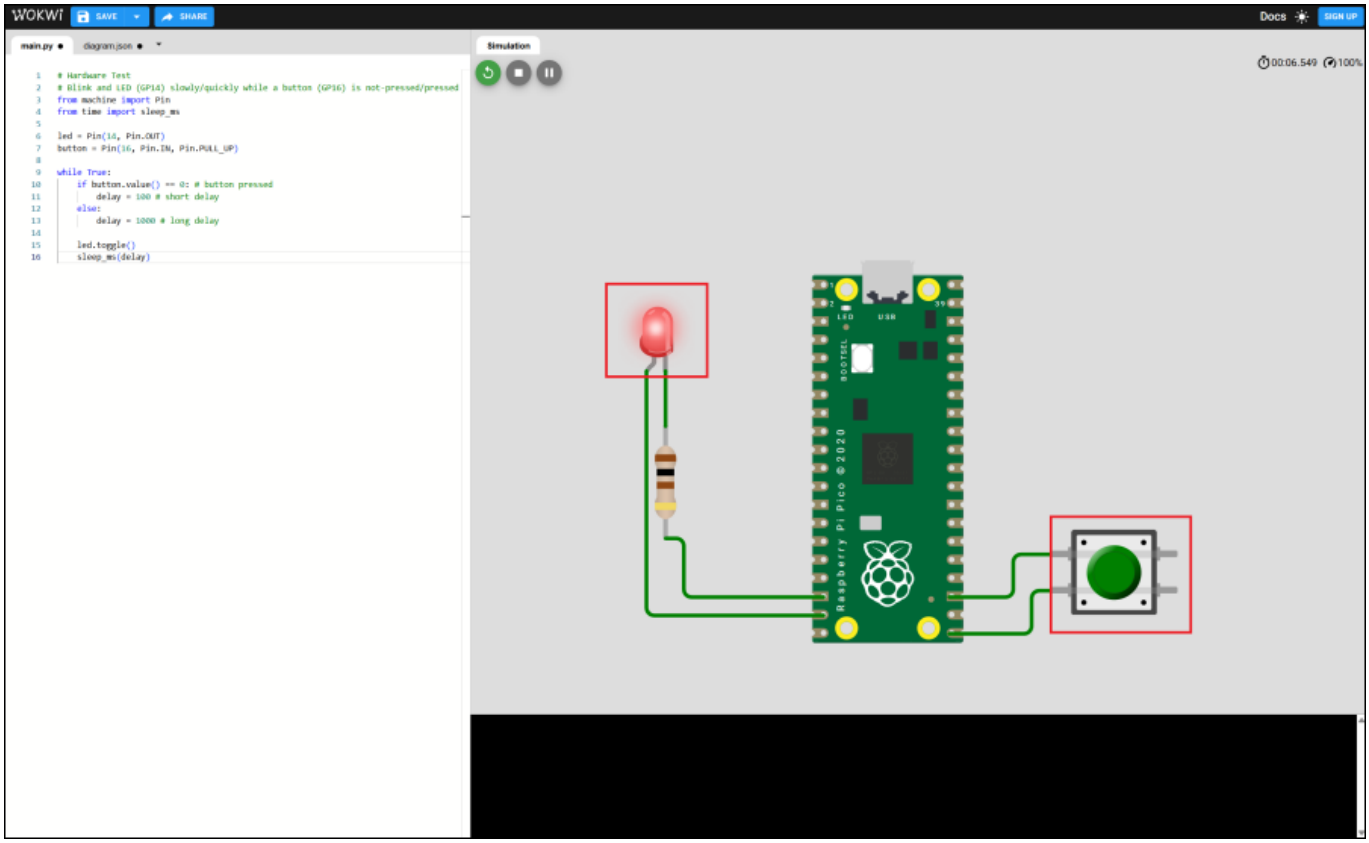
Download the above code, and paste it in to the left hand window of the WokWi page.

You will see the code displayed in the WokWi application.



To start the Micro Python code, click the green play button.

You will see the LED start to flash slowly.



If you click the button with the mouse (or shortcut if you created one) then the LED will start to flash much faster.

Congratulations, you have created your first simulated project. You could now build a physical version of this project to do a real world test. Hopefully you can now see the value of this simulator, its great to test stuff before going to the effort of building something physical, which can be very time consuming. It's great to know if an idea you have will work, before you build it, only to find it does not :)
