LAB 3: ESP32 PROGRAMMING

DAS

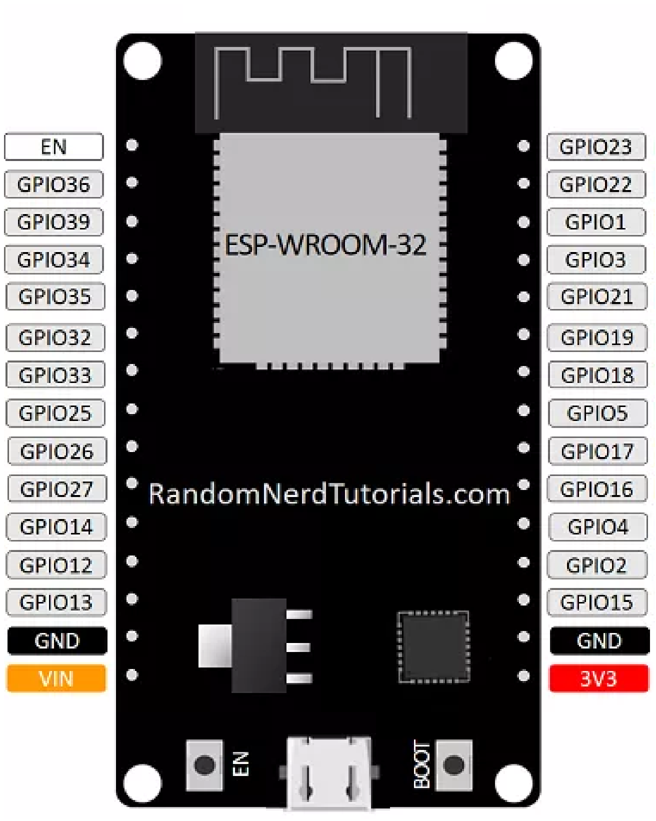
LAB 3: ESP32 PROGRAMMING

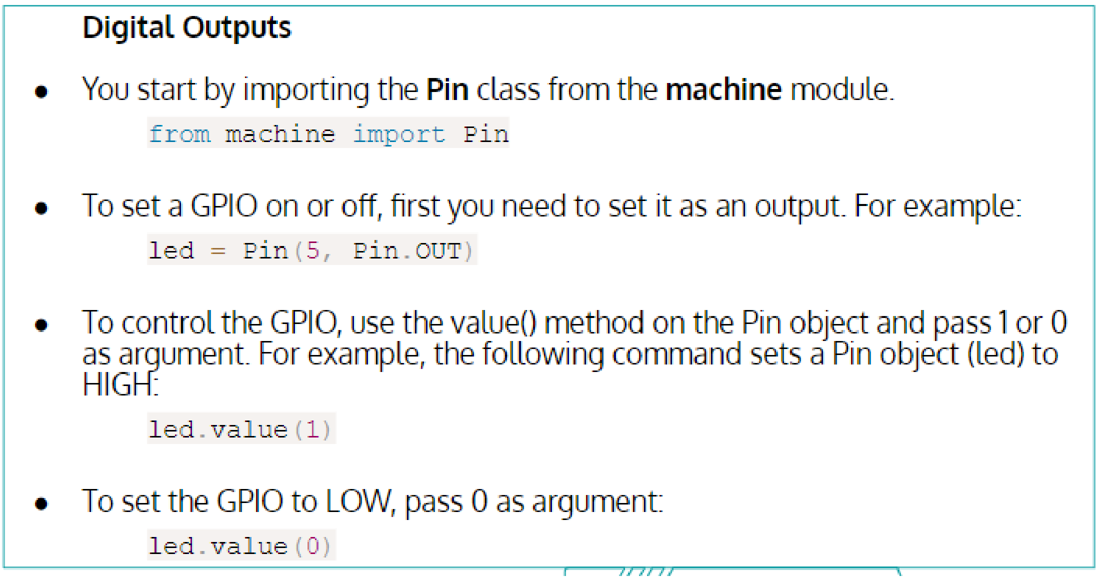
Objective:
In this lab we are going to code ESP32 microcontroller using uPyCraft IDE. Throughout this lab, we will cover ESP32 digital input and output, analog input, DHT sensor and PWM.

Steps:

ESP32 Digital Output Pin

1. We start with ESP32 digital output.




Digital Outputs

- You start by importing the **Pin** class from the **machine** module.

```
from machine import Pin
```

- To set a GPIO on or off, first you need to set it as an output. For example:

```
led = Pin(5, Pin.OUT)
```

- To control the GPIO, use the value() method on the Pin object and pass 1 or 0 as argument. For example, the following command sets a Pin object (led) to HIGH:

```
led.value(1)
```

- To set the GPIO to LOW, pass 0 as argument:

```
led.value(0)
```

2. Prepare all the components needed.

Blinking LED

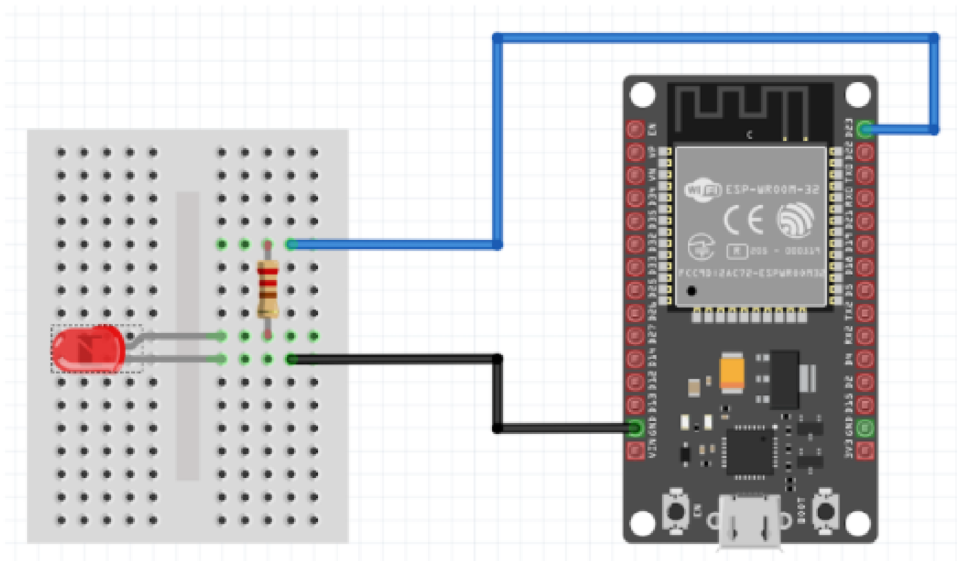
What you'll need:

1) LED in any color (1 unit)
2) Resistor 330 Ohm (1 unit)
3) Male to Female jumper wire (2 unit)
4) ESP32 (1 unit)
5) Breadboard (1 unit)
6) USB Micro B cable (1 unit)

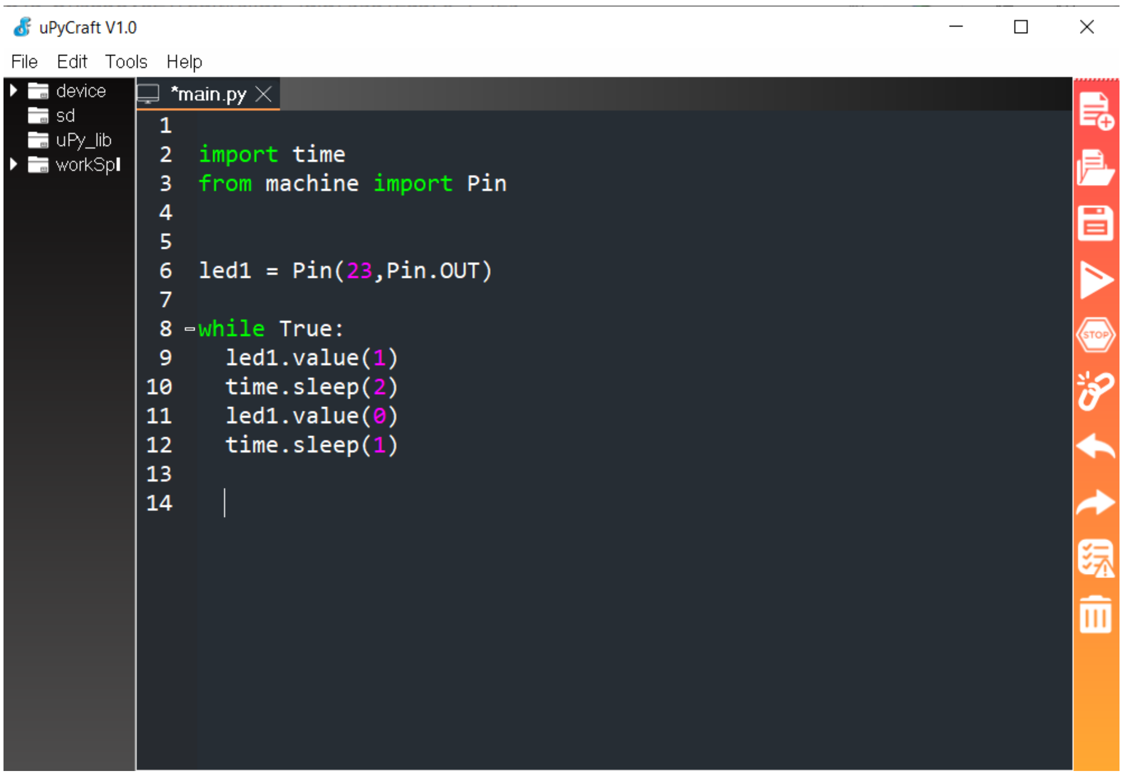
Instructions:

The LED will light up for 2 seconds and turn off for 1 second.

3. Do wiring as shown below.



4. Write the program into uPyCraft IDE, in the main.py and download it into ESP32 board.



```

import time
from machine import Pin


led1 = Pin(23,Pin.OUT)

while True:
  led1.value(1)
  time.sleep(2)
  led1.value(0)
  time.sleep(1)

  
```

5. Do it yourself:

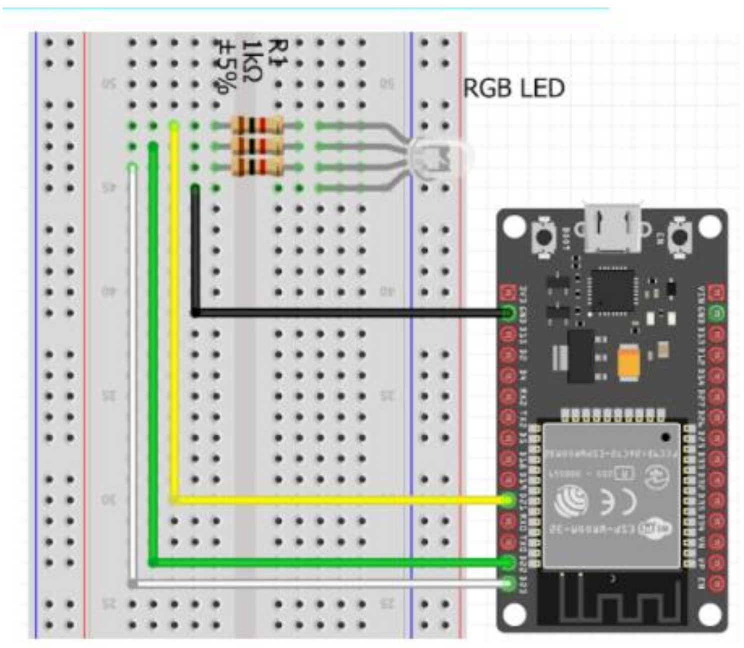
RGB LED Runnning Light

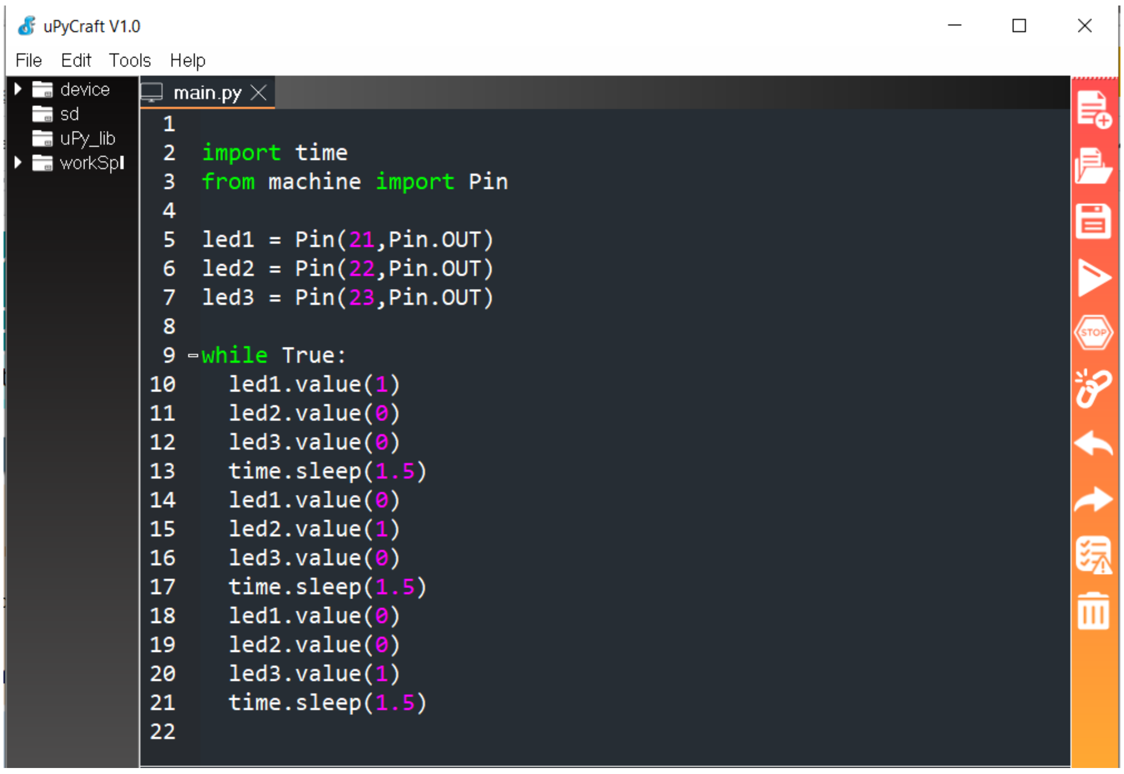
What you'll need:

1) RGB LED (1 unit)
2) Resistor 1K ohm (3 unit)
3) Male to Female jumper wire (5 unit)
4) ESP32 (1 unit)
5) Breadboard (1 unit)
6) USB Micro B cable (1 unit)

Instructions:

The LED will light up in GREEN for 1 second follows by RED and lastly with BLUE

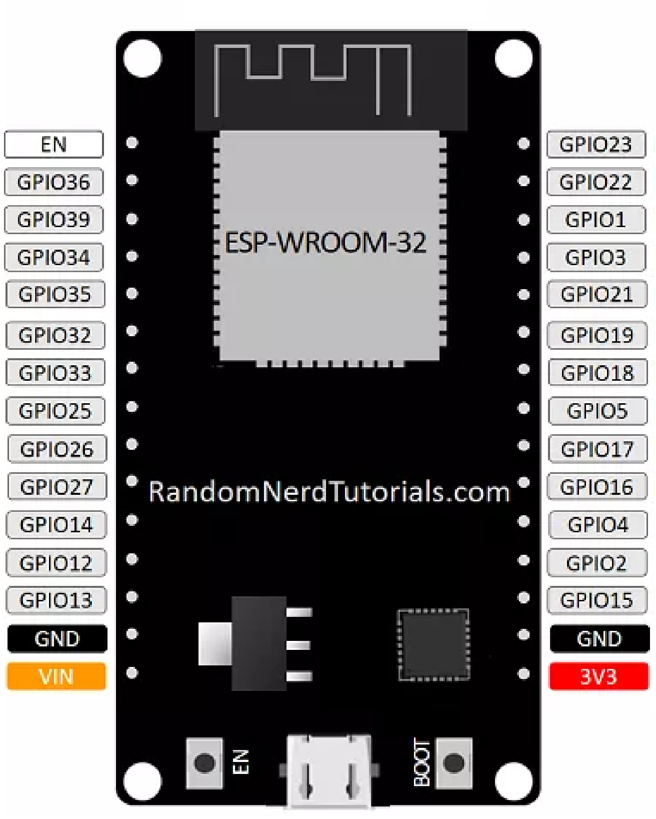


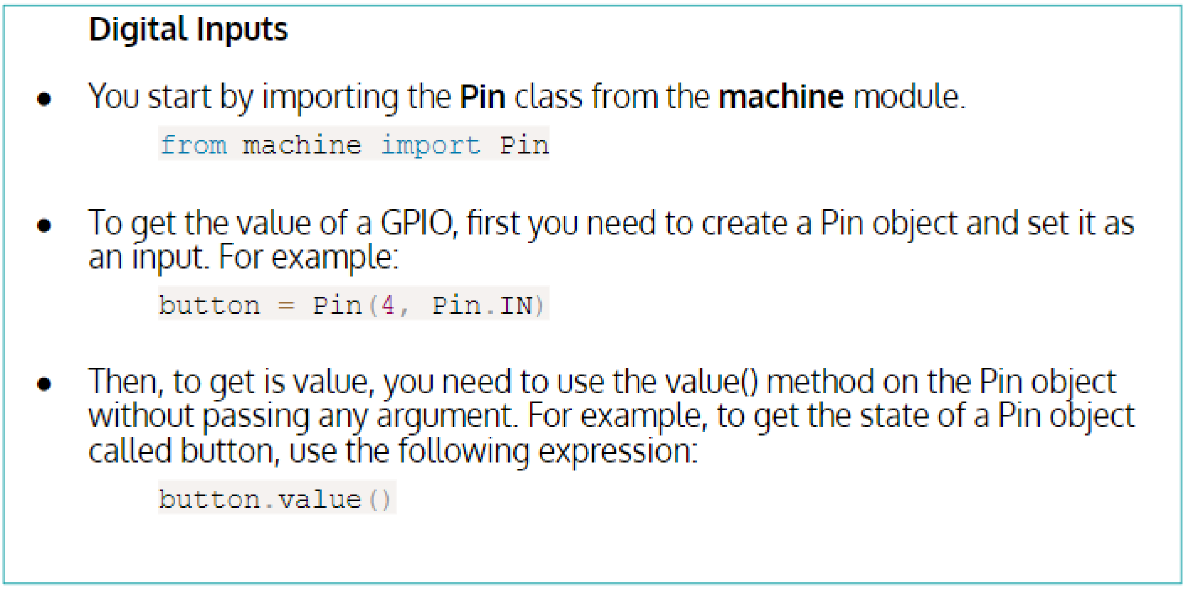


```

import time
from machine import Pin

led1 = Pin(21,Pin.OUT)
led2 = Pin(22,Pin.OUT)
led3 = Pin(23,Pin.OUT)

while True:
  led1.value(1)
  led2.value(0)
  led3.value(0)
  time.sleep(1.5)
  led1.value(0)
  led2.value(1)
  led3.value(0)
  time.sleep(1.5)
  led1.value(0)
  led2.value(0)
  led3.value(1)
  time.sleep(1.5)

```

ESP32 Digital Input Pin

1. We continue with ESP32 digital input.





Digital Inputs

- You start by importing the **Pin** class from the **machine** module.

```
from machine import Pin
```

- To get the value of a GPIO, first you need to create a Pin object and set it as an input. For example:

```
button = Pin(4, Pin.IN)
```

- Then, to get is value, you need to use the value() method on the Pin object without passing any argument. For example, to get the state of a Pin object called button, use the following expression:

```
button.value()
```

2. Prepare all the components needed.

Button - Digital Input

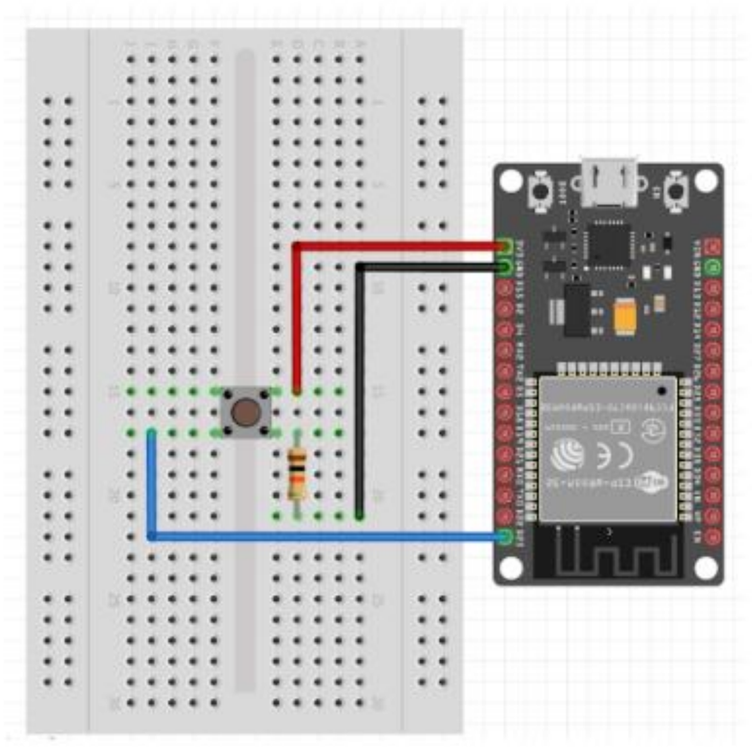
What you'll need:

1) Contact switch or button (1 unit each)
2) Resistor 10K ohm (1 unit)
3) Male to Female jumper wire (3 unit)
4) NodeMCU (1 unit)
5) Breadboard (1 unit)
6) USB Micro B cable (1 unit)

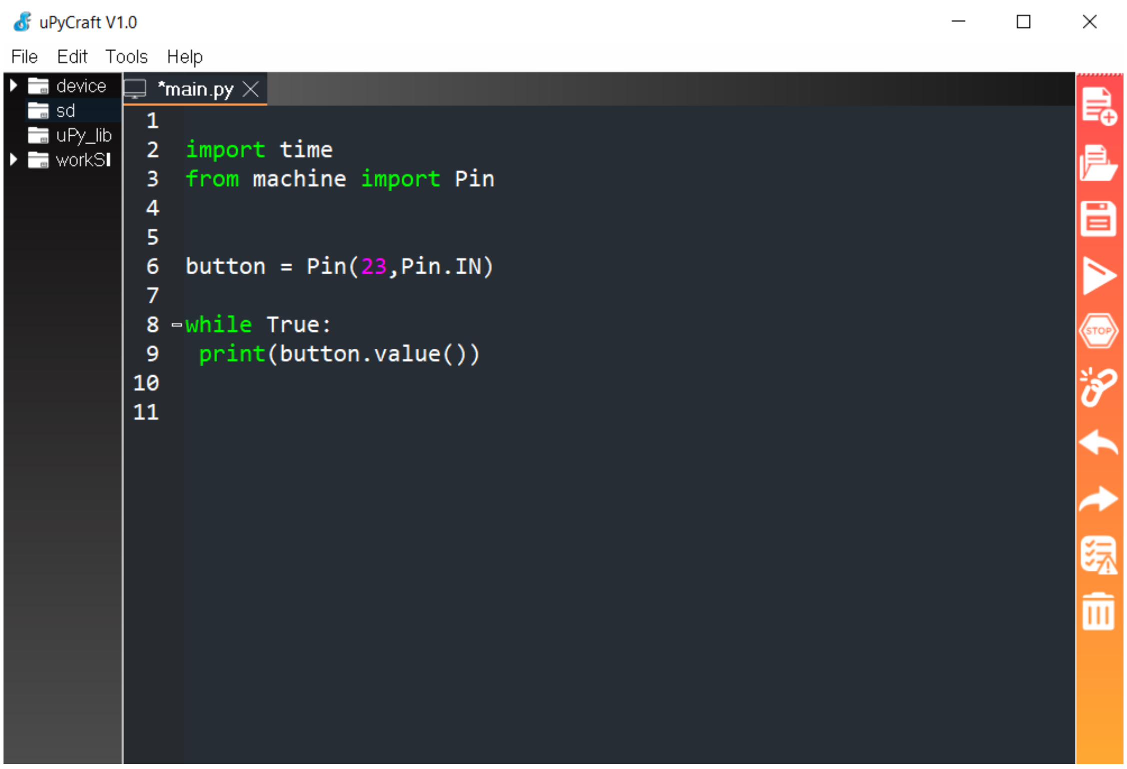
Instructions:

The button will write a value of 1 (ON) when pushed and 0 (OFF) when released.
Show your result at the terminal

3. Do wiring as shown below.



4. Write the program into uPyCraft IDE, in the main.py and download it into ESP32 board.



```

import time
from machine import Pin


button = Pin(23,Pin.IN)

while True:
  print(button.value())


```

5. Do it yourself:

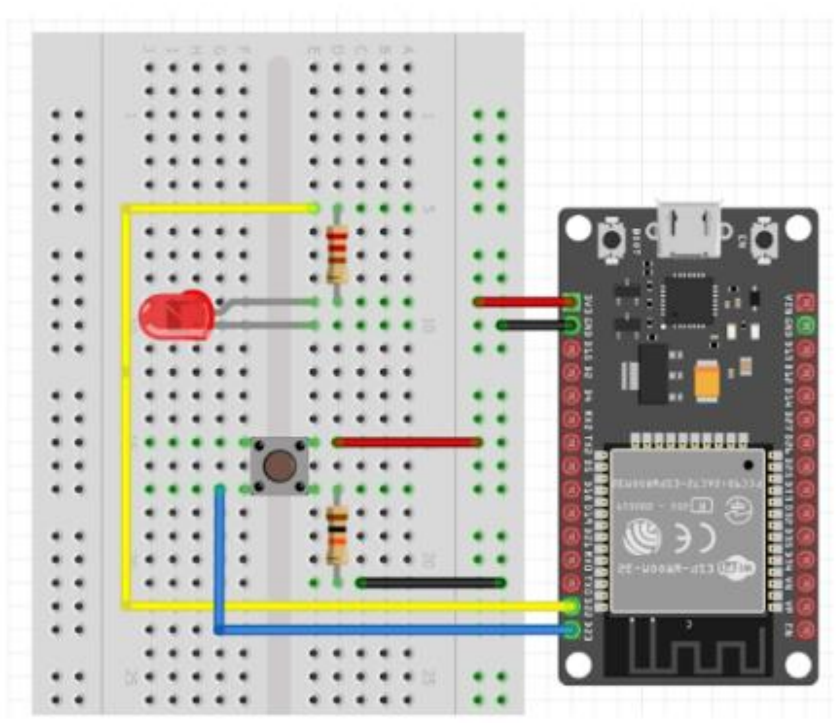
Digital Input & Digital Output

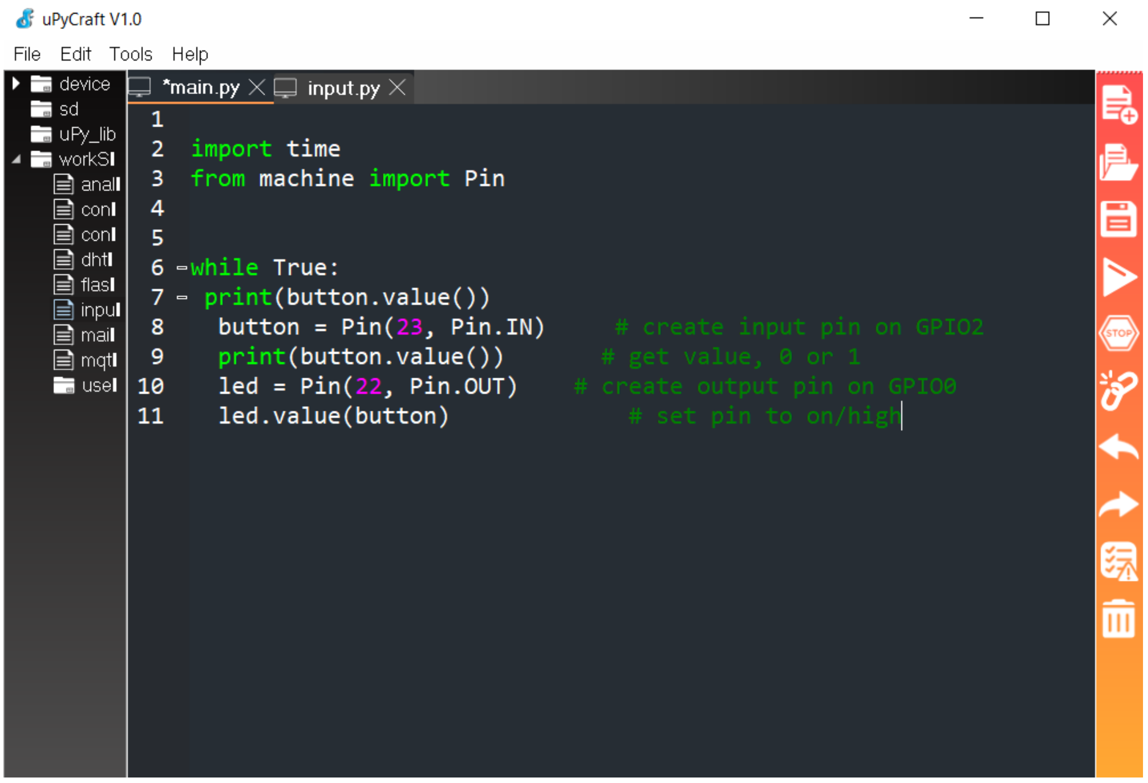
What you'll need:

1) Contact switch or button (1 unit)
2) LED in any color (1 unit)
3) Resistor 10K ohm (1 unit)
4) Resistor 220 ohm (1 unit)
5) Male to Female jumper wire (3 unit)
6) ESP32 (1 unit)
7) Breadboard (1 unit)
8) USB Micro B cable (1 unit)

Instructions:

The button will write a value of 1 (ON) when pushed and will also lights up the LED

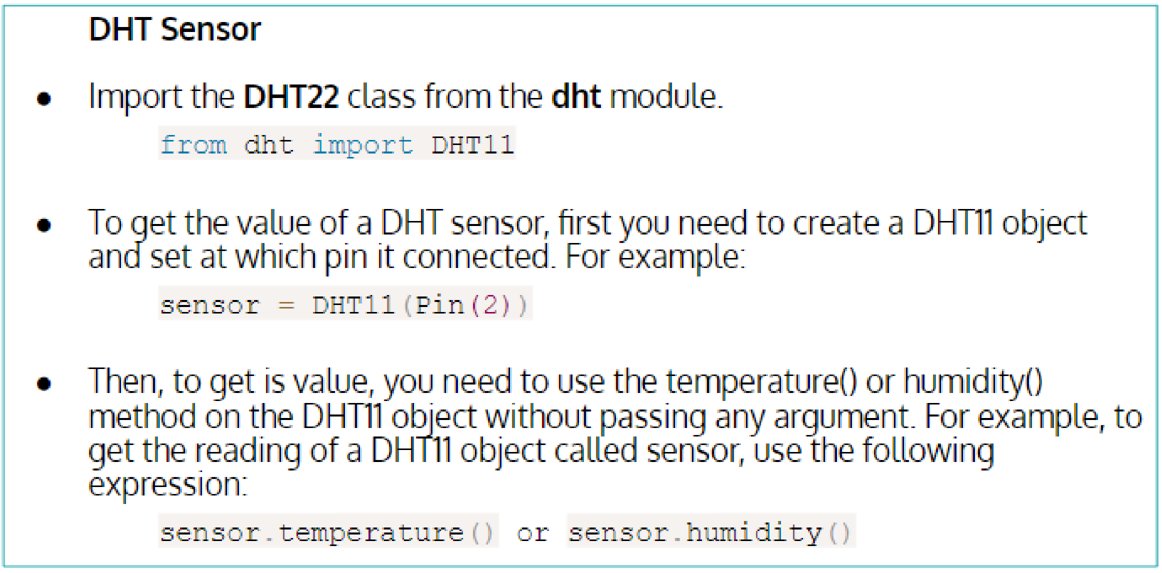




```

import time
from machine import Pin


while True:
  print(button.value())
    button = Pin(23, Pin.IN)      # create input pin on GPIO2
    print(button.value())       # get value, 0 or 1
    led = Pin(22, Pin.OUT)    # create output pin on GPIO0
    led.value(button)             # set pin to on/high
```

ESP32 - DHT sensor

1. We continue with DHT Sensor.



DHT Sensor

- Import the **DHT22** class from the **dht** module.

```
from dht import DHT11
```

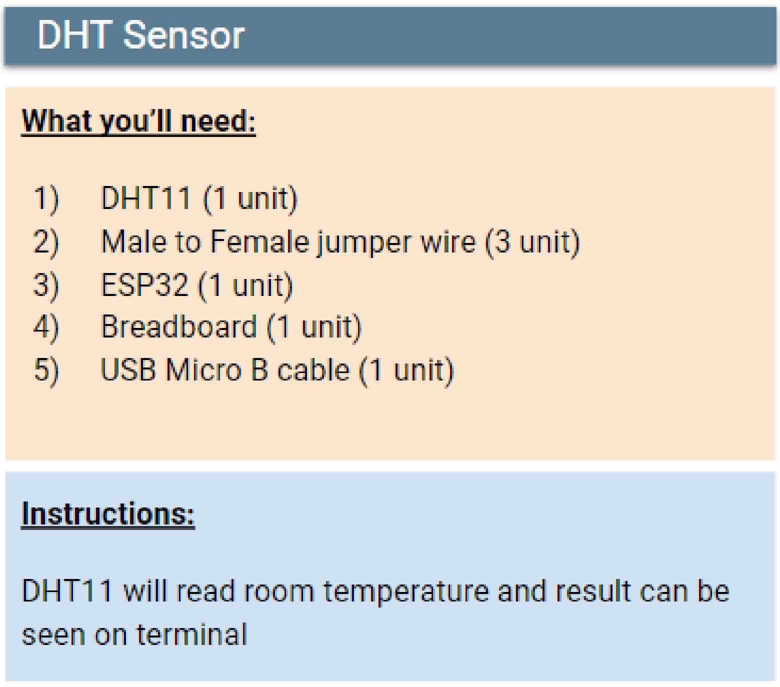
- To get the value of a DHT sensor, first you need to create a DHT11 object and set at which pin it connected. For example:

```
sensor = DHT11(Pin(2))
```

- Then, to get is value, you need to use the temperature() or humidity() method on the DHT11 object without passing any argument. For example, to get the reading of a DHT11 object called sensor, use the following expression:

```
sensor.temperature() or sensor.humidity()
```

2. Prepare all the components needed.



DHT Sensor

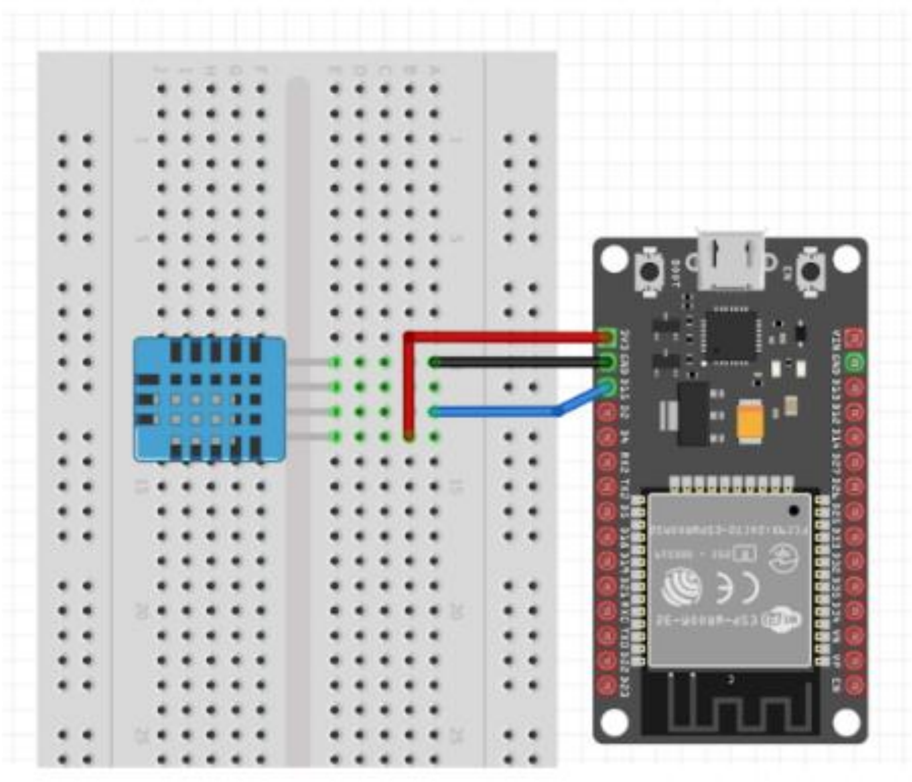
What you'll need:

1) DHT11 (1 unit)
2) Male to Female jumper wire (3 unit)
3) ESP32 (1 unit)
4) Breadboard (1 unit)
5) USB Micro B cable (1 unit)

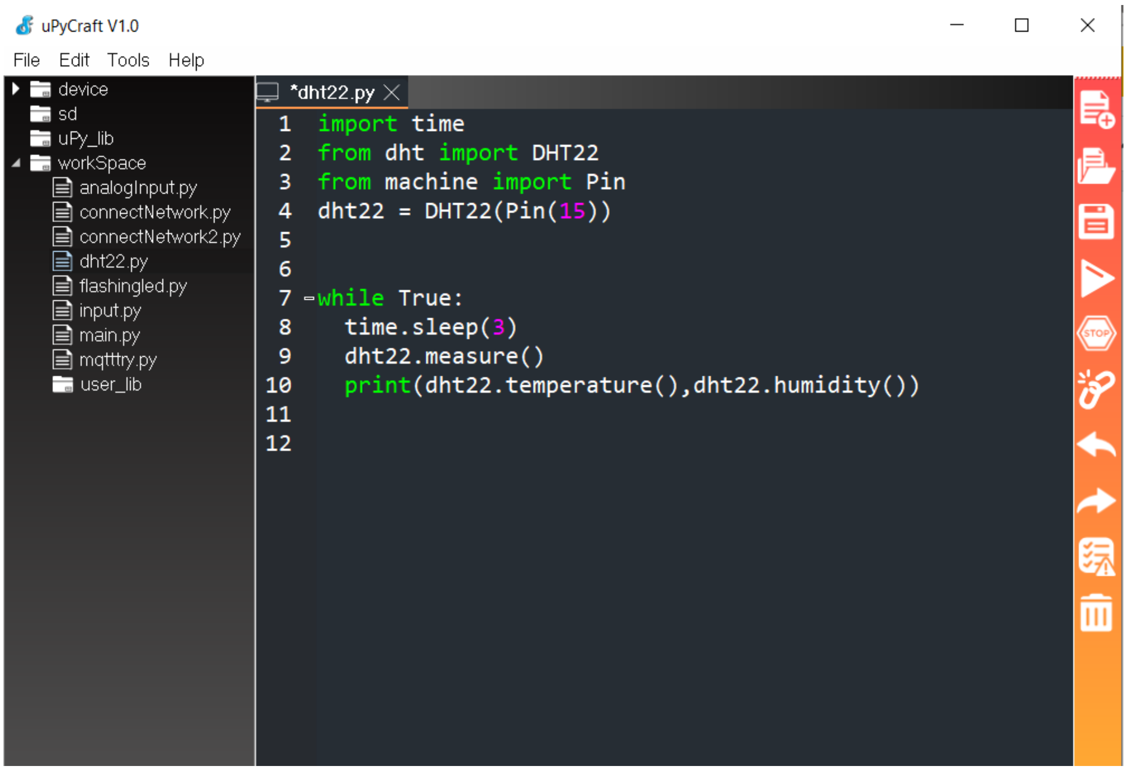
Instructions:

DHT11 will read room temperature and result can be seen on terminal

3. Do wiring as shown below.



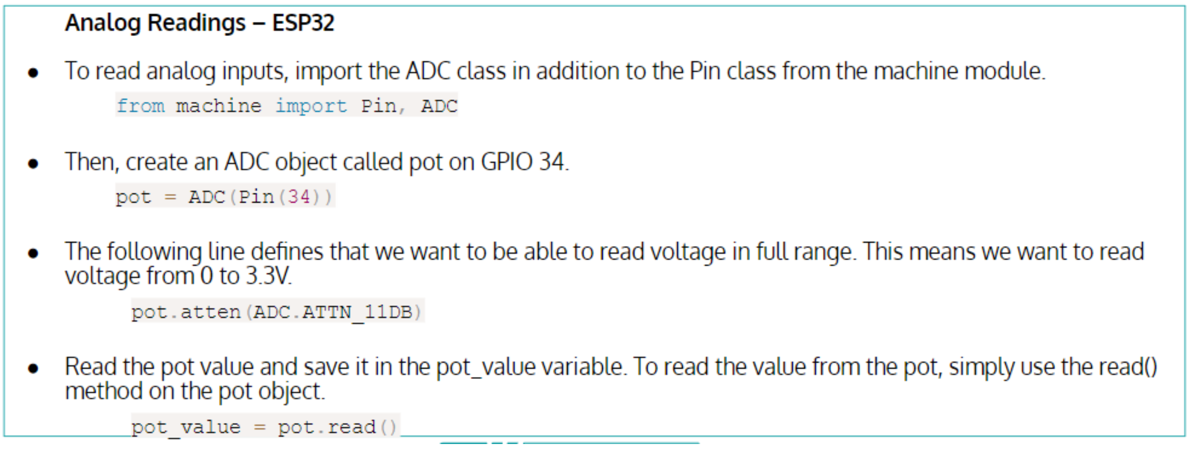
4. Write the program into uPyCraft IDE, in the main.py and download it into ESP32 board.



```
import time
from dht import DHT22
from machine import Pin
dht22 = DHT22(Pin(15))


while True:
    time.sleep(3)
    dht22.measure()
    print(dht22.temperature(),dht22.humidity())


```

ESP32 Analog Input

1. We start with ESP32 analog input.



Analog Readings – ESP32

- To read analog inputs, import the ADC class in addition to the Pin class from the machine module.

```
from machine import Pin, ADC
```

- Then, create an ADC object called pot on GPIO 34.

```
pot = ADC(Pin(34))
```

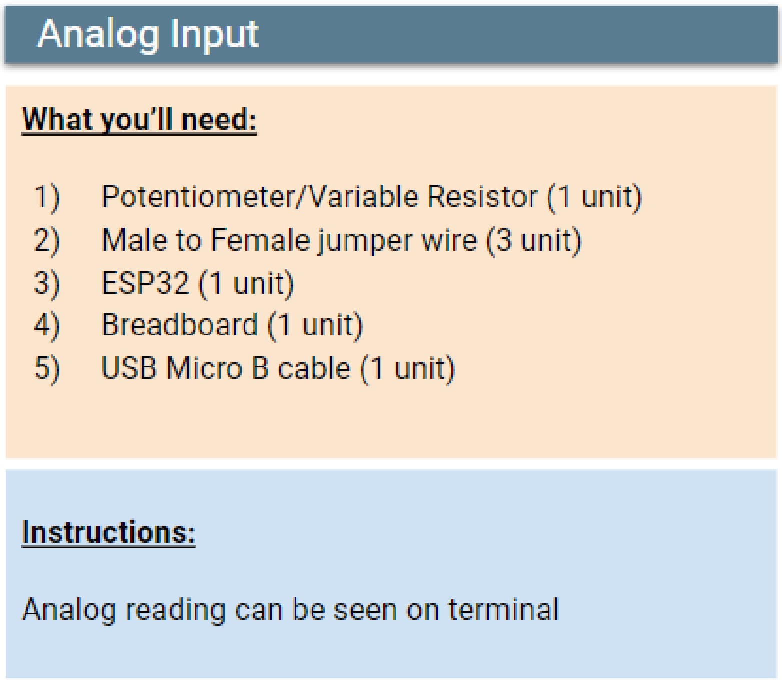
- The following line defines that we want to be able to read voltage in full range. This means we want to read voltage from 0 to 3.3V.

```
pot.atten(ADC.ATTN_11DB)
```

- Read the pot value and save it in the pot_value variable. To read the value from the pot, simply use the read() method on the pot object.

```
pot_value = pot.read()
```

2. Prepare all the components needed.



Analog Input

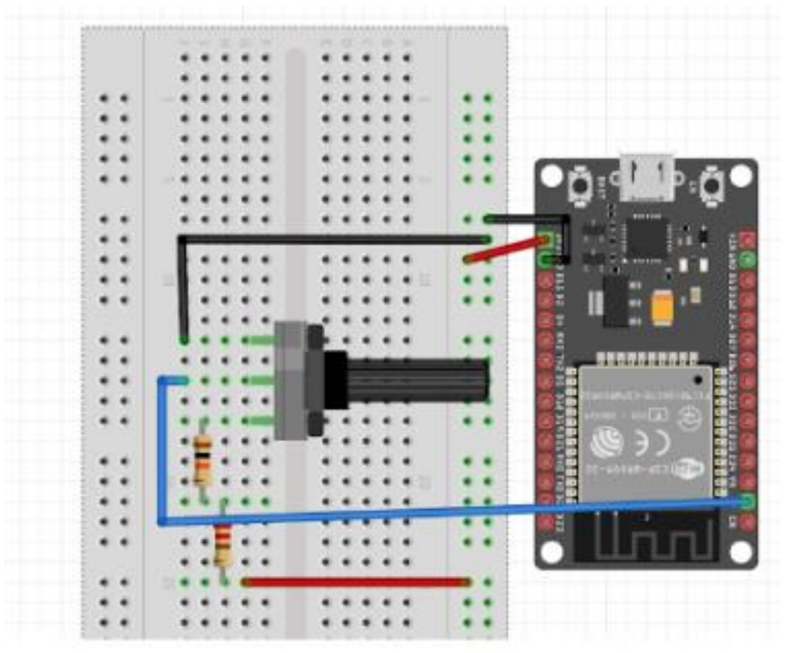
What you'll need:

1) Potentiometer/Variable Resistor (1 unit)
2) Male to Female jumper wire (3 unit)
3) ESP32 (1 unit)
4) Breadboard (1 unit)
5) USB Micro B cable (1 unit)

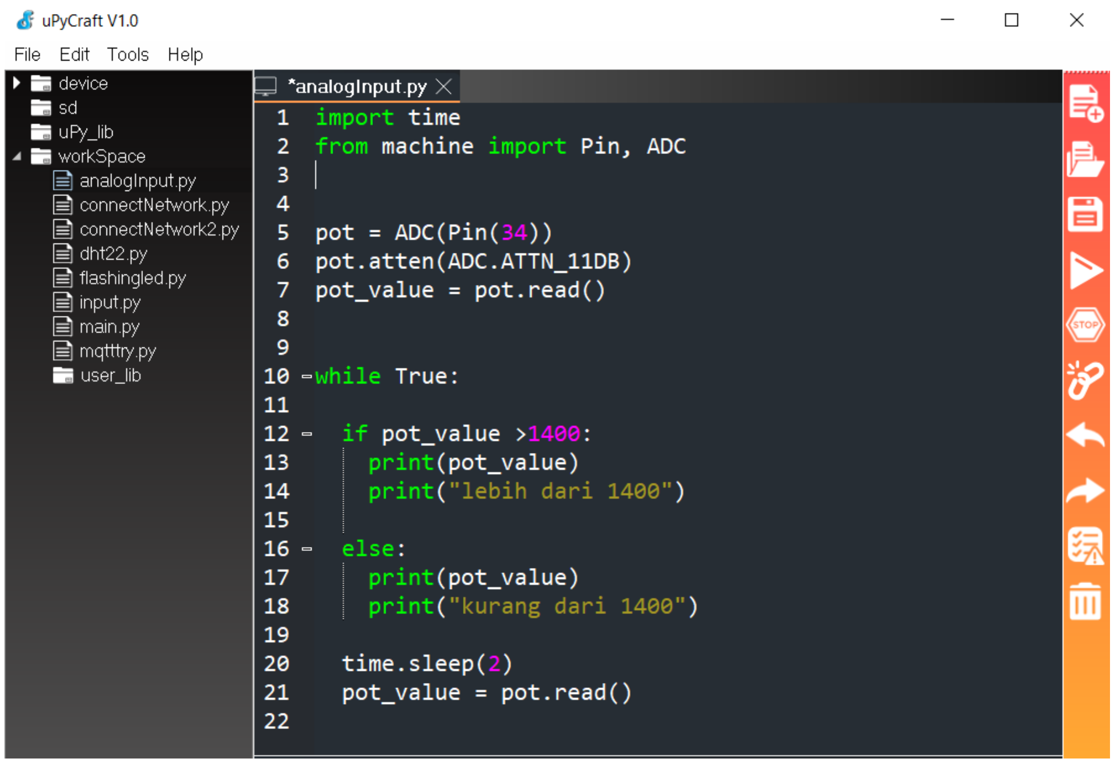
Instructions:

Analog reading can be seen on terminal

3. Do wiring as shown below.



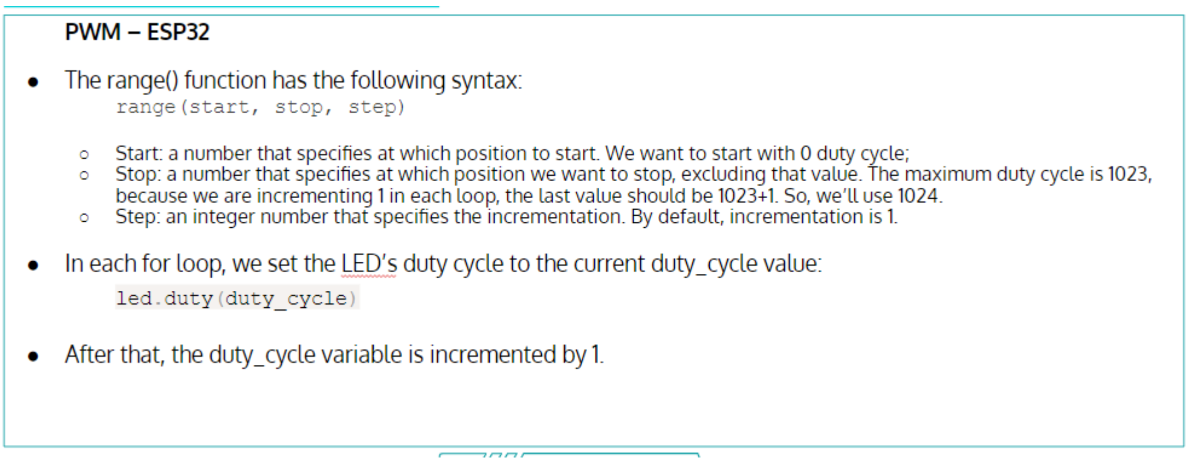
4. Write the program into uPyCraft IDE, in the main.py and download it into ESP32 board.



```
import time
from machine import Pin, ADC


pot = ADC(Pin(34))
pot.atten(ADC.ATTN_11DB)
pot_value = pot.read()


while True:

  if pot_value >1400:
    print(pot_value)
    print("lebih dari 1400")

  else:
    print(pot_value)
    print("kurang dari 1400")

  time.sleep(2)
  pot_value = pot.read()

```

ESP32 PWM

1. We continue with ESP32 PWM.



PWM – ESP32

- The range() function has the following syntax:

```
range(start, stop, step)
```

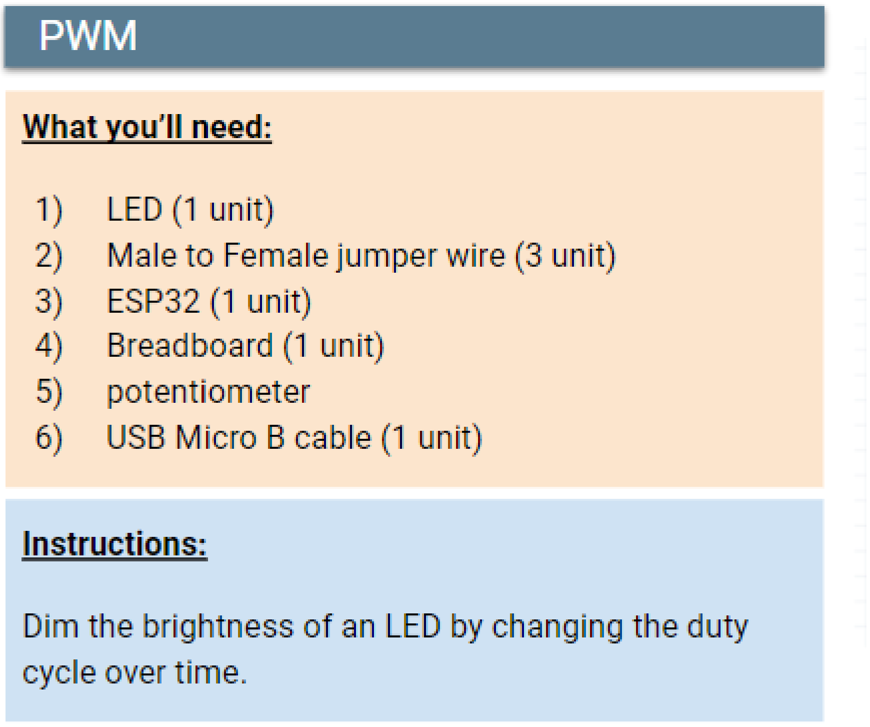
  - Start: a number that specifies at which position to start. We want to start with 0 duty cycle;
  - Stop: a number that specifies at which position we want to stop, excluding that value. The maximum duty cycle is 1023, because we are incrementing 1 in each loop, the last value should be 1023+1. So, we'll use 1024.
  - Step: an integer number that specifies the incrementation. By default, incrementation is 1.

- In each for loop, we set the LED's duty cycle to the current duty_cycle value:

```
led.duty(duty_cycle)
```

- After that, the duty_cycle variable is incremented by 1.

2. Prepare all the components needed.



PWM

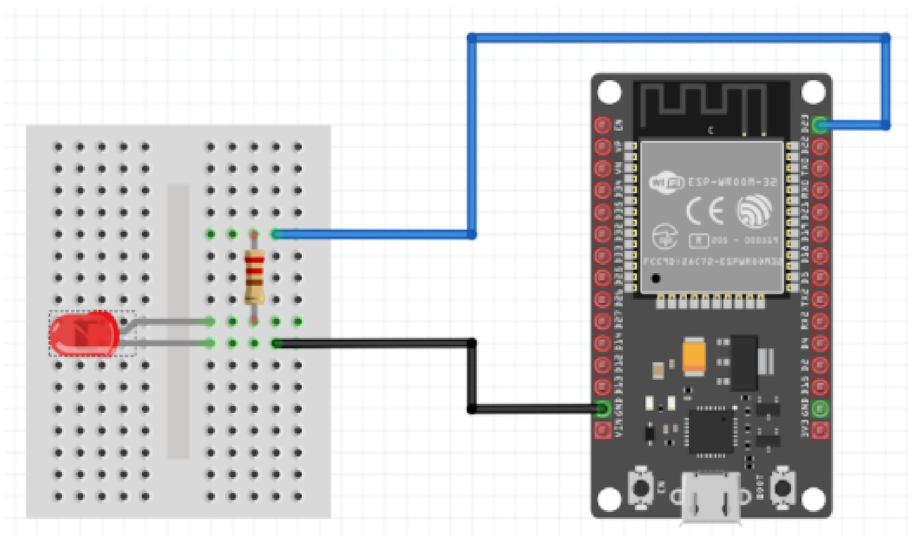
What you'll need:

1) LED (1 unit)
2) Male to Female jumper wire (3 unit)
3) ESP32 (1 unit)
4) Breadboard (1 unit)
5) potentiometer
6) USB Micro B cable (1 unit)

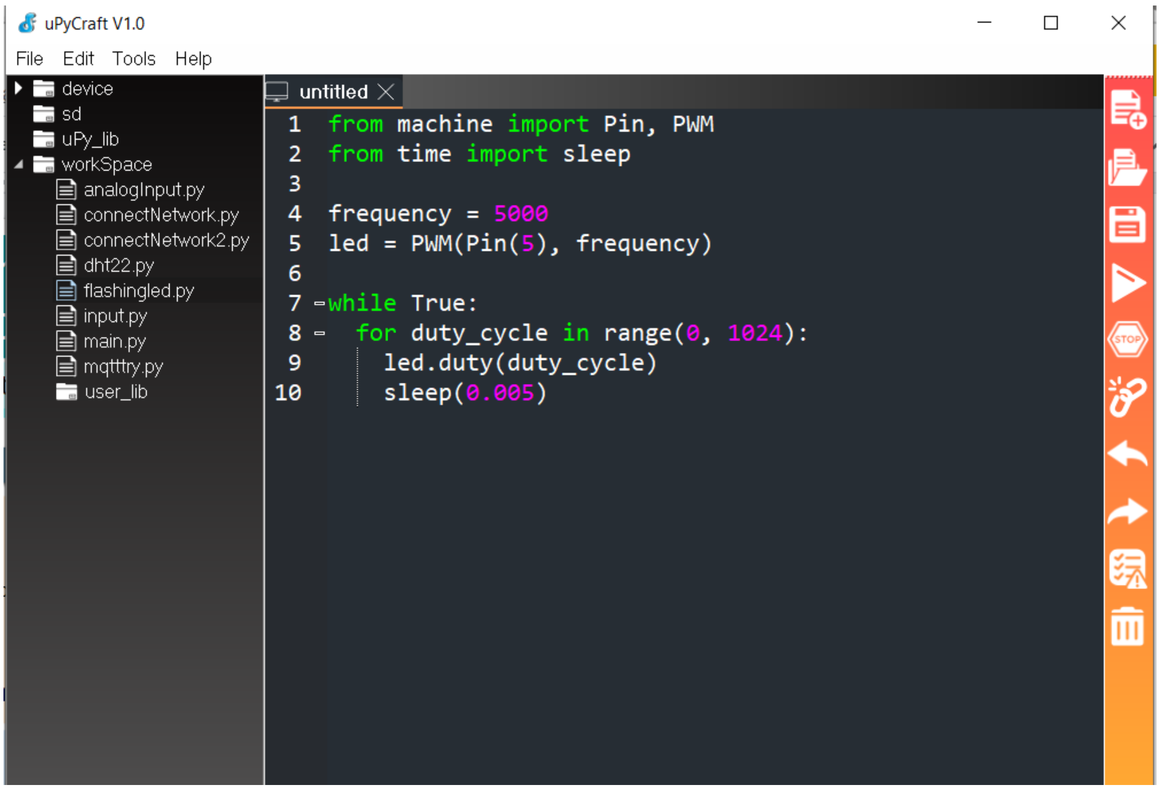
Instructions:

Dim the brightness of an LED by changing the duty cycle over time.

3. Do wiring as shown below.



4. Write the program into uPyCraft IDE, in the main.py and download it into ESP32 board.



```
from machine import Pin, PWM
from time import sleep

frequency = 5000
led = PWM(Pin(5), frequency)

while True:
  for duty_cycle in range(0, 1024):
    led.duty(duty_cycle)
    sleep(0.005)
```

References:

LAB 3: ESP32 PROGRAMMING

1. https://randomnerdtutorials.com/projects-esp32/