LAB 2:
BASIC MICROPYTHON PROGRAMMING

LAB 2:BASIC MICROPYTHON PROGRAMMING

Objective:
In this lab we are going to code using uPyCraft IDE. Throughout this lab, we will cover micropython syntax, element, comment, variable, data types and basic operators.

Steps:

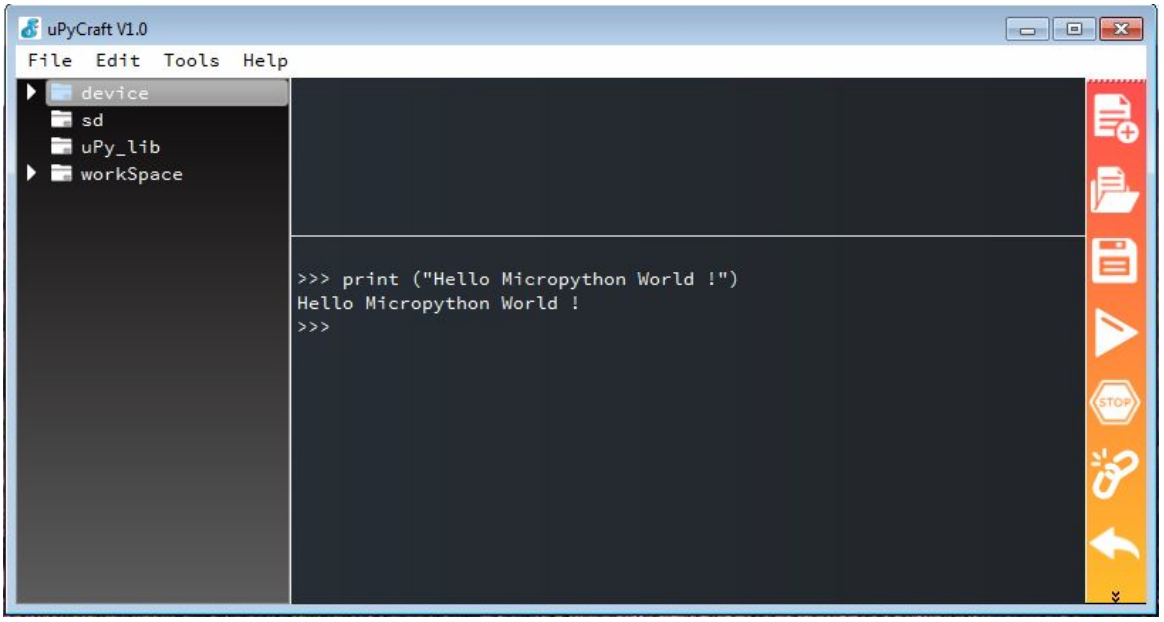
uPyCraft Familiarisation

1. Let's execute an embedded program in a microcontroller, we start by using python REPL (read, evaluate, print loop).In the Shell, try several operations to see how it works.

2. After having the MicroPython firmware installed on your board and having the board connected to your computer through an USB cable, follow the next steps:

   I. Go to Tools > Board and select the board you're using.
   II. Go to Tools > Port and select the com port your ESP is connected to.
   III. Press the Connect button to establish a serial communication with your board.

   IV. The >>> should appear in the Shell window after a successful connection with your board. You can type the print command to test if it's working:



```
>>> 3+5
8
>>> 6-5
1
>>> 8*9
72
>>> 20/10
2.0
>>>
```

```
>>>
>>>
>>> 2==5
False
>>> 4==4
True
>>> 69874 != 65
True
>>> 3>2
True
>>>
```

```
>>> a = 10
>>> b = 12
>>> c = 20.6
>>> text = 'abcdef'
>>> d = True
>>>
>>>
>>> type(a)
<class 'int'>
>>> type(b)
<class 'int'>
>>> type(b)
<class 'int'>
>>>
>>> type(c)
<class 'float'>
>>> type(text)
<class 'str'>
>>>
```

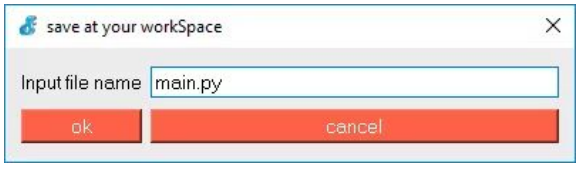
```
>>> number = 1
>>>
>>> while number <= 10:
...     print(number)
...     number = number +1
...
...
1
2
3
4
5
6
7
8
9
10
>>>
```

For this exercise, press Shift+Enter to execute.

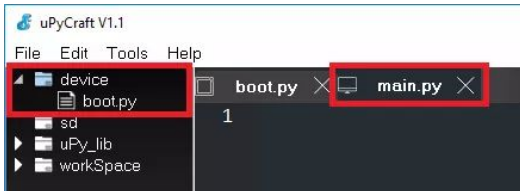
3. Creating the main.py file on your board.
   I. Press the “New file” button to create a new file.

   II. Press the “Save file” button to save the file in your computer.

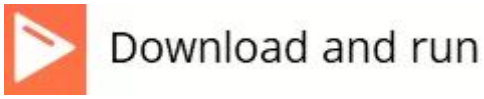
   III. A new window opens, name your file main.py and save it in your computer.



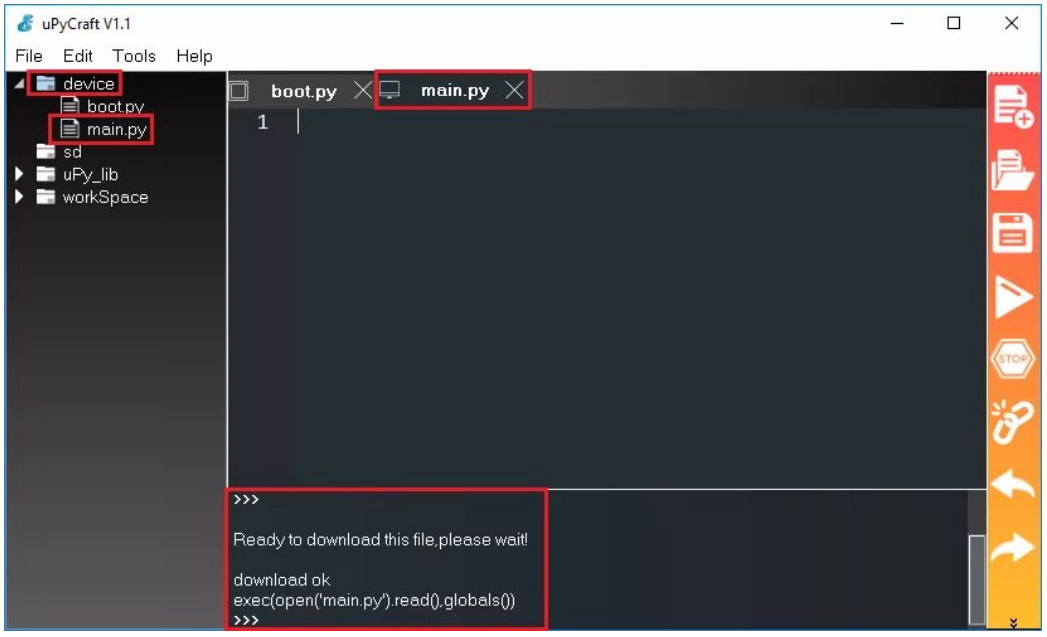
   IV. After that, you should see the following in your uPyCraft IDE (the boot.py file in your device and a new tab with the main.py file)



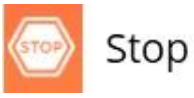
V. Click the “Download and run” button to upload the file to your ESP board.



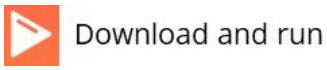
VI. The device directory should now load the main.py file. Your ESP has the file main.py stored.



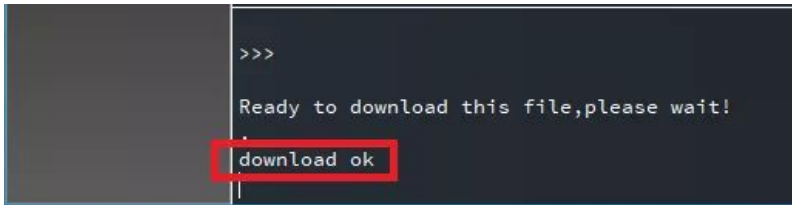
4. Uploading the blink LED script.
   I. Code to the Editor on the main.py file.
   II. Press the “Stop” button to stop any script from running in your board.



   III. Click the “Download and Run button” to upload the script to the ESP32 or ESP8266.

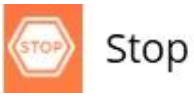


   IV. You should see a message saying “download ok” in the Shell window.

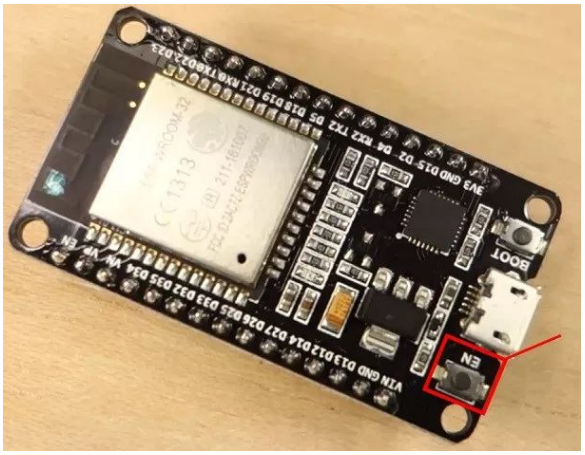


5. Testing the script

I. Press the “Stop” button



II. Press the on-board ESP32/ESP8266 EN (ENABLE) or RST (RESET) button to restart your board and run the script from the start:



III. If you’re using an ESP32, your Terminal messages should look something as shown in the following figure after a EN/RST button press:

6. print() and sleep()

print() and sleep()

- Import the **sleep** class from the **time** module.

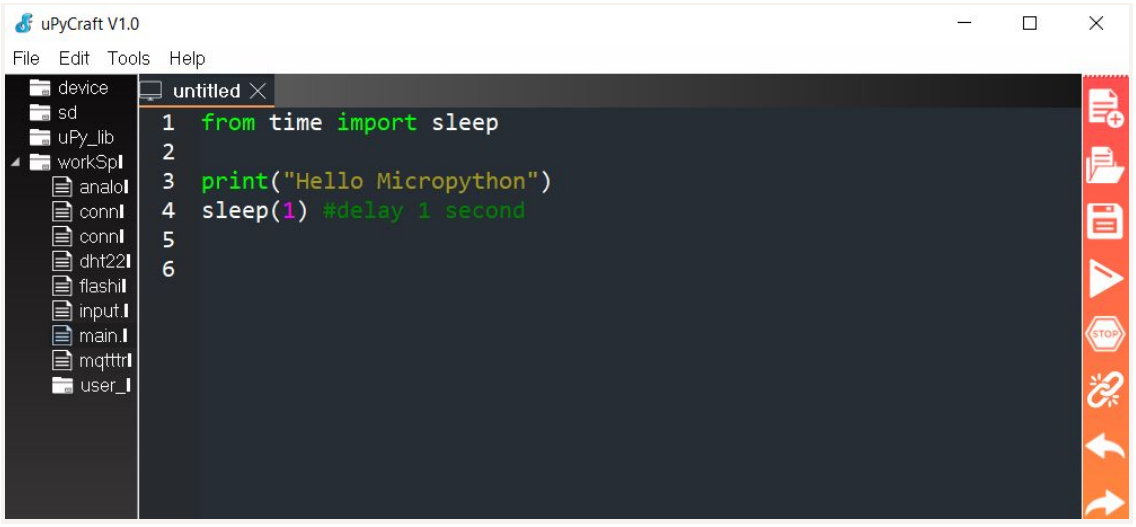
```
from time import sleep
```

- To delay, use sleep(time_in_second). For example:

```
sleep(1)
```

- To print any value use print(val). For example:

```
print("Hello Micropython")
```



LAB 2:BASIC MICROPYTHON PROGRAMMING

References:

1. https://randomnerdtutorials.com/flash-upload-micropython-firmware-esp32-esp8266/