Micropython with NodeMcu V3.0 – ESP8266

Note:
There are two files that are treated specially by the ESP8266 when it starts up: boot.py and main.py. The boot.py script is executed first (if it exists) and then once it completes the main.py script is executed. You can create these files yourself and populate them with the code that you want to run when the device starts up.

Important Links:
https://docs.micropython.org/en/latest/esp8266/esp8266/tutorial/intro.html
https://docs.micropython.org/en/latest/esp8266/quickref.html#general-board-control
https://docs.micropython.org/en/latest/esp8266/tutorial/filesystem.html
https://learn.adafruit.com/micropython-basics-load-files-and-run-code/overview
http://docs.micropython.org/en/v1.9.4/esp8266/esp8266/tutorial/intro.html

1-Deploying the firmware:
```
sudo pip install esptool
```

to see the connected devices and the ports
```
ls /dev/tty*
```

to deploy the new frimwork, go to the directory of the frimwork ‘esp8266-20180511-v1.9.4.bin’, open the terminal and excute
```
sudo esptool.py --port /dev/ttyUSB0 erase_flash
sudo esptool.py --port /dev/ttyUSB0 --baud 115200 write_flash --flash_size=detect 0 esp8266-20180511-v1.9.4.bin
```

2-Connect to the board through linux terminal:
to connect to the board from linux terminal, first install ‘screen’
```
sudo apt-get install screen
```

then connect to the right port, in my case it’s ttyUSB0
```
sudo screen /dev/ttyUSB0 115200
```

Setup the board for the first time:

setup the network:
```
import network
sta_if = network.WLAN(network.STA_IF)
sta_if.active(True)
sta_if.connect("Wifi-Username", "Wifi-Password")
sta_if.isconnected()
```

-don’t forget to change the 'Wifi-Username' and ‘'Wifi-Password' to your wifinetwork username & password.

setup the WebREPL (web browser interactive prompt):
```
import webrepl_setup
```

you will be asked to enter a password for the WebREPl, after that choose not to reboot the board, then excute
```
import webrepl
webrepl.start()
```

you will get something like this

```
WebREPL daemon started on ws://192.168.4.1:8266
WebREPL daemon started on ws://192.168.1.100:8266
Started webrepl in normal mode
```

at the second line you can see ‘ws://192.168.1.100:8266’ , you will need this address in the WebREPl webpage.

Connecting through WebREPl:
-open the WebREPl webpage ‘webrepl.html’ and use the address ‘ws://192.168.1.100:8266’ , then click connecte.
-you will be asked to enter the password, which you used in the previous step.
-now you are conncted to the board python consol through the wifi.

Final step save all your work in the boot.py file:
-If you restarted the board the network information will be erased, so you need to setup everything to be done in the boot file.
-in the PC make file
-in the WebREPl webpage, type in ‘boot.py’ , then click ‘Get from device’

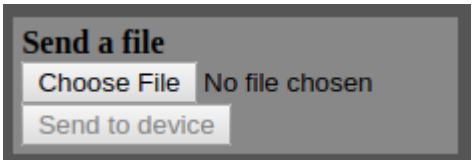
-you will get the file ‘boot.py’ form the board, open the file and overwrite the content to be like this

```
# This file is executed on every boot (including wake-boot from deepsleep)
#import esp
#esp.osdebug(None)
import gc
import webrepl

def do_connect(): #script to connect to wifi
    import network
    sta_if = network.WLAN(network.STA_IF)
    if not sta_if.isconnected():
        print('connecting to network...')
        sta_if.active(True)
        sta_if.connect('Wifi-Username', 'Wifi-Password')
        while not sta_if.isconnected():
            pass
    print('network config:', sta_if.ifconfig())

do_connect()
webrepl.start()
gc.collect()
```

-don’t forget to change the 'Wifi-Username' and ‘'Wifi-Password' to your wifinetwork username & password.
-in the WebREPl webpage, click ‘Choose File’ and choose the new ‘boot.py’ and click ‘Send to device’.



-now you are done with the first time setup, you can reboot the board and connect to it through the linux terminal or the WebREPl webpage.

3-Start using the board:

-creat a file ‘main.py’ and write the code you want to run after booting the board.
-in the WebREPl webpage, click ‘Choose File’ and choose the ‘main.py’ file and click ‘Send to device’.
-now the board will keep looping in the code written in the ‘main.py’ file.

#Note:
To scan the network for the wifi connections available, type in

```
import network
network.WLAN(network.STA_IF).scan()
```

#Note:
Take in account that the comments in the python code, take size in the .py file, and the file will be uploaded directly to the board, as the code will not be compiled as in c-language for example, so if you have any problem in the files size in the board, delete the comments.

4-Load Files & Run Code from linux terminal without the WebREPl webpage:

useful links:
https://learn.adafruit.com/micropython-basics-load-files-and-run-code/install-ampy
https://learn.adafruit.com/micropython-basics-load-files-and-run-code/file-operations
https://github.com/adafruit/ampy/issues/19

Install ampy:
-in linux command

```
pip install adafruit-ampy
```

Disable ESP8266 Debug Output:
-in micropython consol

```
import esp
esp.osdebug(None)
```

-It is highly recommended to add the above two lines to the board's boot.py so debug output is disabled permanently. If you don't make this change you'll need to manually disable debug output every time you reset the board!

Fixing ampy:
-in linux go to the directory `/usr/local/lib/python2.7/dist-packages/ampy` edit the ‘pyboard.py’
-go to line with this code

```
while n > 0:
    self.serial.read(n)
    n = self.serial.inWaiting()
```

then add ‘time.sleep(2)’

```
while n > 0:
    self.serial.read(n)
    n = self.serial.inWaiting()
time.sleep(2)
```

Start using ampy:

-run code from PC on the board, without loading the file to the board, in linux command excute

```
sudo ampy --port /dev/ttyUSB0 run test.py
```

-load file to the board, in linux comman excute

```
sudo ampy --port /dev/ttyUSB0 put main.py
```

-ampy has many features, look at the
https://learn.adafruit.com/micropython-basics-load-files-and-run-code/file-operations
and you will find many interesting features.

#Note:
-you cannot use ampy while connecting to the board through linux terminal or WebREPl webpage, so disconect the board and reconnect the board before using ampy.
-make sure to restart the board before using ampy.
-sometimes restarting the board will not work, so you should unplug the board and replug it again, to make it work right.
-in my board, when defining a GPIO as digital input, the GPIO16 and GPIO15 are by default pulled-down and cannot be pulled-up, but the rest of the GPIOs are by default pulled-up and cannot be pulled-down.