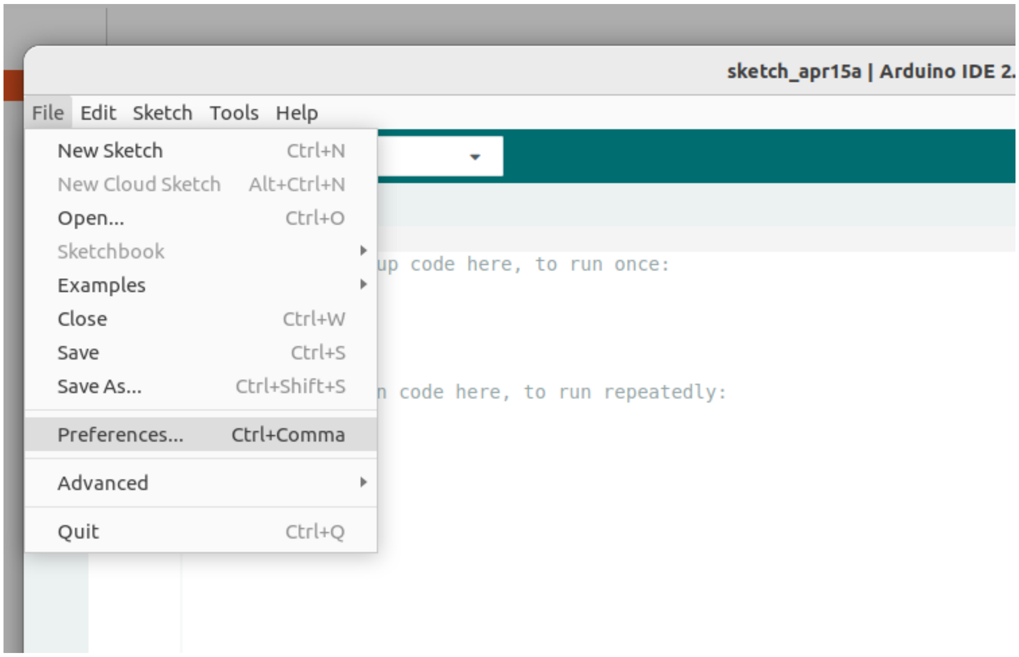
NodeMCU + STM32F4

In the final project, you need to connect STM32 and NodeMCU together. Given that you have not used Node MCU, there is a short tutorial.

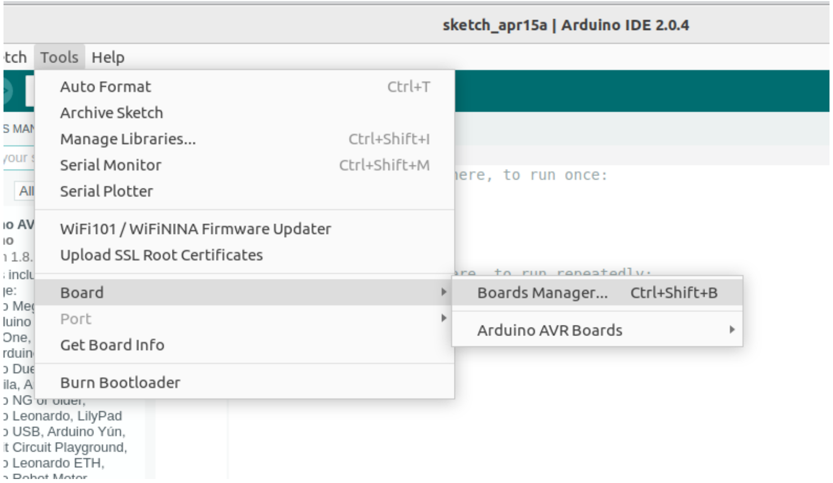
There are several methods you can use to program NodeMCU for example, PlatformIO, ESP-IDF or Arudino IDE. In this tutorial, we will be using Arudio IDE.

1. Download software from here: https://www.arduino.cc/en/software then install
2. Open the IDE, then go File->Preferences,

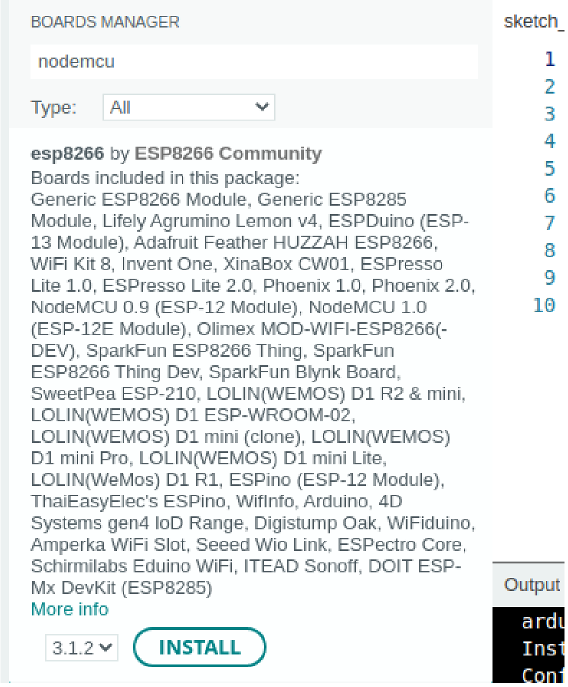


```
up code here, to run once:
n code here, to run repeatedly:
```

3. Enter http://arduino.esp8266.com/stable/package_esp8266com_index.json into the “Additional Boards Manager URLs” field as shown in the figure below. Then, click the “OK” button:
4. Click Tools -> Board-> Board Manager


sketch_apr15a | Arduino IDE 2.0.4
tch
Tools
Help
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Board
Port
Get Board Info
Burn Bootloader
Boards Manager... Ctrl+Shift+B
Arduino AVR Boards
here, to run once:
re, to run repeatedly:

5. Search nodemcu in the "Board Manager"

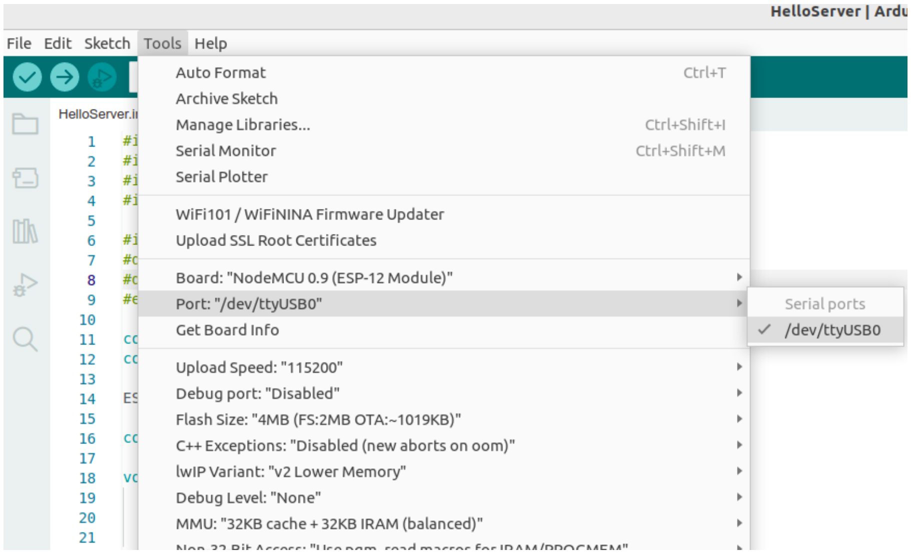


esp8266 by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, WiFi Kit 8, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 ESP-WROOM-02, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285)
More info

Hit install

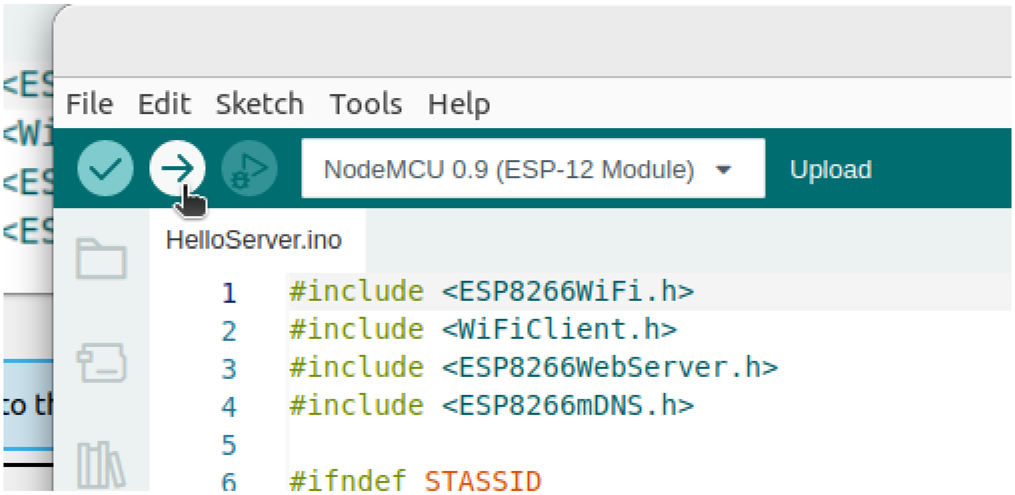
6. Go to Tool-> Board -> NodeMCU 0.9 (ESP-12 Module)
7. You can choose and play around with a few examples using NodeMCU, for example, you can go to File->Examples->ESP8266WebServer->HelloServer
8. If you want to test this example, turn on wifi-sharing on your cell phone, then change "your-ssid" and "your-password" to your SSID displayed on your cell phone
9. Choose to program to your board in the serial terminal

   Tool -> Port -> "select your serial port here"



Board: "NodeMCU 0.9 (ESP-12 Module)"
Port: "/dev/ttyUSB0"
✓ /dev/ttyUSB0
Upload Speed: "115200"
Debug port: "Disabled"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
C++ Exceptions: "Disabled (new aborts on oom)"
lwIP Variant: "v2 Lower Memory"
Debug Level: "None"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"

10. Then you can press Upload, which will compile and upload the code to your board



```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

#ifndef STASSID
```

Communication between STM32 and NodeMCU

If you would like to make STM32 and NodeMCU talk to each other, you need to set up some kind of communication. You can choose whichever method you want to use, one of the easiest methods is using UART.

1. ESP8266 has a hardwire serial through UART-to-USB similar to STM32, so you need to setup other pins to operate as serial. One of the way is to use a hardware serial, which emulate UART communication through GPIO. You may implement this yourself, but there is a library you can use.
   To install the library go to Tools->Manage Libraries…

2. In the search box, search espsoftwareserial
3. Hit install

4. Implement a simple program to use Software Serial

```
#include <SoftwareSerial.h>
EspSoftwareSerial::UART testSerial;
void setup() {
   testSerial.begin(115200, EspSoftwareSerial::SWSERIAL_8N1, D7, D8, false, 95, 11);
}
void loop() {
 // Echo
   while (testSerial.available() > 0) {
       testSerial.write(testSerial.read());
       yield();
   }
}
```

5. Implement a simple program to send data/receive data from STM32 using UART. Note that you will want to use USART1/6, but not UART2 as UART2 is already connected to UART-to-USB.
6. Connect ESP8266 Ground to STM32 Ground, ESP8266-RX (D7 in the example above) to STM32-TX (PA9 for a default USART1 in STM32CubeIDE) and ESP8266-TX (D8 in the example above) to STM32-RX (PA10 for a default USART1 in STM32CubeIDE). If you use