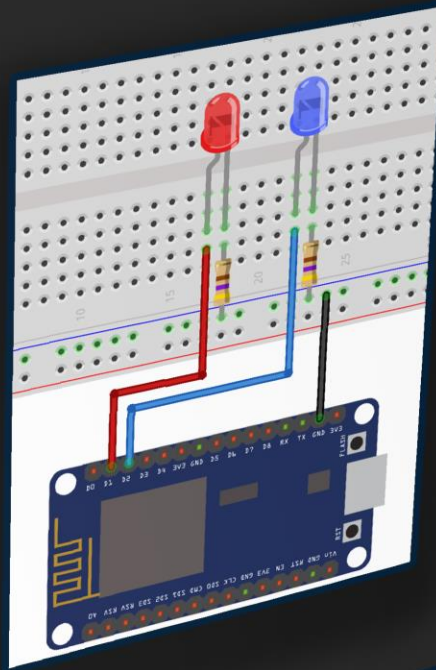


ESP8266 WEB SERVER WITH ARDUINO IDE

# ESP8266 WEB SERVER WITH ARDUINO IDE



Written by Rui Santos

# ESP8266 Web Server with Arduino IDE

---

**Hello and thank you for downloading this project eBook!**

This quick eBook is my step-by-step guide designed to help you build a web server with a WiFi module called ESP8266.

## **About the ESP8266**

The ESP8266 is a \$4 (up to \$10) WiFi module with an ARM processor that is great for home automation/internet of things applications.

So what can you do with this low cost module?

You can create a web server, send HTTP requests, control outputs, read inputs and interrupts, send emails, post tweets, etc.

### ***ESP8266 specifications***

- 802.11 b/g/n protocol
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Built-in low-power 32-bit CPU
- SDIO 2.0, SPI, UART

## Finding Your ESP8266

The ESP8266 comes in a wide variety of versions. The ESP-12E or often called ESP-12E NodeMCU Kit is currently the most practical version and that's the module we'll be using most throughout this eBook.

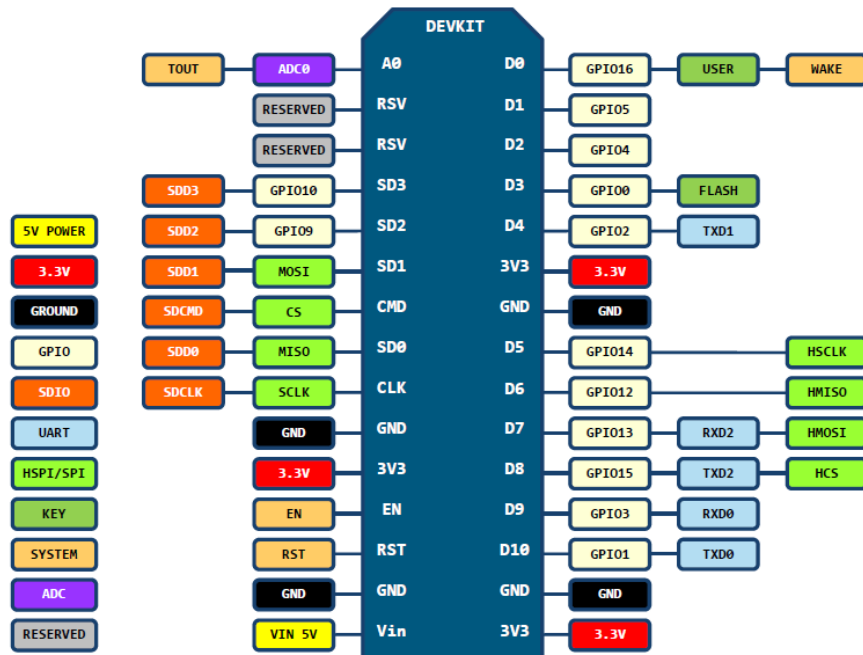
This project was tested with ESP-01, ESP-07, ESP-12 and ESP-12E. So you can make all the projects presented in this eBook using any of those boards.

I highly recommend using the ESP8266-12E NodeMCU Kit, the one that has built-in programmer. The built-in programmer makes it easy to prototype and upload your programs. [Click here to buy this module on eBay.](#)



# ESP-12E NodeMCU Kit Pinout

Here's a quick overview of the ESP-12E NodeMCU Kit pinout:



*D0(GPI016) can only be used as gpio read/write, no interrupt supported, no pwm/i2c/ow supported.*

# ESP8266 with Arduino IDE

---

In this section you're going to download, install and prepare your Arduino IDE to work with the ESP8266. This means you can program your ESP using the friendly Arduino programming language.

## What's the Arduino IDE?

The Arduino IDE is an open-source software that makes it easy to write code and upload it to the Arduino board. [This GitHub repository](#) added support for the ESP board to integrate with the Arduino IDE.

The Arduino IDE is a multiplatform software, which means that it runs on Windows, Mac OS X or Linux (it was created in JAVA).

### **Requirements**

You need to have JAVA installed in your computer. If you don't have, go to this website: <http://java.com/download>, download and install the latest version.

## Downloading Arduino IDE


To download the Arduino IDE, visit the following URL: <https://www.arduino.cc/en/Main/Software>.

Then, select your operating system and download the software (as shown below).

DOWNLOAD


ENGLISH ▾

## Download the Arduino Software



**ARDUINO 1.8.0**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

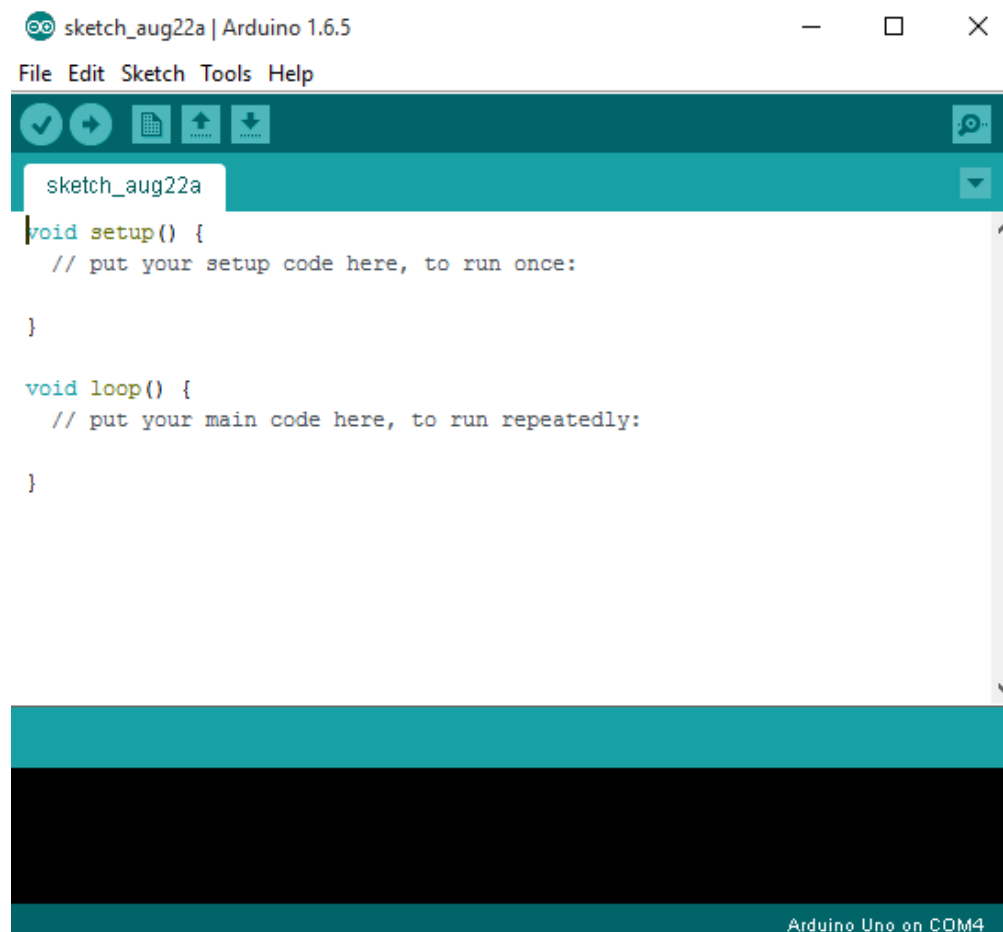
- Windows Installer**
- Windows ZIP file for non admin install**
- Windows app** 
- Mac OS X 10.7 Lion or newer**
- Linux 32 bits**
- Linux 64 bits**
- Linux ARM**
- [Release Notes](#)
- [Source Code](#)
- [Checksums \(sha512\)](#)

## Installing Arduino IDE

Grab the file you’ve just downloaded named “arduino-(...).zip”. Run that file and follow the installation wizard that shows on your screen. Open the Arduino IDE application file (see figure below).

Name	Date modified	Type	Size
dist	13/08/2015 20:53	File folder	
drivers	13/08/2015 20:53	File folder	
examples	13/08/2015 20:54	File folder	
hardware	13/08/2015 20:54	File folder	
java	13/08/2015 20:57	File folder	
lib	13/08/2015 20:59	File folder	
libraries	13/08/2015 21:00	File folder	
reference	13/08/2015 21:03	File folder	
tools	13/08/2015 21:03	File folder	
<b>arduino</b>	<b>14/08/2015 17:42</b>	<b>Application</b>	<b>393 KB</b>
arduino.l4j	13/08/2015 20:53	Configuration sett...	1 KB
arduino_debug	13/08/2015 20:53	Application	390 KB
arduino_debug.l4j	13/08/2015 20:53	Configuration sett...	1 KB
libusb0.dll	13/08/2015 20:53	Application extens...	43 KB
msvcp100.dll	13/08/2015 20:53	Application extens...	412 KB
msvcr100.dll	13/08/2015 20:53	Application extens...	753 KB
revisions	13/08/2015 20:53	Text Document	66 KB

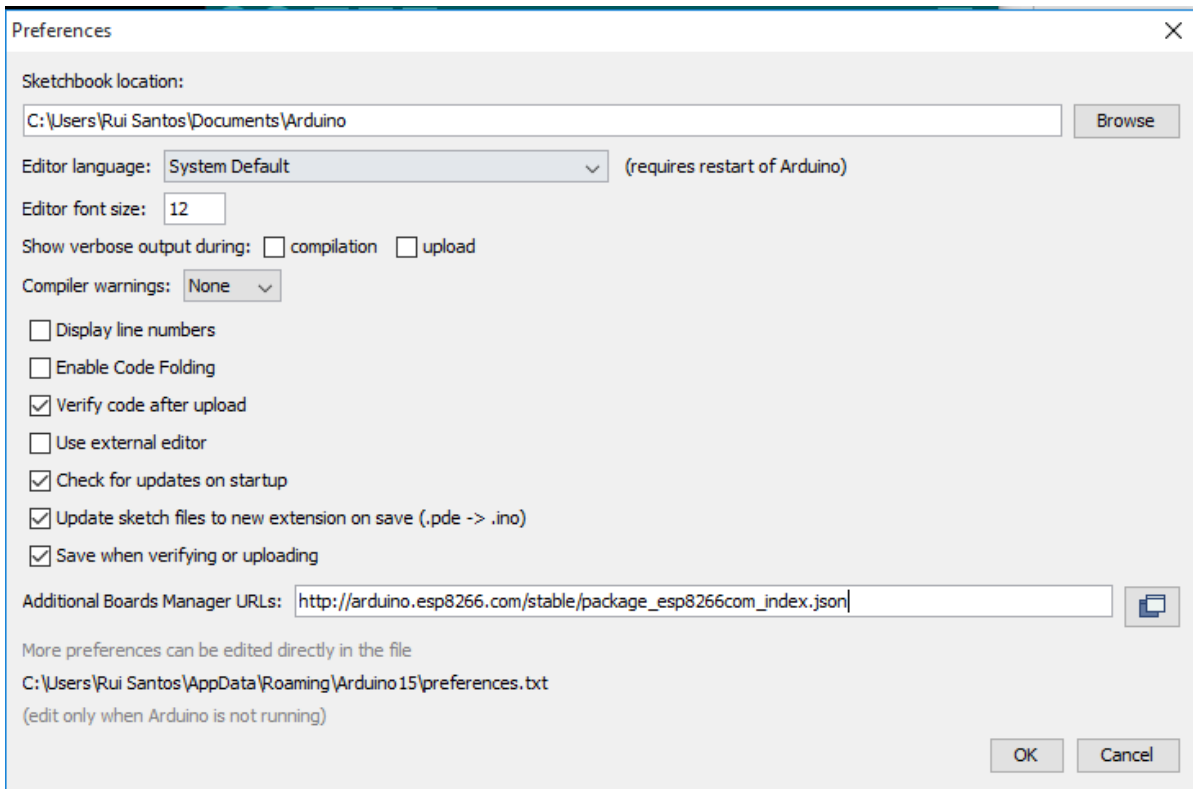
When the Arduino IDE first opens, this is what you should see:



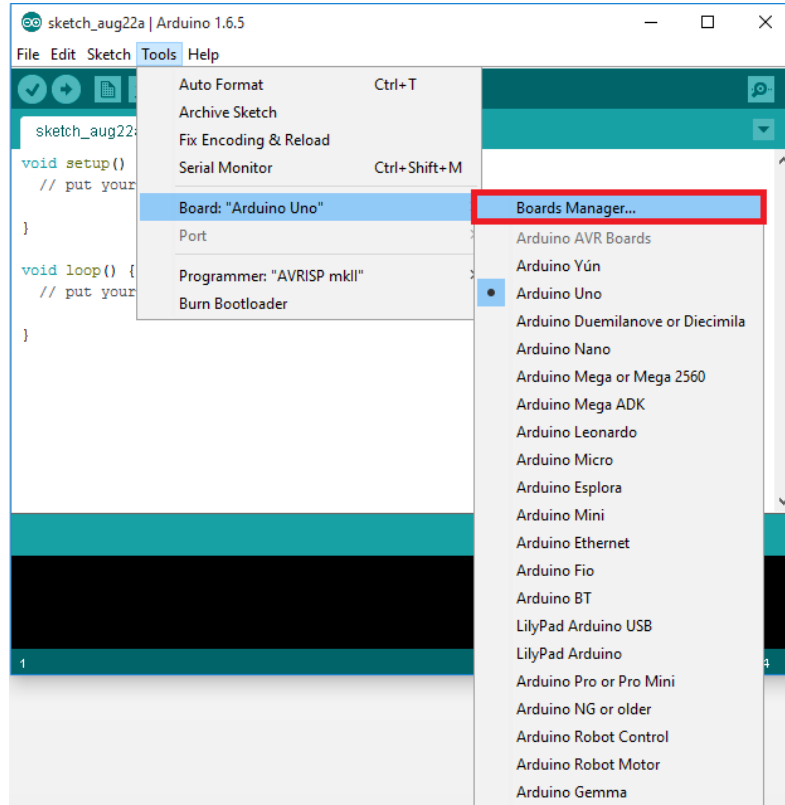
## Installing ESP8266 Board

To install the ESP8266 board in your Arduino IDE, follow these next instructions:

1. Open the preferences window from the Arduino IDE. Go to **File > Preferences**
2. Enter [http://arduino.esp8266.com/stable/package\\_esp8266com\\_in\\_dex.json](http://arduino.esp8266.com/stable/package_esp8266com_in_dex.json) into **Additional Board Manager URLs** field and press the “OK” button

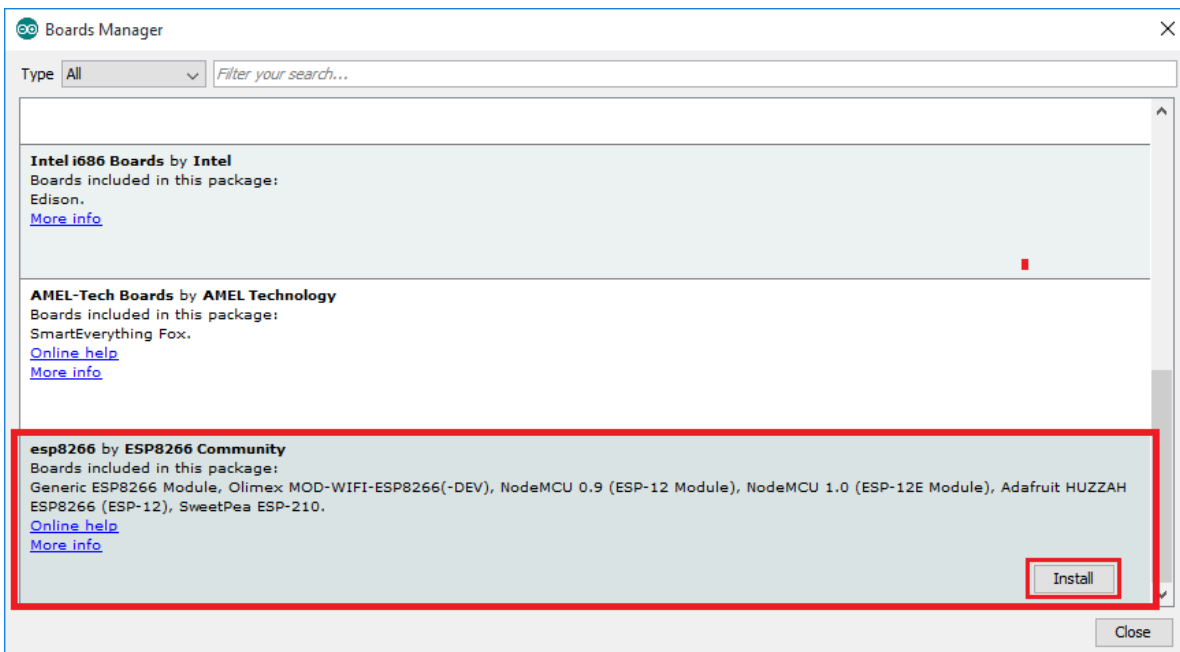


### 3. Go to **Tools > Board > Boards Manager...**



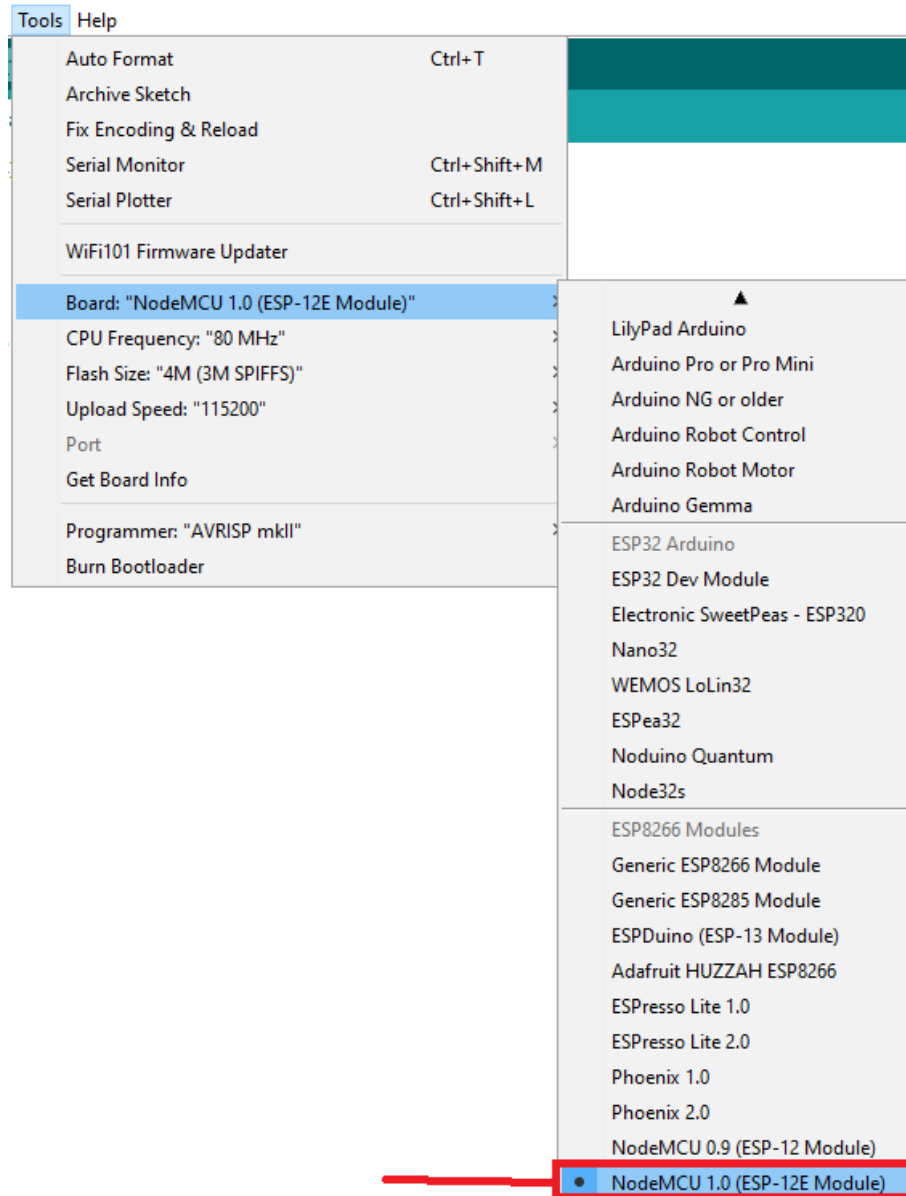


4. Scroll down, select the ESP8266 board menu and Install “**esp8266 by ESP8266 Community**”



5. Open the Arduino **Tools** menu

## 6. Select **Board** > **NodeMCU 1.0 (ESP-12E Module)**



7. Finally, re-open your Arduino IDE to ensure that it launches with the new boards installed

# Blinking LED with Arduino IDE

---

In this section, you're going to design a simple circuit to blink an LED with the ESP using Arduino IDE.

Why do we always blink an LED first? That's a great question! If you can blink an LED you can pretty much say that you can turn any electronic device on or off.

## Writing Your Arduino Sketch

The sketch for blinking an LED is very simple. You can find it in the link below:

### SOURCE CODE

<https://gist.github.com/RuiSantosdotme/789861277da9680e9cfb>

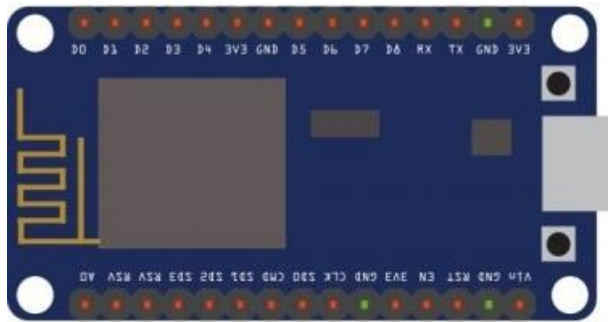
```
int pin = 2;

void setup() {
  // initialize GPIO 2 as an output.
  pinMode(pin, OUTPUT);
}

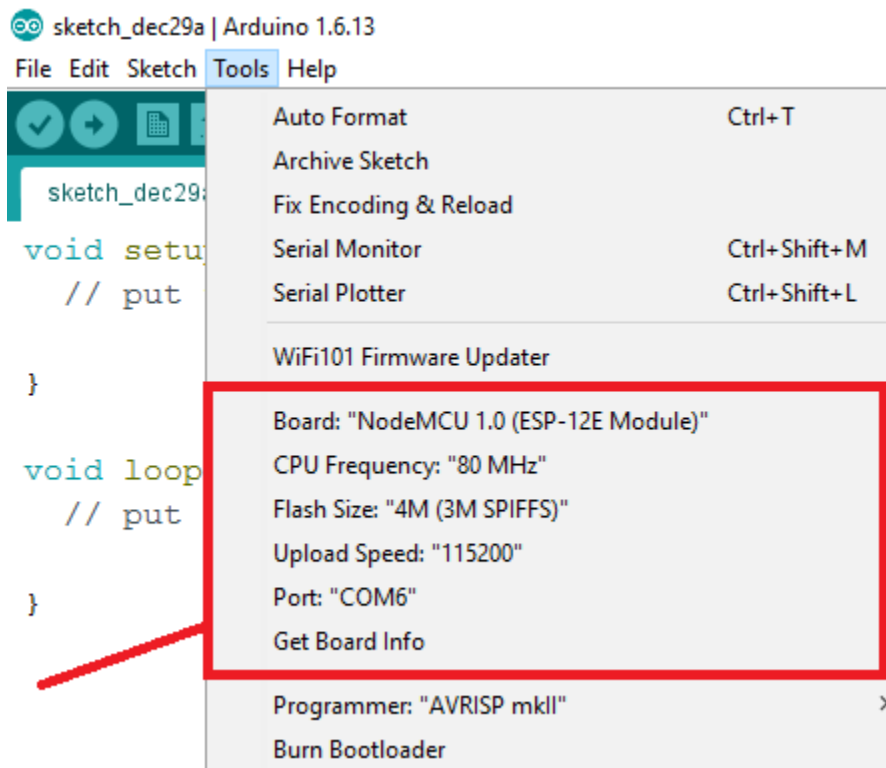
// the loop function runs over and over again forever
void loop() {
  digitalWrite(pin, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(pin, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

# Uploading Code to ESP8266

Upload code to your ESP-12E NodeMCU Kit is very simple, since it has built-in programmer. You plug your board to your computer and you don't need to make any additional connections.

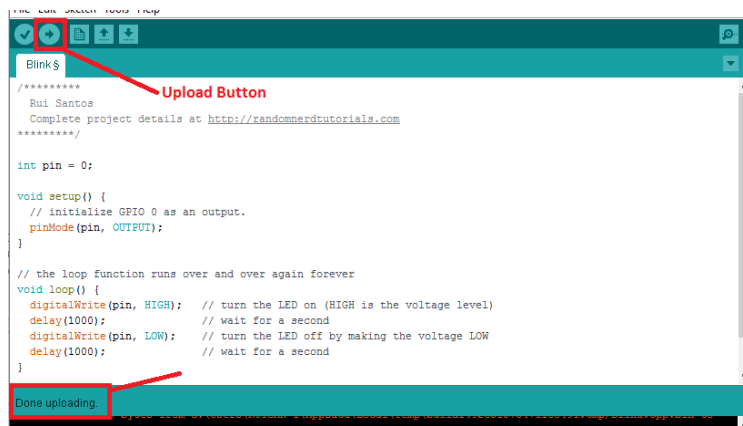


Look at the **Tools** menu, select Board “**NodeMCU 1.0 (ESP-12E Module)**” and all the configurations, by default, should look like this:



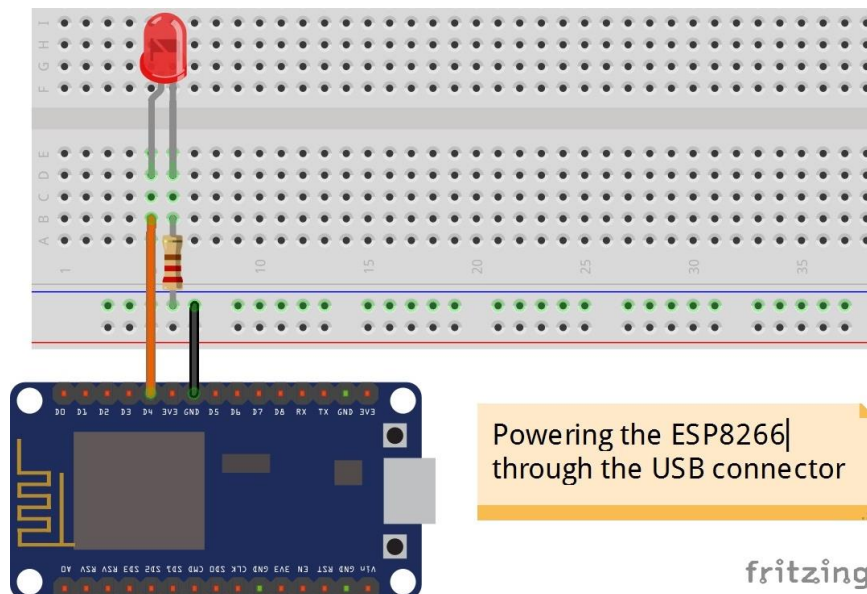
**Important:** your COM port is very likely to be different from the preceding screenshot (Port: “COM6”). That’s fine, because it doesn’t interfere with anything. On the other hand, all the other configurations should look exactly like mine.

After checking the configurations, click the **“Upload Button”** in the Arduino IDE and wait a few seconds until you see the message **“Done uploading.”** in the bottom left corner.

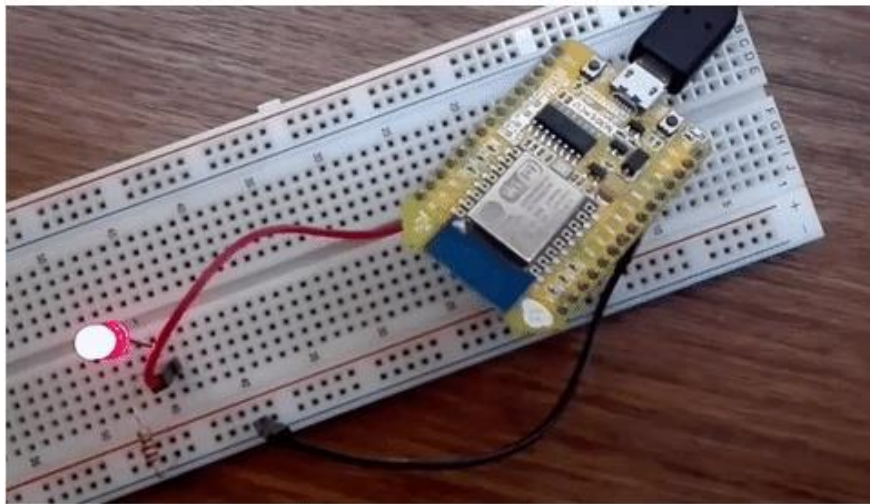


## Final ESP-12E circuit

Connect an LED and a 220 Ohm resistor to your ESP D4 (GPIO 2).



Restart your ESP8266. Congratulations, you've made it! Your LED should be blinking every 1 second!



# ESP8266 Web Server

---

In this project, you're going to build a web server that controls two LEDs. This is just an example the idea is to replace those LEDs with a relay to control any electronic devices that you want.

## Writing Your Arduino Sketch

Let's create a web server that controls two outputs (GPIO 5 and GPIO 4).

Copy the sketch below to your Arduino IDE. Replace the SSID and password with your own credentials.

After modifying my sketch upload it to your ESP8266 (If you can't upload code to your ESP8266, [read this troubleshooting guide](#)).

### SOURCE CODE

<https://gist.github.com/RuiSantosdotme/802869a70fa9e83ec78826d38ca89c4c>

```
/*  
  Rui Santos  
  Complete project details at http://randomnerdtutorials.com  
*/  
  
#include <ESP8266WiFi.h>  
#include <WiFiClient.h>  
#include <ESP8266WebServer.h>  
#include <ESP8266mDNS.h>  
  
MDNSResponder mdns;
```

```

// Replace with your network credentials
const char* ssid = "YOUR_SSID";
const char* password = "YOUR_PASSWORD";

ESP8266WebServer server(80);

String webPage = "";

int gpio5_pin = 5;
int gpio4_pin = 4;

void setup(void){
  webPage += "<h1>ESP8266 Web Server</h1><p>Socket #1 <a
href=\"socket10n\"><button>ON</button></a>&nbsp;<a
href=\"socket10ff\"><button>OFF</button></a></p>";
  webPage += "<p>Socket #2 <a
href=\"socket20n\"><button>ON</button></a>&nbsp;<a
href=\"socket20ff\"><button>OFF</button></a></p>";

  // preparing GPIOs
  pinMode(gpio5_pin, OUTPUT);
  digitalWrite(gpio5_pin, LOW);
  pinMode(gpio4_pin, OUTPUT);
  digitalWrite(gpio4_pin, LOW);

  delay(1000);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");

```



```

Serial.println(ssid);
Serial.print("IP address: ");
Serial.println(WiFi.localIP());

if (mdns.begin("esp8266", WiFi.localIP())) {
    Serial.println("MDNS responder started");
}

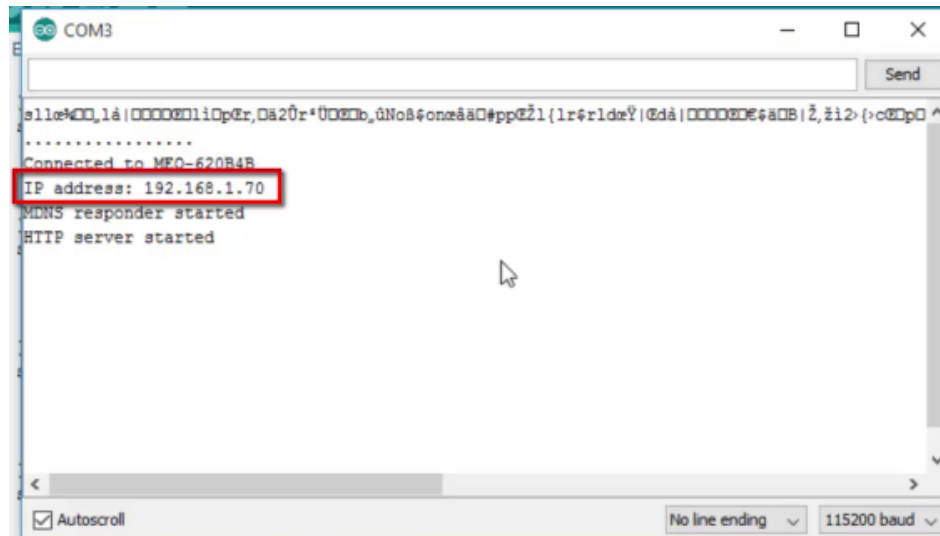
server.on("/", [](){
    server.send(200, "text/html", webPage);
});
server.on("/socket10n", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio5_pin, HIGH);
    delay(1000);
});
server.on("/socket10ff", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio5_pin, LOW);
    delay(1000);
});
server.on("/socket20n", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio4_pin, HIGH);
    delay(1000);
});
server.on("/socket20ff", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio4_pin, LOW);
    delay(1000);
});
server.begin();
Serial.println("HTTP server started");
}

void loop(void){
    server.handleClient();
}

```

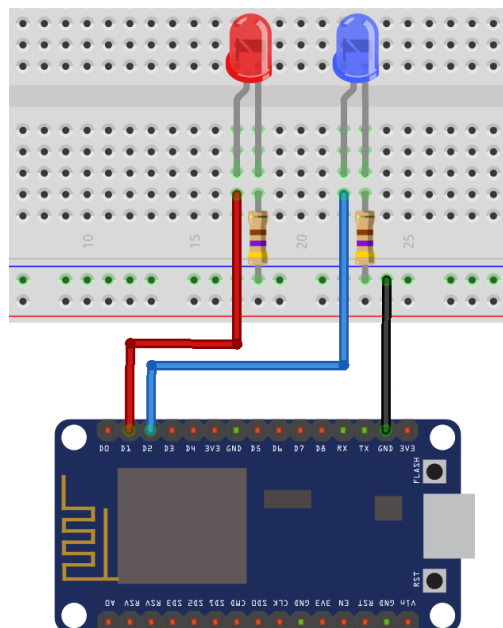
# ESP8266 IP Address

Open the Arduino serial monitor at a baud rate of 115200. After a few seconds, your IP address should appear. In my case it's 192.168.1.70.



# Final Circuit

Now follow the schematics below to create your web server to control two LEDs.



## Demonstration

For the final demonstration open any browser from a device that is connected to the same router that your ESP is. Then type the IP address and click Enter!



---

**Congratulations for completing this project!**

# Download Other RNT Products

---

[Random Nerd Tutorials](#) is an online resource with electronics projects, tutorials and reviews.

Creating and posting new projects takes a lot of time. At this moment, Random Nerd Tutorials has nearly 150 free blog posts with complete tutorials using open-source hardware that anyone can read, remix and apply to their own projects: <http://randomnerdtutorials.com>

To keep free tutorials coming, there's also paid content or as I like to call "premium content".

To support Random Nerd Tutorials you can [download premium content here](#). If you enjoyed this eBook make sure you [check all the others](#).

Thanks for taking the time to read my work!

Good luck with all your projects,

-Rui Santos

[P.S. Click here for more courses and eBooks like this one.](#)

[Click here to Download other Courses and eBooks](#)

<http://randomnerdtutorials.com/products>