

ITBA

6/9/2019



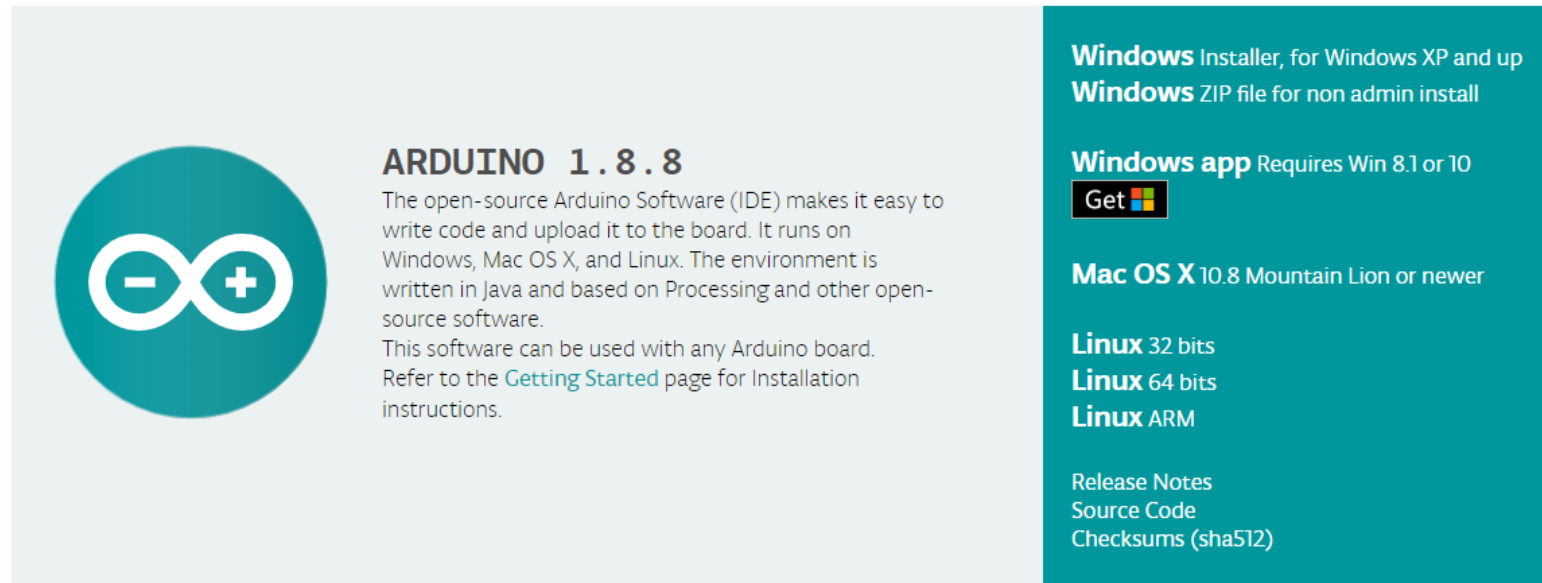
IoT Arduino

Puesta en Marcha

1 – Descargar el entorno de desarrollo de Arduino.

LINK: <https://www.arduino.cc/en/Main/Software>

Download the Arduino IDE



The screenshot shows the Arduino IDE download page. On the left, there is a teal circle with a white infinity symbol containing a minus and a plus sign. To its right, the text reads: **ARDUINO 1.8.8**, followed by a description of the IDE as open-source software that runs on Windows, Mac OS X, and Linux. Below this, it states that the software can be used with any Arduino board and refers to the 'Getting Started' page for installation instructions. On the right side of the page, there are links for downloading the IDE for different operating systems: Windows (installer and ZIP file), Windows app (requiring Win 8.1 or 10), Mac OS X (10.8 Mountain Lion or newer), and Linux (32 bits, 64 bits, and ARM). There are also links for Release Notes, Source Code, and Checksums (sha512).

2 - Bajar la versión para el sistema operativo utilizado

3 - Las instrucciones de instalación las podrá encontrar en este link:

<https://www.arduino.cc/en/Guide/Windows>

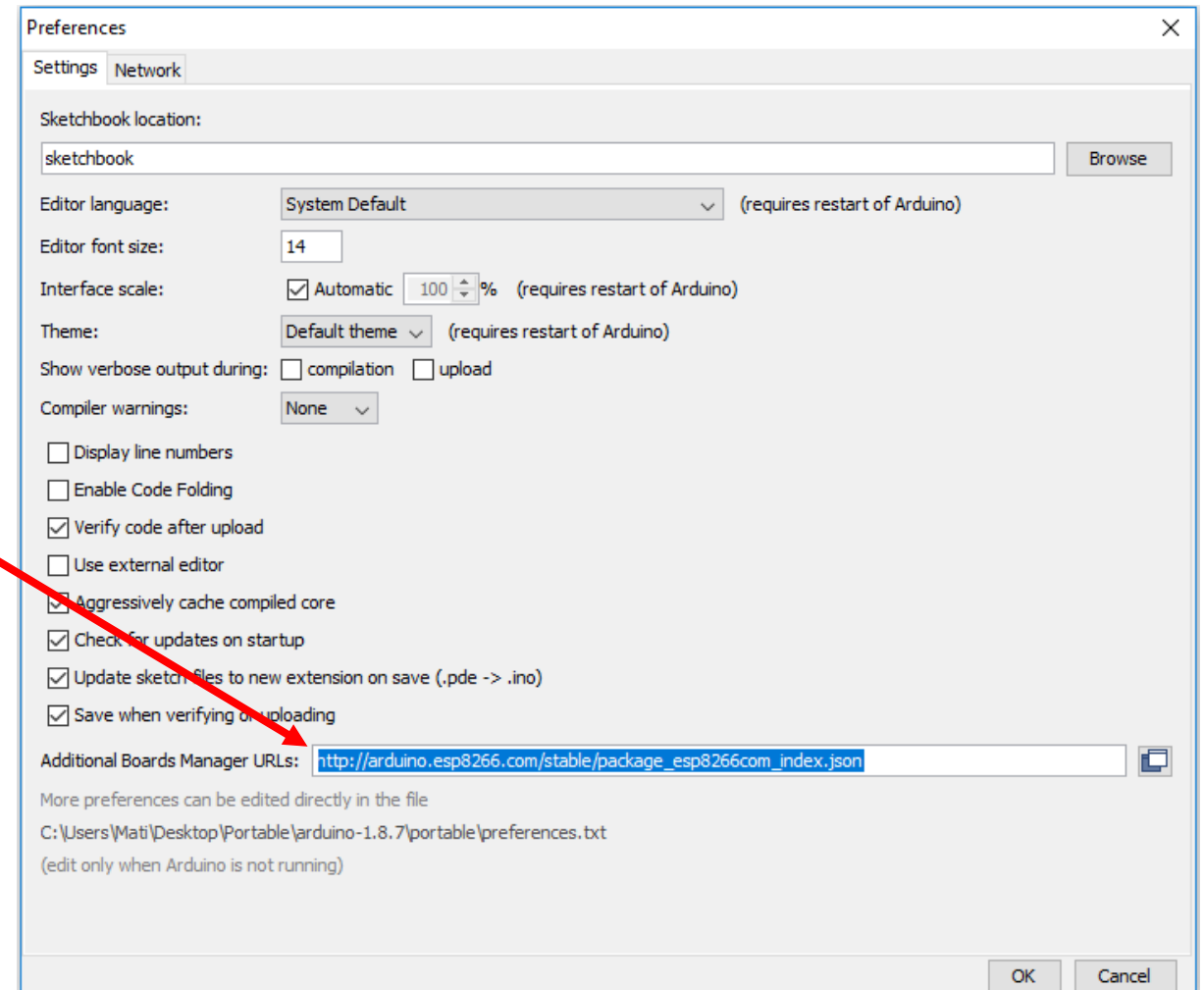
Puesta en Marcha

4 - Instalar el complemento para las placas con el módulo ESP8266.

4.A - Abrir el Arduino IDE, ir a Archivo -> Preferencias.

4.B - Copiar el siguiente URL en el cuadro de texto bajo el nombre "Additional Boards Manager URLs:".

http://arduino.esp8266.com/stable/package_esp8266com_index.json



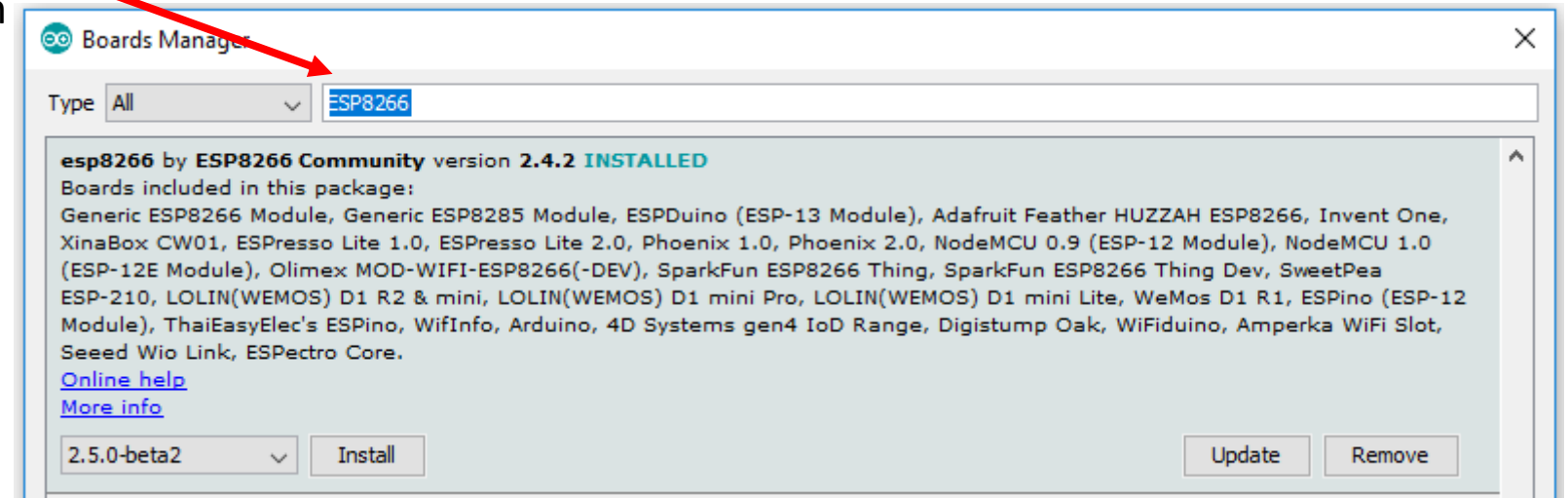
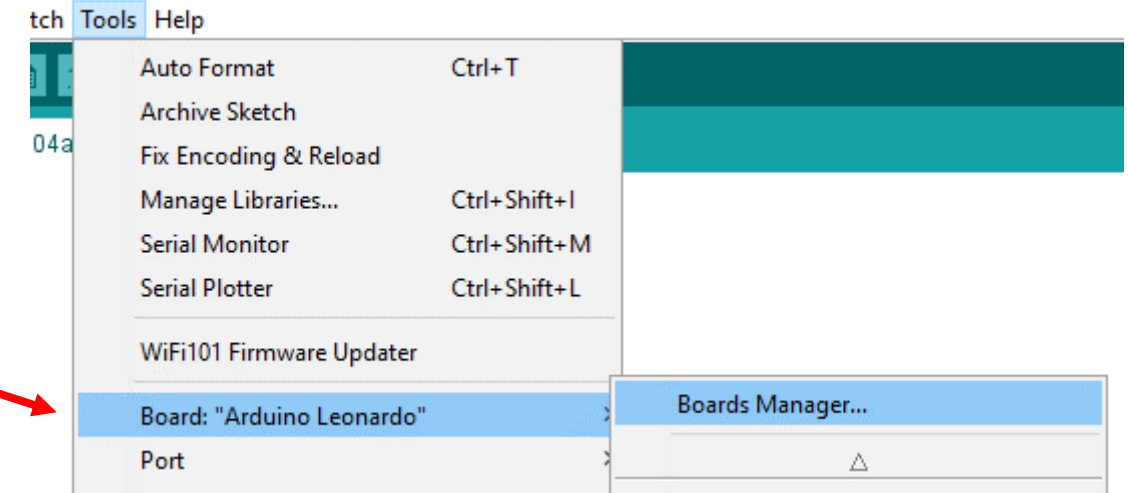
Puesta en Marcha

5 - Descargar las librerías para el módulo ESP8266.

5.A - Abrir el Arduino IDE, ir a Herramientas -> Placa -> Boards Manager...

5.B - Buscar “ESP8266” e instalar la librería con la última versión.

NOTA: La descarga y la Instalación podría tardar algunos minutos.



Puesta en Marcha

6 - Descargar e instalar el driver del USB.

NOTA: La mayoría de las placas usan el controlador CP2102 de SILABS, aunque hay placas que usan el controlador CH340 (de origen chino). En el caso de la placa NodeMCU – AMICA, usa el controlador de SILABS.



6.A – En el siguiente link podrá encontrar gran variedad de controladores de acuerdo al sistema operativo que esté utilizando:

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

Download for Windows 10 Universal (v10.1.4)

Platform	Software
 Windows 10 Universal	Download VCP (2.3 MB)

Download for Windows 7/8/8.1 (v6.7.6)

Platform	Software
 Windows 7/8/8.1	Download VCP (5.3 MB) (Default)
 Windows 7/8/8.1	Download VCP with Serial Enumeration (5.3 MB) Learn More »

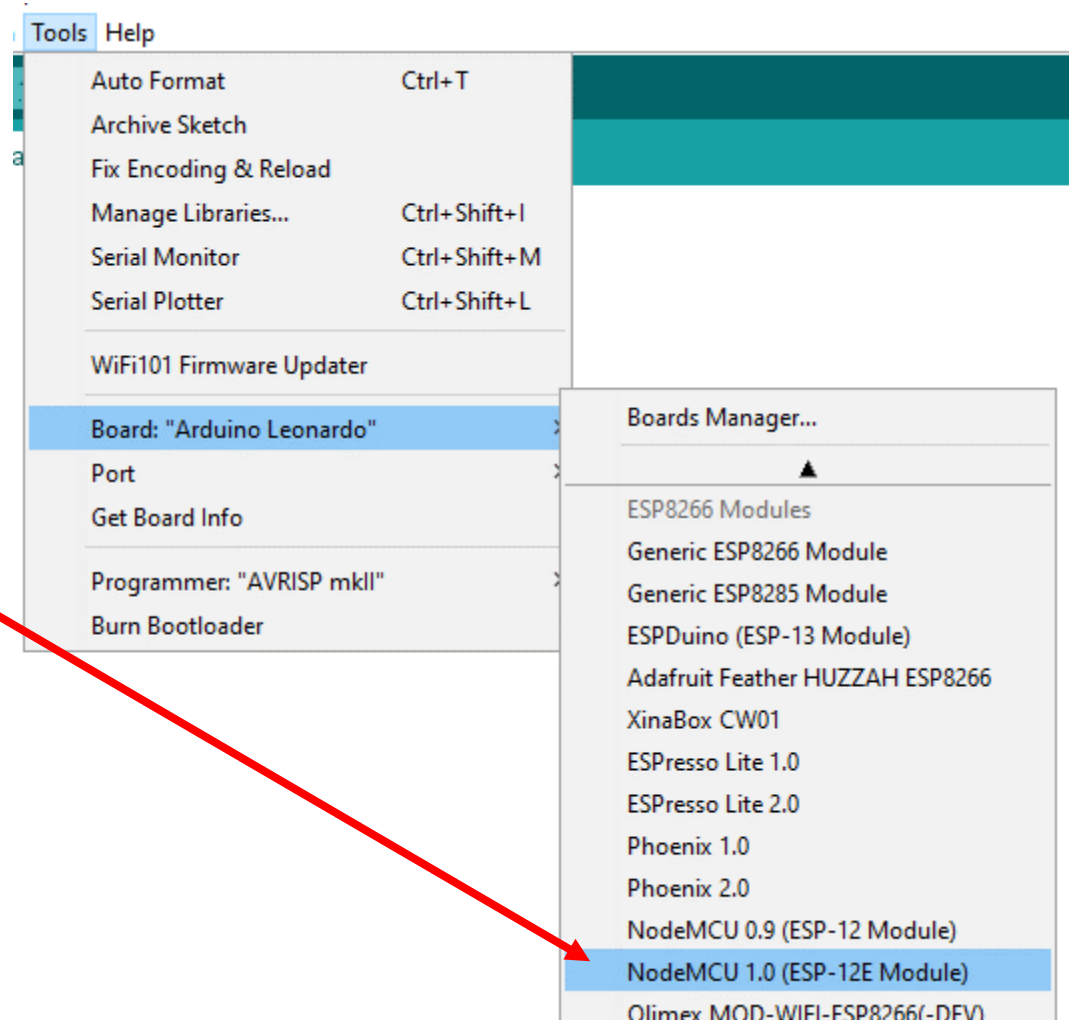
Puesta en Marcha

7 - Configurar la placa con el módulo ESP8266.

7.A – Una vez conectada la placa a la PC usando un cable USB. Abrir el Arduino IDE y seleccionar la placa que se este usando, para esto debemos ir a Herramientas -> Placa ->

NodeMCU 1.0 (ESP-12E Module)

NOTA: En nuestro caso estamos utilizando la placa NodeMCU, pero en el listado podrá encontrar una gran variedad de placas con el módulo ESP9266.

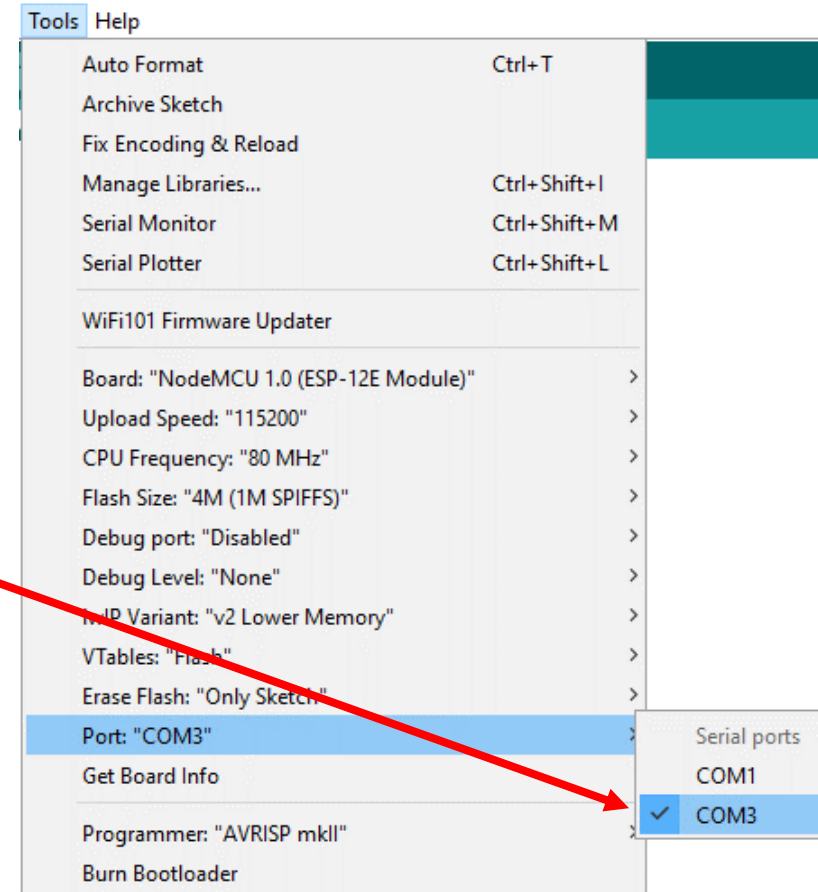


Puesta en Marcha

8 - Configurar el puerto USB.

8.A - En el Arduino IDE ir a Herramientas -> Puerto -> COMxx

NOTA: Si usa Windows, debería aparecer la palabra “COM” seguida de un número.

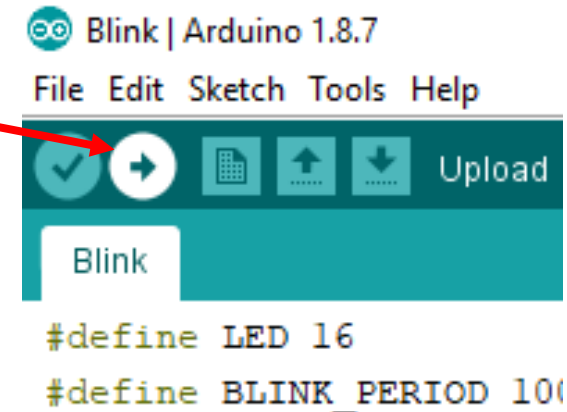
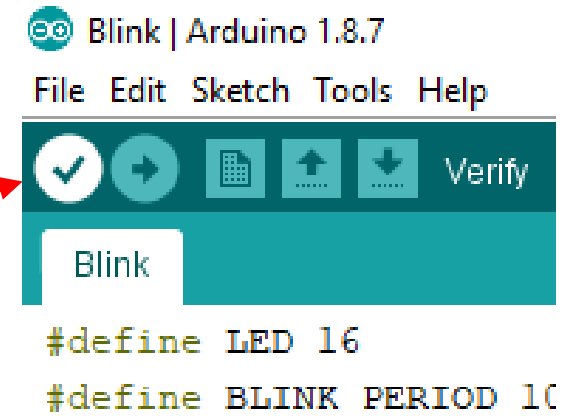


Puesta en Marcha

9 - Compilar y descargar un programa a la placa.

9.A - Para compilar el programa hay que oprimir el botón que tiene un tilde. Si no hay error en el programa, se puede descargar a la placa.

9.B - Para descargar el programa a la placa hay que oprimir el botón que tiene una flecha hacia la derecha.



Ejemplo 1

BLINK

```
#define LED D0  
#define BLINK_PERIOD 500
```

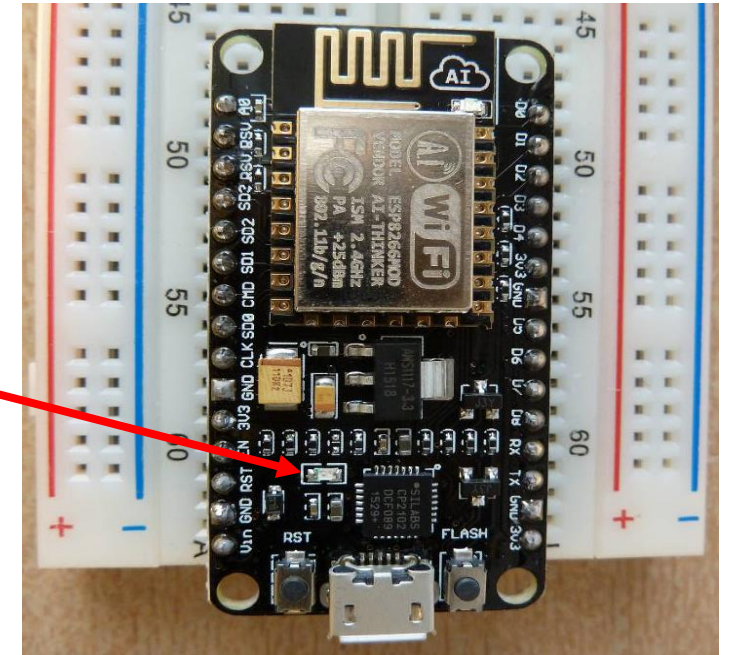
```
long lastMillis = 0;  
long currentMillis = 0;  
boolean ledState = false;
```

```
void setup() {  
  pinMode(LED, OUTPUT);  
}
```

```
void loop() {  
  currentMillis = millis();  
  if( (currentMillis - lastMillis) > BLINK_PERIOD) {  
    lastMillis = currentMillis;  
    ledState = !ledState;  
    digitalWrite(LED, ledState);  
  }  
}
```

NOTAS:

- El led de la placa está conectado al PIN D0.
- El led se prende escribiendo un LOW en el PIN D0 (Activo bajo).



Este condicional permite tener un programa no bloqueante. A diferencia de la función Delay, el programa no se queda “trabado.”

Ejemplo 2

DEBUG

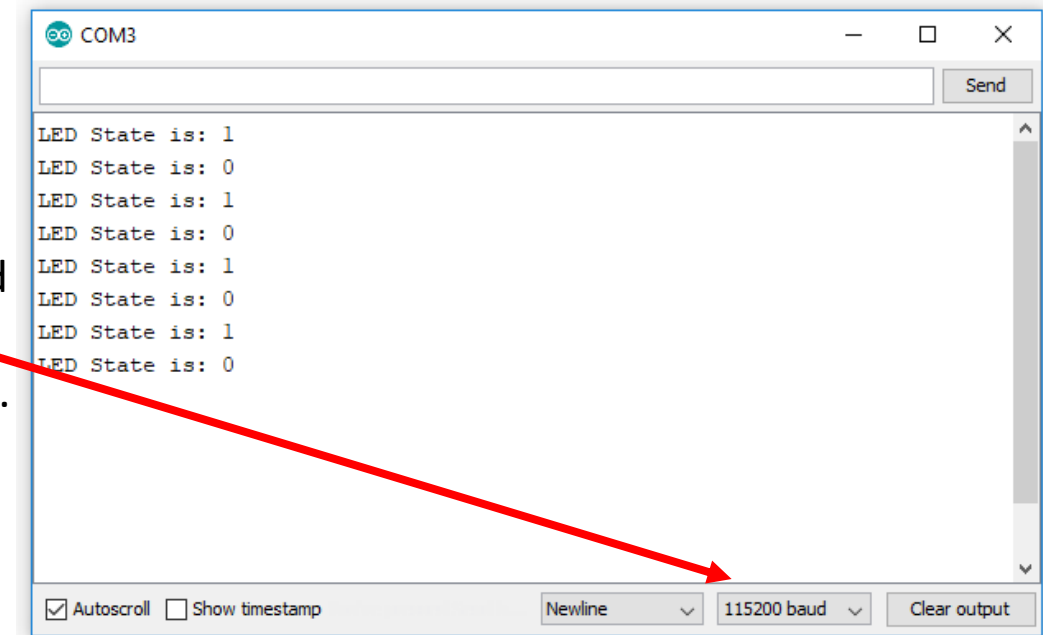
```
#define LED D0  
#define BLINK_PERIOD 1000
```

```
long lastMillis = 0;  
long currentMillis = 0;  
boolean ledState = true;
```

```
void setup() {  
  pinMode(LED, OUTPUT);  
  Serial.begin(115200);  
}
```

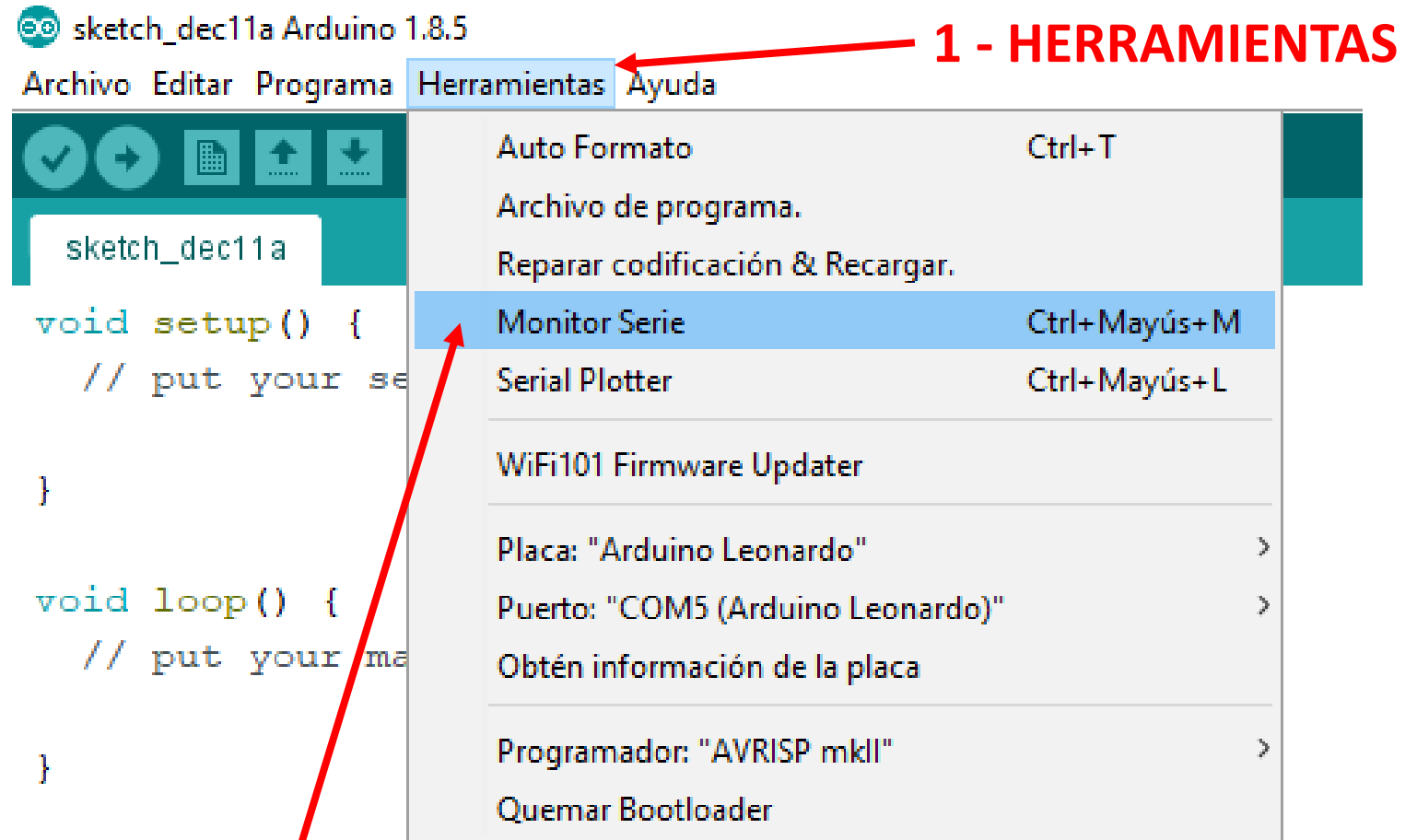
Es importante
configurar la velocidad
de transferencia de
datos a través del USB.

```
void loop() {  
  currentMillis = millis();  
  if( (currentMillis - lastMillis) > BLINK_PERIOD) {  
    lastMillis = currentMillis;  
    ledState = !ledState;  
    digitalWrite(LED, ledState);  
    Serial.print("LED State is: ");  
    Serial.println(ledState);  
  }  
}
```



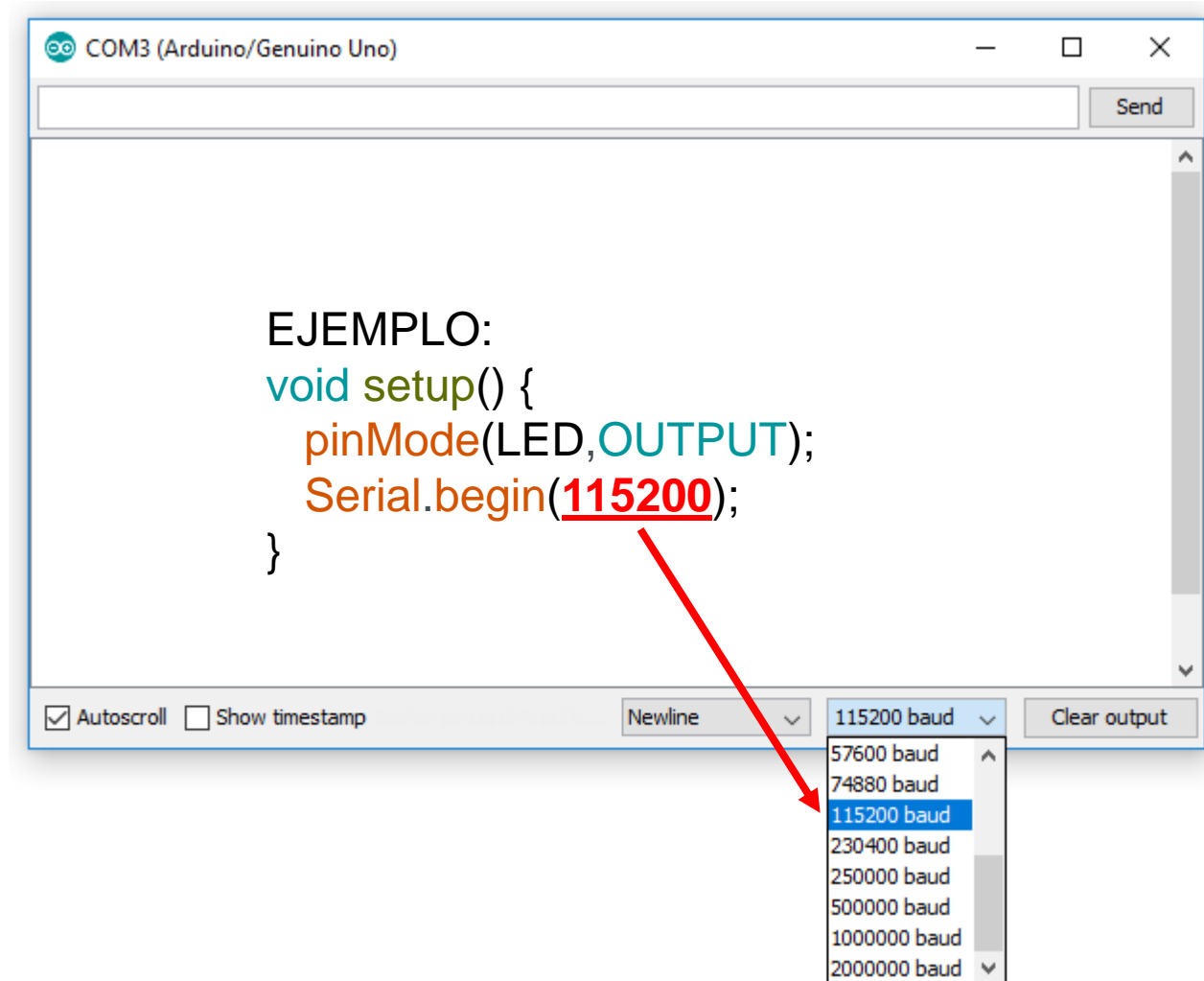
Para saber que está haciendo nuestro programa, podemos usar la función `Serial.print()`, que imprime una cadena de caracteres o valores de variables a través del puerto USB.

Monitor Serie

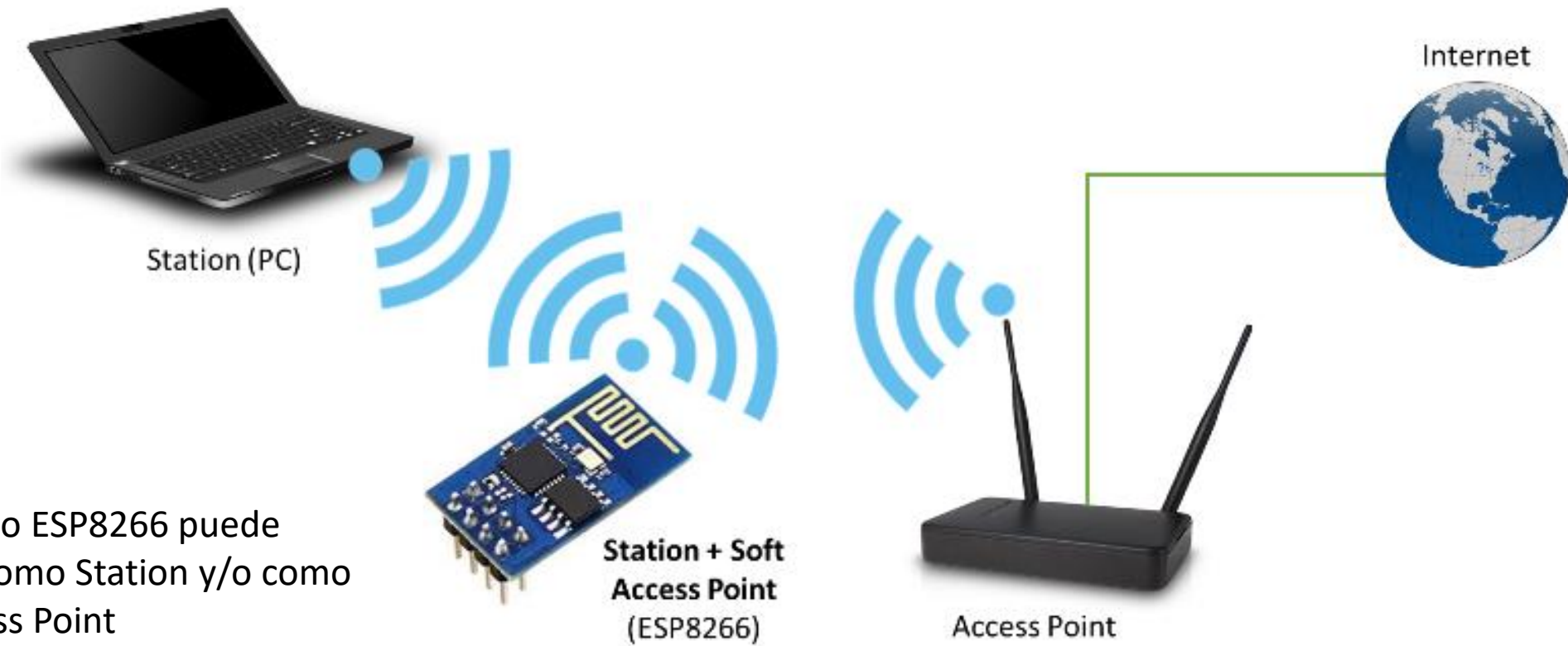


Monitor Serie

Verificar la
tasa de
transferencia
de datos



Wi-Fi



El módulo ESP8266 puede operar como Station y/o como Soft Access Point

Wi-Fi

ESP8266 operando en modo Station



El modo estación (STA) se utiliza cuando queremos que el módulo se conecte a una red Wi-Fi establecida por un Access Point.

Wi-Fi

ESP8266 operando en modo Soft AP



Un Access Point (AP) es un dispositivo que provee acceso a una red Wi-Fi a otros dispositivos (Stations), y los conecta a una red cableada. El ESP8266 puede proporcionar una funcionalidad similar, excepto que no tiene una interfaz con una red cableada. Este modo de operación se lo conoce como Soft Acces Point (soft-AP). Y el número máximo de dispositivos conectados (Stations) al soft-AP es de cinco.

Ejemplo 3

SCAN

```
#include "ESP8266WiFi.h"
```

```
#define LED D0  
#define SCAN_PERIOD 5000
```

```
long lastMillis = 0;  
long currentMillis = 0;  
int redes = 0;
```

```
void setup() {  
  pinMode(LED, OUTPUT);  
  Serial.begin(115200);  
  WiFi.mode(WIFI_STA);  
  WiFi.disconnect();  
  delay(100);  
}
```

Si se configura como true, el escaneo de redes se hace de fondo y la función retorna sin esperar a un resultado.

Modos:

- WIFI_AP
- WIFI_STA
- WIFI_AP_STA
- WIFI_OFF

Retorna el número de redes encontradas.

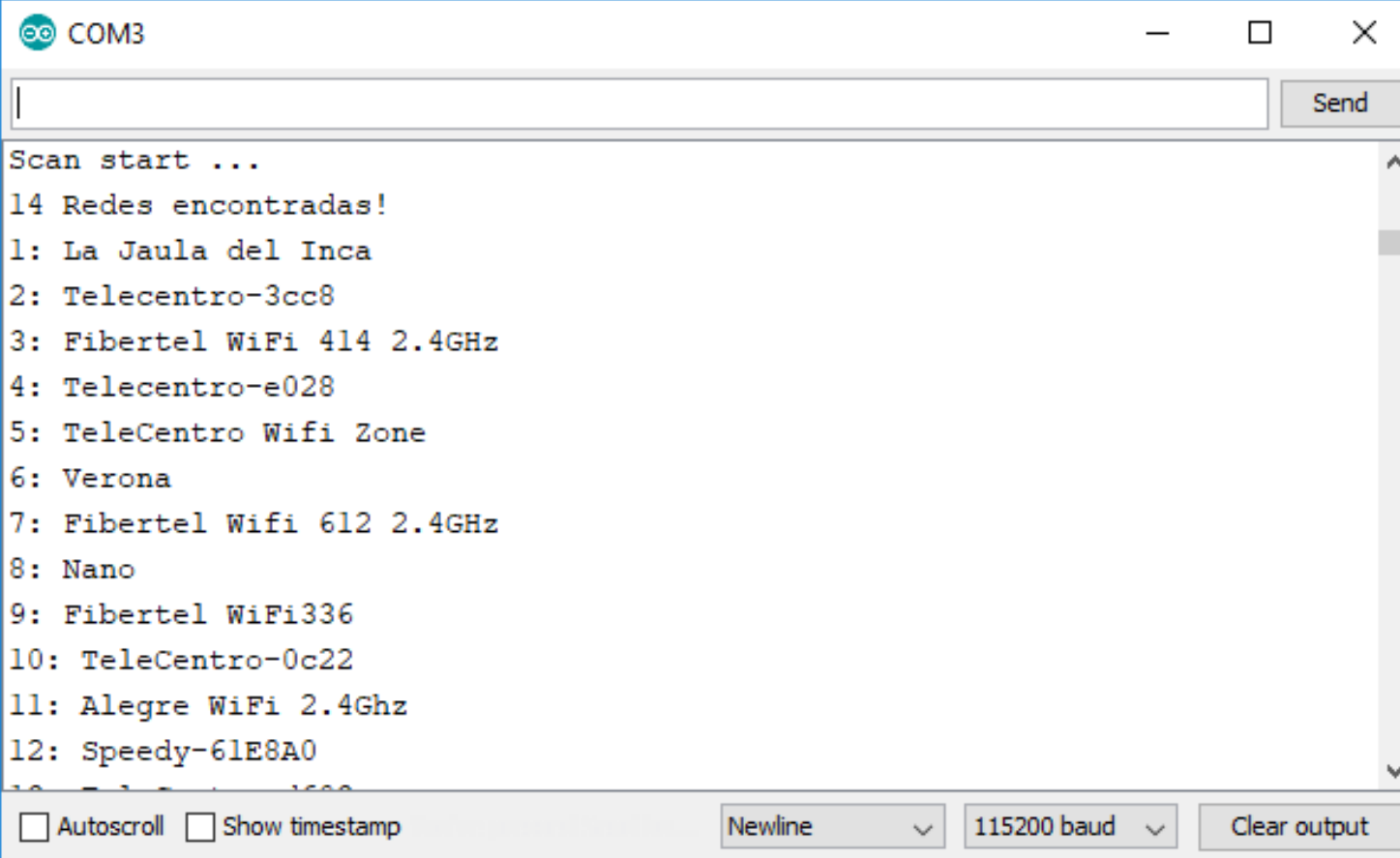
Cierra la conexión a un Access Point que el módulo pudo haber hecho automáticamente usando las credenciales guardadas previamente.

Se borra el resultado de la búsqueda.

```
void loop() {  
  currentMillis = millis();  
  if (currentMillis - lastMillis > SCAN_PERIOD) {  
    lastMillis = currentMillis;  
    digitalWrite(LED, LOW);  
    WiFi.scanNetworks(true);  
    Serial.println();  
    Serial.println("Scan start ... ");  
  }  
  
  redes = WiFi.scanComplete();  
  if (redes >= 0) {  
    digitalWrite(LED, HIGH);  
    Serial.print(redes);  
    Serial.println(" Redes encontradas!");  
    for (int i = 0; i < redes; i++) {  
      Serial.print(i+1);  
      Serial.print(": ");  
      Serial.println(WiFi.SSID(i));  
    }  
    WiFi.scanDelete();  
  }  
}
```


Ejemplo 3

SCAN - Resultado



```
COM3
Scan start ...
14 Redes encontradas!
1: La Jaula del Inca
2: Telecentro-3cc8
3: Fibertel WiFi 414 2.4GHz
4: Telecentro-e028
5: TeleCentro Wifi Zone
6: Verona
7: Fibertel Wifi 612 2.4GHz
8: Nano
9: Fibertel WiFi336
10: TeleCentro-0c22
11: Alegre WiFi 2.4Ghz
12: Speedy-61E8A0
13: ...
14: ...
```

Wi-Fi

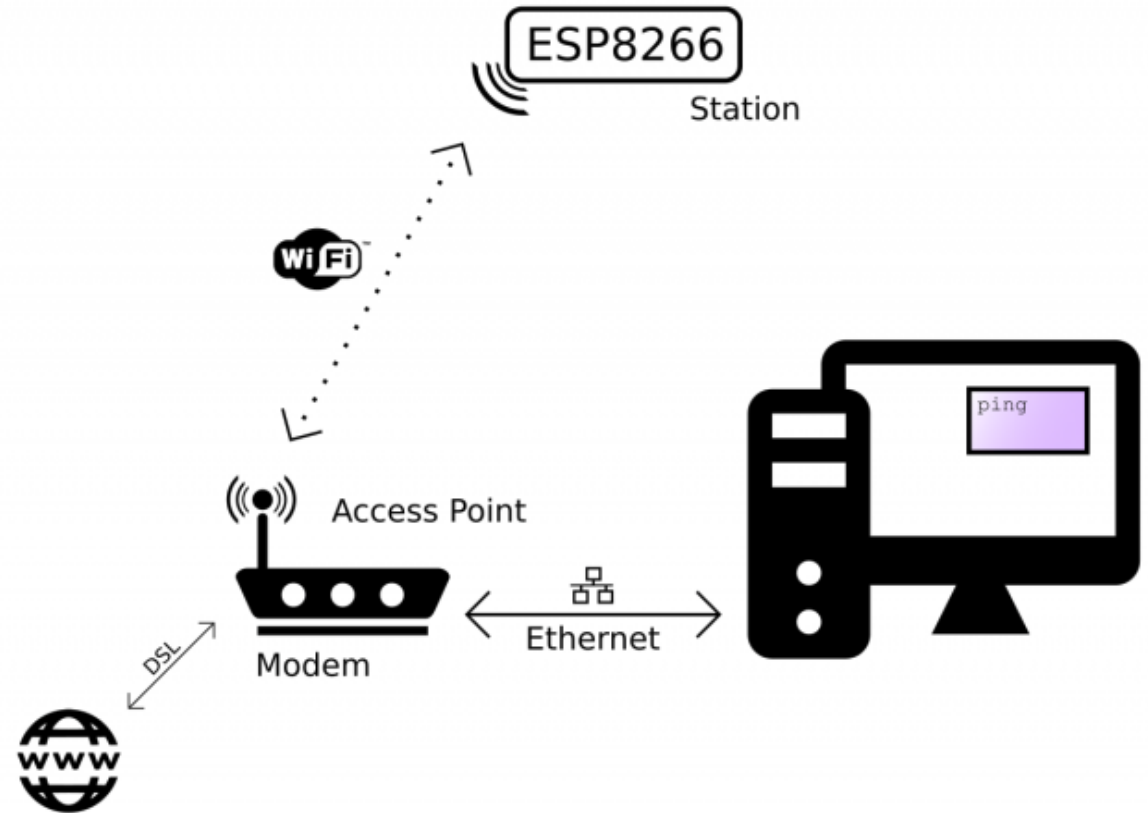
ESP8266 operando en modo Soft AP

Para acceder a una red Wi-Fi es necesario conocer el SSID (nombre de la red) y la contraseña. Para realizar la conexión se debe usar la función:

WiFi.begin(ssid,password);

Una vez llamada a esta función, el módulo ESP8266 intentará reconectarse automáticamente cuando se genere una desconexión.

Llamando a **WiFi.begin()**; sin parámetros, el modulo intentará conectarse a la última red que se ha conectado.



Wi-Fi

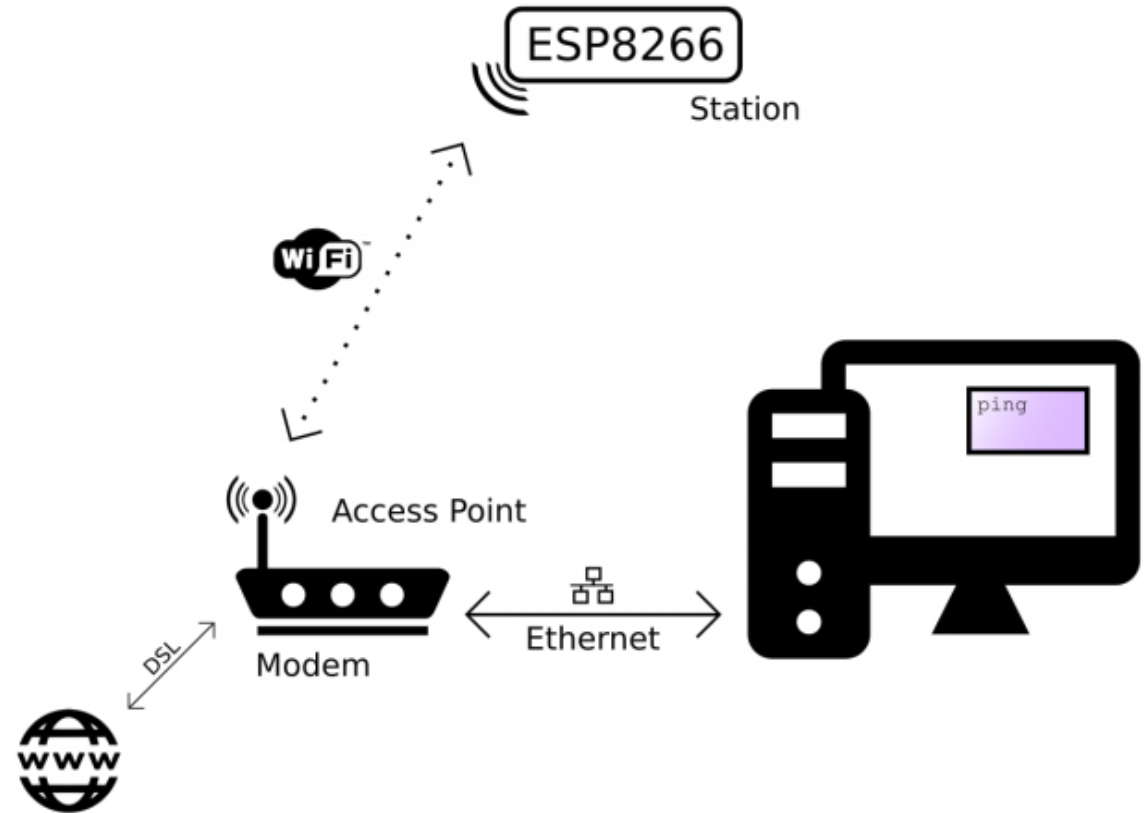
ESP8266 operando en modo Soft AP

La flexibilidad del ESP8266 nos permite configurar distintos parámetros:

`WiFi.begin(ssid, password, channel, bssid, connect);`

Donde:

- **Ssid:** Nombre de la red
- **Password:** Contraseña de la red
- **Channel:** Canal de la red
- **Bssid:** MAC address del Access point
- **Connect:** Es un parámetro booleano que si se establece en 'false', el módulo que solo guardará los otros parámetros sin establecer la conexión con el Access Point.



Wi-Fi

ESP8266 operando en modo Soft AP

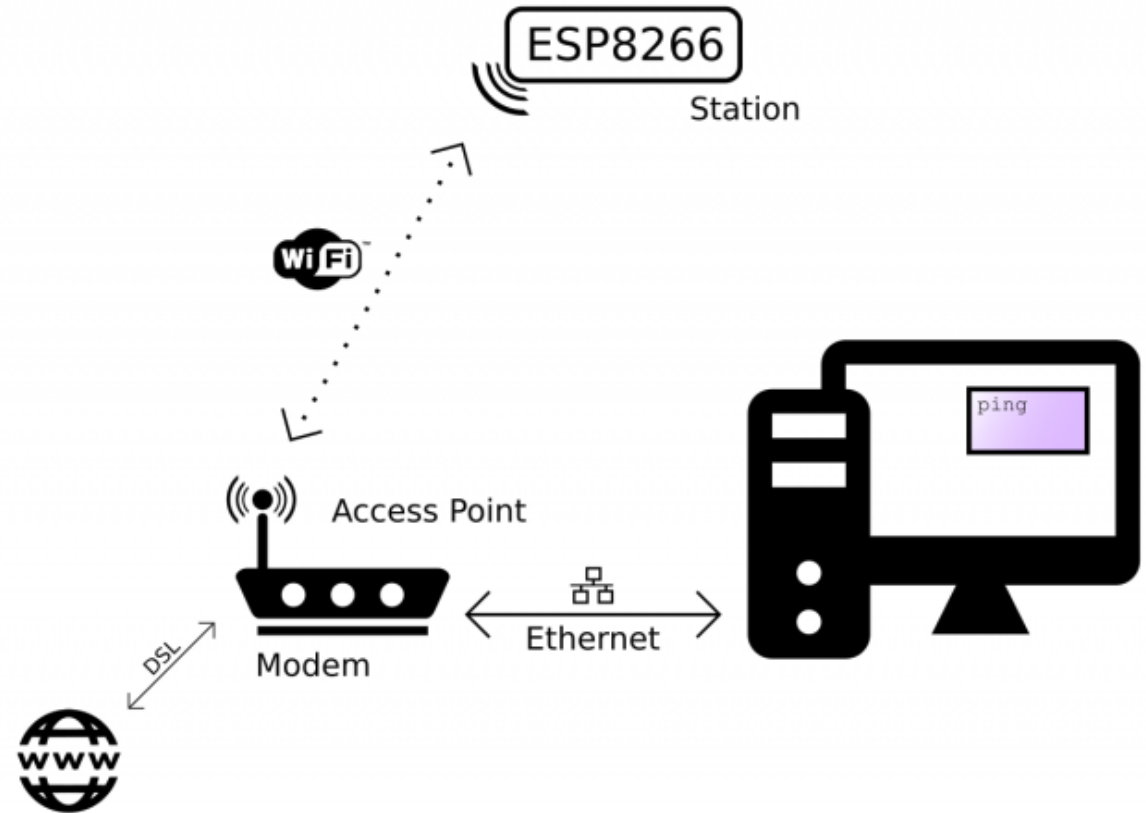
También podemos deshabilitar el DHCP y configurar la dirección IP del dispositivo manualmente llamando a la función:

WiFi.config(local_ip, gateway, subnet, dns1, dns2);

Donde:

- **Local_ip**: Dirección IP del dispositivo
- **Gateway**: Dirección IP del Gateway (Router) para acceder a una red externa
- **Subnet**: Mascara que define el rango de las direcciones IP dentro de la red
- **Dns1, dns2**: Direcciones IP de los Domain Name Servers (DNS). El DNS contiene el directorio de nombres de dominio.

EJ: itba.edu.ar (190.104.250.104)



Ejemplo 4

HOOK

```
#include "ESP8266WiFi.h"

#define LED D0
#define SCAN_PERIOD 5000

long lastMillis = 0;
long currentMillis = 0;
int redes = 0;
boolean ledState = false;

void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(100);
  WiFi.begin("NODE_RED", "GEDA2016");
  Serial.println("Connecting...");
```

```
while(WiFi.status() != WL_CONNECTED) {
  ledState = !ledState;
  digitalWrite(LED, ledState);
  delay(100);
}
Serial.print("Connected! IP address: ");
Serial.println(WiFi.localIP());
digitalWrite(LED, LOW);
} FIN setup()

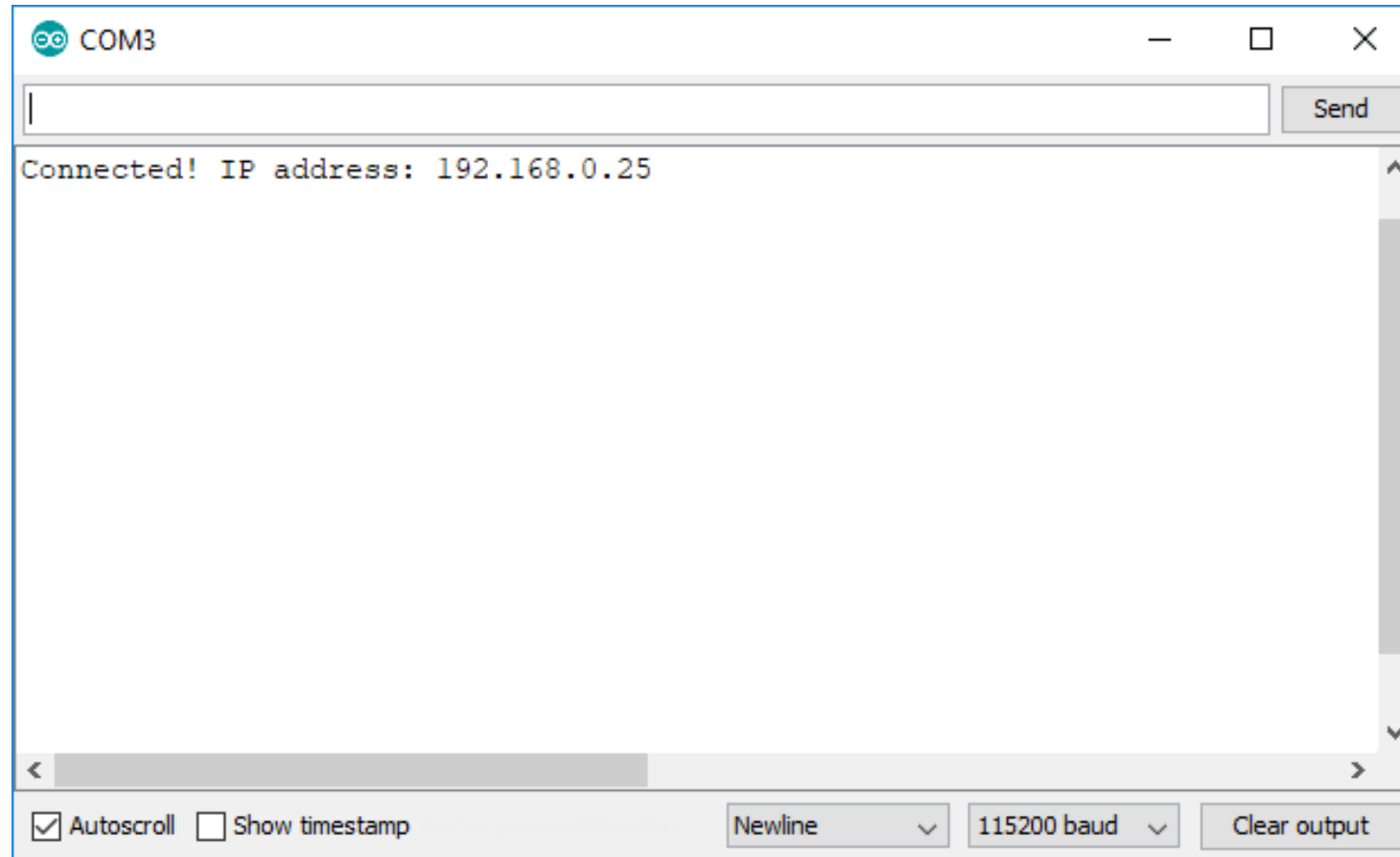
void loop() {
}
```

Mientras esté
desconectado

SSID y PASSWORD de la red

Ejemplo 4

HOOK - Resultado



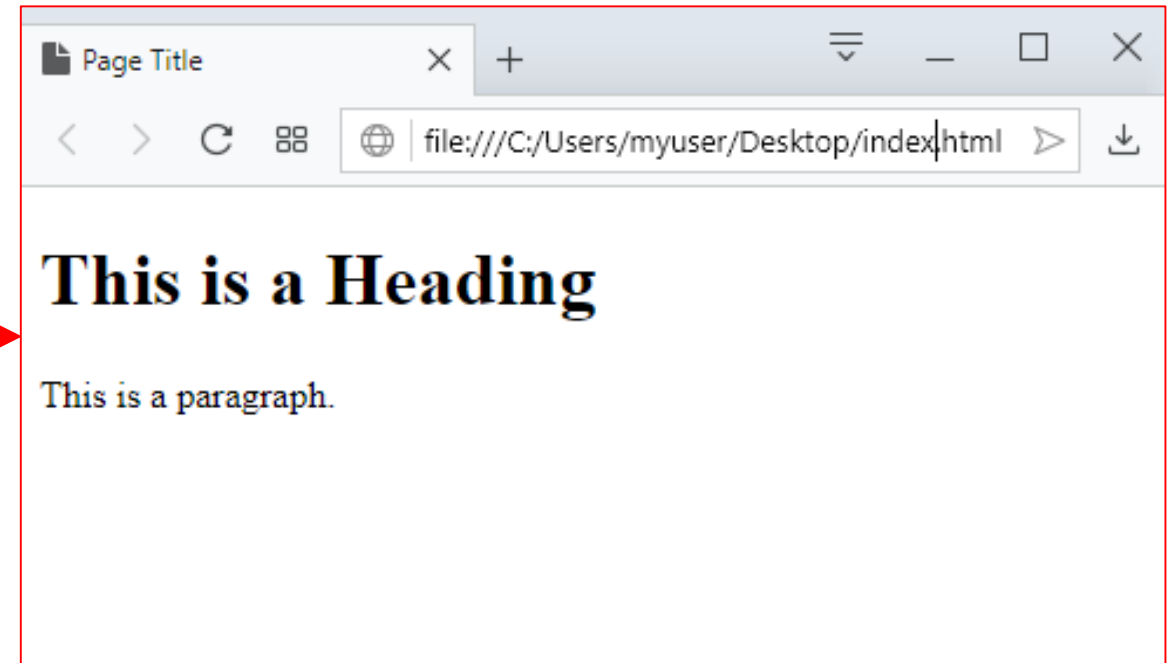
Web Server

HTML (Hyper Text Markup Language)

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

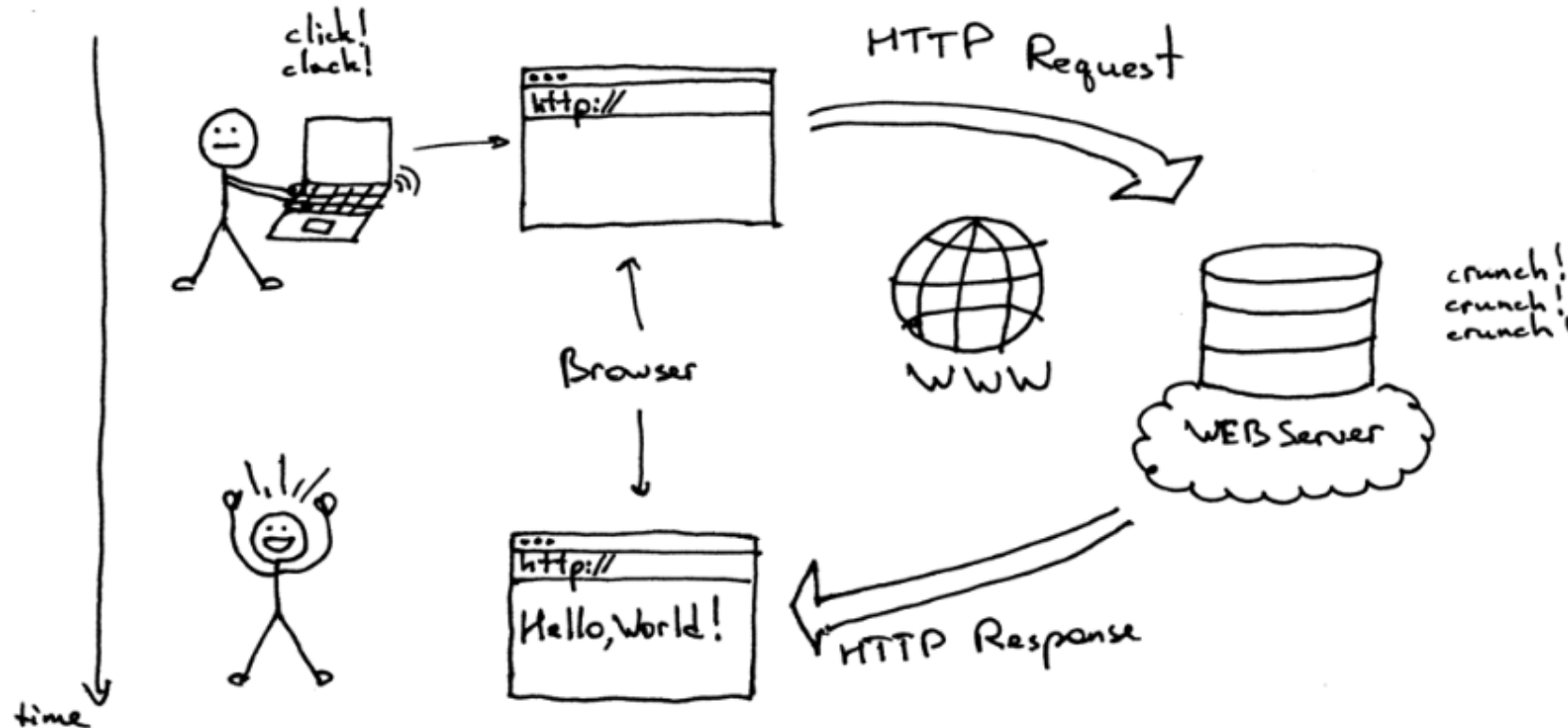
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```



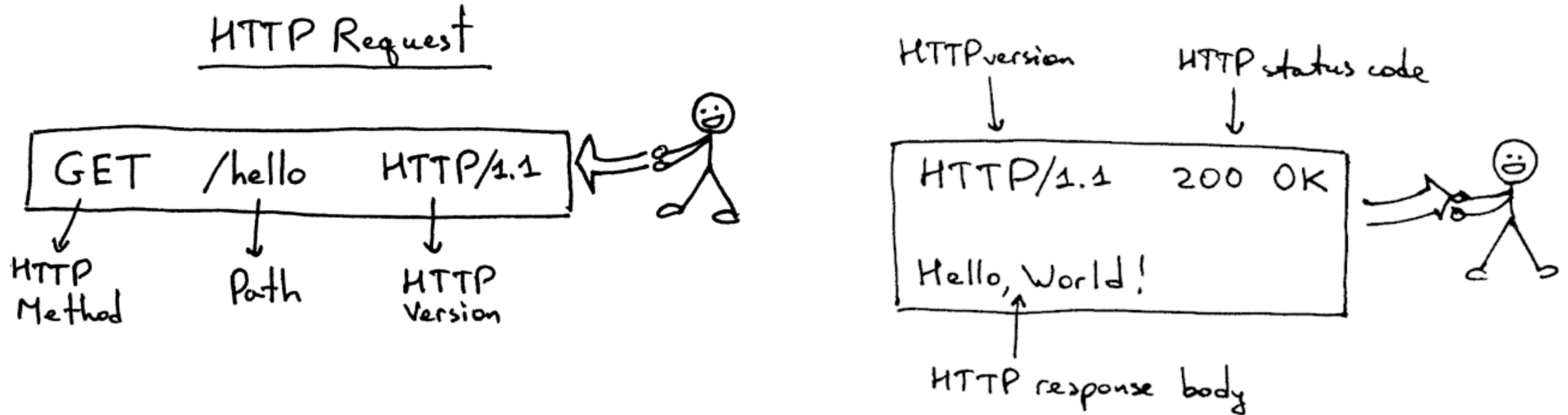
Web Server

HTTP Request / Response



Web Server

HTTP Request / Response



Web Server

HTTP Status Codes

Code	Description	Code	Description
200	OK	400	Bad Request
201	Created	401	Unauthorized
202	Accepted	403	Forbidden
301	Moved Permanently	404	Not Found
303	See Other	410	Gone
304	Not Modified	500	Internal Server Error
307	Temporary Redirect	503	Service Unavailable

Ejemplo 5

SERVER 1 – Página Simple

```
#include "ESP8266WiFi.h"
#include <ESP8266WebServer.h>
#define LED D0
#define LED_OFF HIGH
#define LED_ON LOW
```

```
boolean ledState=false;
```

```
ESP8266WebServer server(80);
```

Configuración
del puerto.

```
void setup() {
  pinMode(LED, OUTPUT);
  digitalWrite(LED, LED_OFF);
  Serial.begin(115200);
  Serial.println();
  Serial.println("Connecting to Access Point ...");
  WiFi.disconnect();
  WiFi.begin("NODE_RED", "GEDA2016");
```

```
while (WiFi.status() != WL_CONNECTED) {
  ledState = !ledState;
  digitalWrite(LED, ledState);
  delay(100);
}
digitalWrite(LED, LED_ON);
```

```
Serial.print("Conectado a NODE_RED! IP address: ");
Serial.println(WiFi.localIP());
```

```
server.on("/", root);
server.begin();
Serial.println("WEB SERVER INICIADO");
}
```

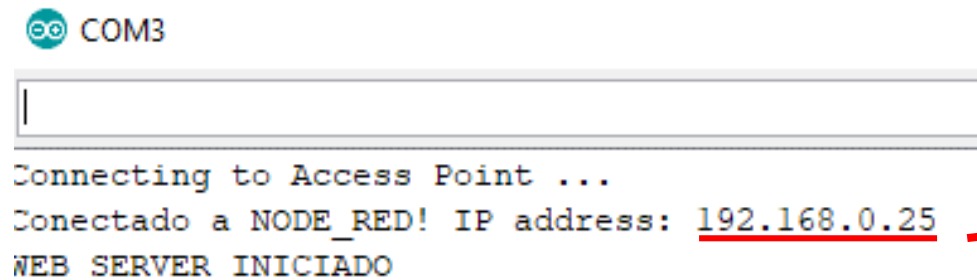
Configuración del
directorio raíz
(página principal)

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

```
void root() {
  server.send(200, "text/html", "<h1>Conectado al servidor web!</h1>");
}
```

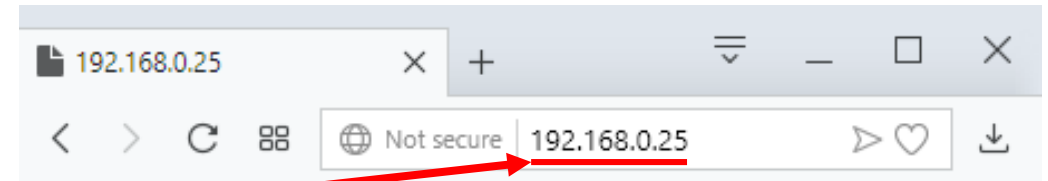
Ejemplo 5

SERVER 1 - Resultado



COM3

Connecting to Access Point ...
Conectado a NODE_RED! IP address: 192.168.0.25
WEB SERVER INICIADO



Conectado al servidor web!

Nota: Si no aparece el mensaje en el monitor serie es porque se envía durante la ejecución del `setup()` (una sola vez). Para poder ver el mensaje deberíamos tener el monitor serie abierto y pulsar el botón de *Reset* que contiene la placa.

Ejemplo 6

SERVER 2 – Múltiples Páginas

```
#include "ESP8266WiFi.h"
#include <ESP8266WebServer.h>
#define LED D0
#define LED_OFF HIGH
#define LED_ON LOW

boolean ledState=false;
ESP8266WebServer server(80);

void setup() {
  pinMode(LED, OUTPUT);
  digitalWrite(LED,LED_OFF);
  Serial.begin(115200);
  Serial.println();
  Serial.println("Connecting to Access Point ...");
  WiFi.disconnect();
  WiFi.begin("NODE_RED", "GEDA2016");
```

```
  while (WiFi.status() != WL_CONNECTED) {
    ledState = !ledState;
    digitalWrite(LED, ledState);
    delay(100);
  }
  digitalWrite(LED,LED_ON);

  Serial.print("Conectado a NODE_RED! IP address: ");
  Serial.println(WiFi.localIP());
  server.on("/", root);
  server.on("/plain", plain);
  server.on("/red", red);
  server.on("/blue", blue);
  server.on("/image", image);
  server.begin();
  Serial.println("WEB SERVER INICIADO");
}

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

Configuración de
las sub-páginas

Ejemplo 6

SERVER 2

```
void root() {  
    String webPage;  
    webPage += "<h1>Conectado al servidor  
web!</h1>";  
    webPage += "<p>Su direccion IP es: </p>";  
    webPage += WiFi.localIP().toString();  
    server.send(200, "text/html", webPage);  
}
```

```
void plain() {  
    server.send(200, "text/plain", "Texto plano (sin  
formato)");  
}  
  
void red() {  
    server.send(200, "text/html", " <!DOCTYPE html>  
<html> <body> <h2 style=\"background-color:red\">  
Podemos escribir con fondo rojo! </h2> </body>  
</html>");  
}
```

Ejemplo 6

SERVER 2

```
void blue() {  
  String webPage;  
  webPage += "<!DOCTYPE html>";  
  webPage += "<html>";  
  webPage += "<body>";  
  webPage += "<h2 style=\"color:#0000FF;\">  
Caracteres azules! </h2>";  
  webPage += "</body>";  
  webPage += "</html>";  
  server.send(200, "text/html", webPage);  
}
```

```
void image() {  
  String webPage;  
  String image = "<img src=\"https://www.itba.edu.ar/wp-  
content/uploads/2016/12/Gal-Elec-6.jpg\">";  
  webPage += "<!DOCTYPE html>";  
  webPage += "<html>";  
  webPage += "<body>";  
  webPage += image;  
  webPage += "</body>";  
  webPage += "</html>";  
  server.send(200, "text/html", webPage);  
}
```

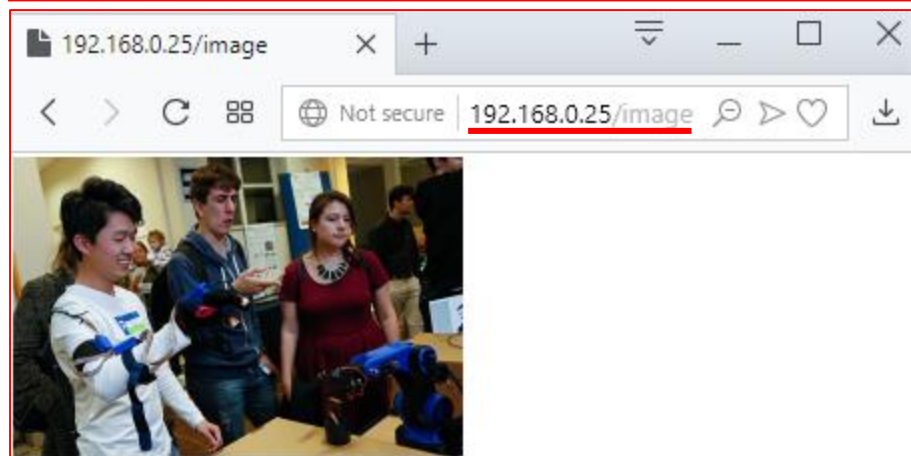
Ejemplo 6

SERVER 2 - Resultado

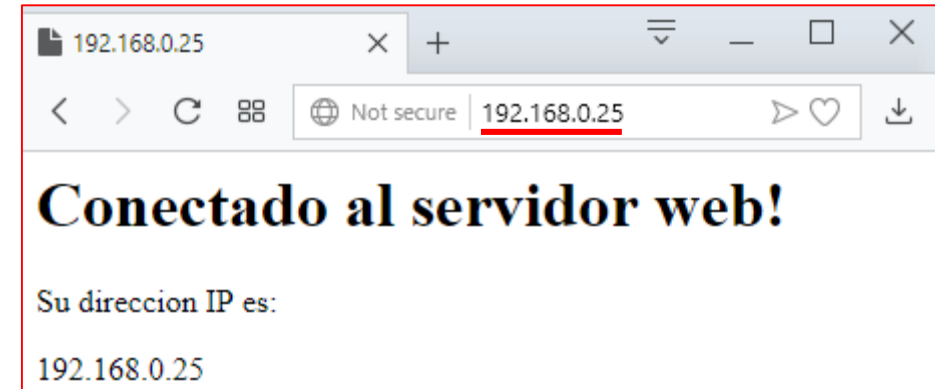
```
COM3
Conectado a NODE_RED! IP address: 192.168.0.25
WEB SERVER INICIADO
```



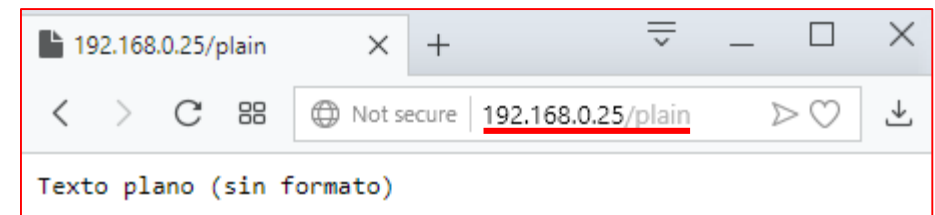
/red



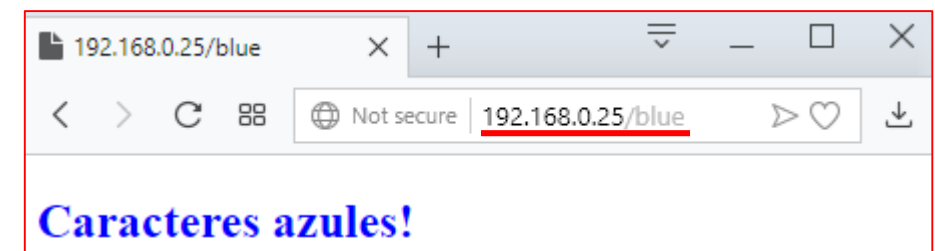
/image



/



/plain



/blue

Ejemplo 7

SERVER 3 – Botones

```
#include "ESP8266WiFi.h"
#include <ESP8266WebServer.h>
#define LED D0
#define LED_OFF HIGH
#define LED_ON LOW
```

```
boolean ledState=false;
String webPage = "";
ESP8266WebServer server(80);
```

Creamos una
única página web

```
void setup() {
  pinMode(LED, OUTPUT);
  digitalWrite(LED, LED_OFF);
  Serial.begin(115200);
  Serial.println();
  Serial.println("Connecting to Access Point ...");
  WiFi.disconnect();
  WiFi.begin("NODE_RED", "GEDA2016");
  while (WiFi.status() != WL_CONNECTED) {
    ledState = !ledState;
    digitalWrite(LED, ledState);
    delay(100);
  }
```

```
    digitalWrite(LED, LED_ON);
    Serial.print("Conectado a NODE_RED! IP address: ");
    Serial.println(WiFi.localIP());
    initWebPage();
    server.on("/", root);
    server.on("/ledON", ledON);
    server.on("/ledOFF", ledOFF);
    server.begin();
    Serial.println("WEB SERVER INICIADO");
  }
```

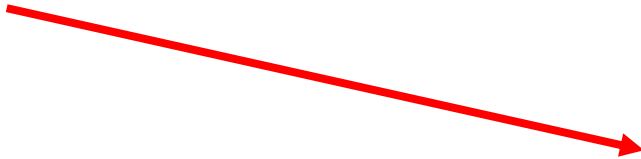
Función que crea la
página principal

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

Ejemplo 7

SERVER 3 – Botones

```
void initWebPage() {  
  webPage += "<!DOCTYPE html>";  
  webPage += "<html>";  
  webPage += "<body>";  
  webPage += "<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\"></style>";  
  webPage += "<h1>CONTROL DEL LED</h1>";  
  webPage += "<p>Utilice los botones para apagar y prender el led.</p>";  
  webPage += "<p><a href=\"ledOFF\"><button>OFF</button></a>&nbsp;<a  
href=\"ledON\"><button>ON</button></a></p>";  
  webPage += "</body>";  
  webPage += "</html>";  
}
```

A red arrow originates from the HTML code line `<button>ON</button>` and points towards a red-bordered box on the right side of the slide.

Estructura para
insertar botones en
la página web

NOTA: Observar que el botón OFF llama a la página “ledOFF” y que el botón ON llama a la página “ledON”

Ejemplo 7

SERVER 3 – Botones

```
void root() {  
  server.send(200, "text/html", webPage);  
}
```

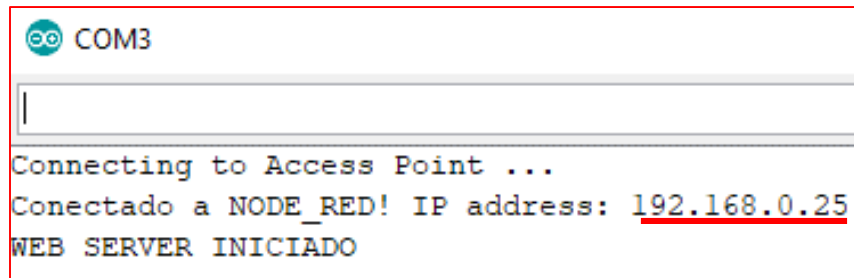
```
void ledON() {  
  digitalWrite(LED, LED_ON);  
  server.send(200, "text/html", webPage);  
}
```

```
void ledOFF() {  
  digitalWrite(LED, LED_OFF);  
  server.send(200, "text/html", webPage);  
}
```

Antes de enviar la página web podemos hacer otras cosas, como prender y apagar un led.

Ejemplo 7

SERVER 3 - Resultado





Buena Suerte!!!

Arduino