IRAN Internet Of Things

نحوه دسترسی به WiFiبا esp8266

استاد صالحی

۱۰ سال تدریس در دانشگاههای دولتی ، آزاد ، غیر انتفاعی ، فنی حرفه ای ، الکترونیکی و علمی کاربردی در استان تهران

@alisalehi556

ESP-8266 #include <ESP8266WiFi.h>

کلاسهای موجود در کتابخانه Esp8266wifi.h

Classes — مود عملیاتی را تعیین می‌کند که آیا در مود client ، station یا AP کار میکند

WiFi Class — Esp8266 را به یک کلاینت برای اتصال به یک سرور تبدیل می‌کند.

Client Class

Server Class — Esp8266 را به یک سرور برای اتصال به یک کلاینت تبدیل می‌کند.

IP Address Class — تنظیمات IP Address ها

Scan Class — نمایش لیست AP های اطراف ماژول

UDP Class — برقراری ارتباطات UDP و ارسال و دریافت دیتا

استاد : صالحی مدرس دانشگاههای برتر استان تهران

WiFi Class

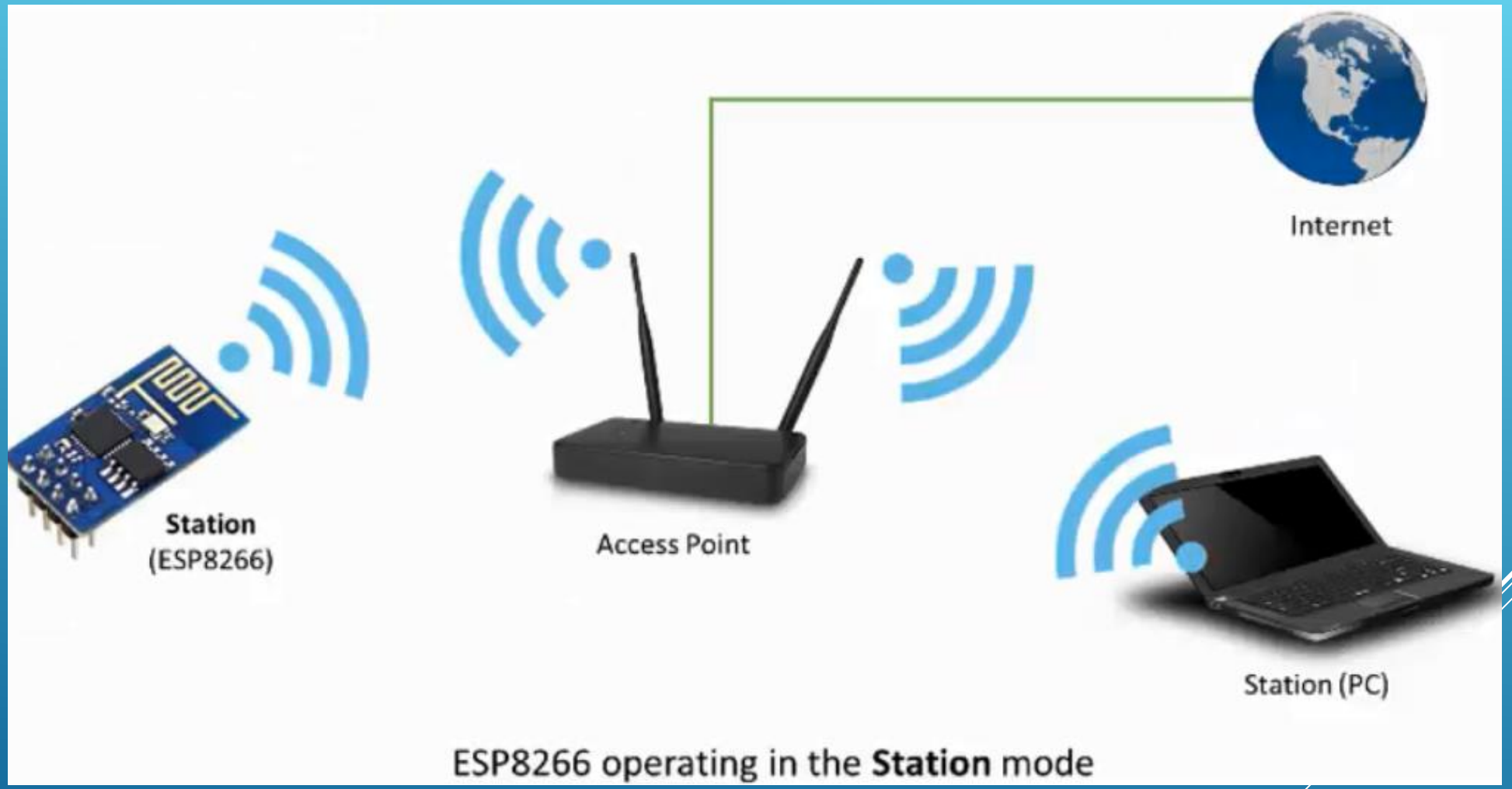
استاد : صالحی مدرس دانشگاههای برتر استان تهران

مودهای موجود در کلاس WiFi

۱– Station (پیش فرض)
۲– SoftAP
۳– Station + SoftAP

استاد : صالحی مدرس دانشگاههای برتر استان تهران

WiFi.mode(WIFI_STA) ⟵ جهت ورود به مود station



درمود station ، ماژول esp8266 می‌توانید به AP وصل شوید و یک ارتباط دو طرفه برقرار کنید
و از طریق AP به دنیای اینترنت وصل می‌شویم.

توابع موجود در مود station

WiFi.begin(SSID,password) →

هم SSID (نامAP) و هم password داخل دابل کوتیشن قرار می‌گیرد
در حقیقت شروع اتصال به اکسس پوینت توسط begin انجام می‌شود
در صورتیکه در DHCP داشتیم ،از AP یک IP می‌گیرد و در غیر اینصورت
یک IP ثابت برای آن تنظیم می کنیم.

begin | config | reconnect | disconnect | isConnected | setAutoConnect | getAutoConnect | setAutoReconnect | waitForConnectResult | macAddress | localIP | subnetMask | gatewayIP | dnsIP | hostname | status | SSID | psk | BSSID | RSSI | WPS | Smart Config

میزان قدرت سیگنال که از AP دریافت میکنیم.
RSSI: Received Signal Strength indicator
BSSID: Basis Service Set Identifier
ESSID: Extended Service Set Identifier

استاد : صالحی مدرس دانشگاههای برتر استان تهران

WiFi.mode(WIFI_AP) → جهت ورود به مود Software Access Point

درمود Soft AP ، ماژول esp8266 می‌تواند خودش به عنوان AP عمل کند و تمام سیستمهای دیگر می توانند به آن وصل شوند.
که میتوان برای آن SSID تنظیم کرد یا اینکه آنرا بعنوان DHCP تنظیم کرد که به سایر دستگاهها IP بدهد.

استاد : صالحی مدرس دانشگاههای برتر استان تهران

توابع موجود در مود Soft Access Point

softAP | softAPConfig | softAPdisconnect | softAPgetStationNum | softAPIP | softAPmacAddress

تمام توابع با SoftAP شروع شده اند

استاد : صالحی مدرس دانشگاههای برتر استان تهران

اگر بخواهیم فقط از میکروکنترلر استفاده کنیم و WiFi را قطع کنیم ، مثلا زمانیکه به یک ماژول GPRS یا GPS یا GSM متصل کنیم، دستور WIFI_OFF را داخل پرانتز بنویسیم.

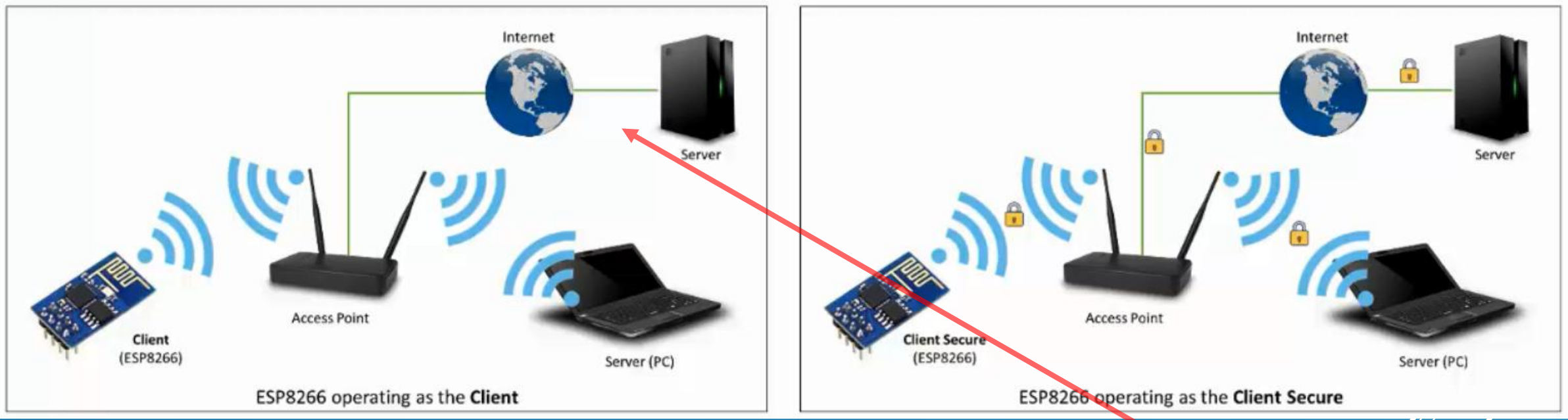
WiFi.mode(WIFI_AP_STA) or WIFI_OFF → جهت ورود به مود station + SoftAP

استاد : صالحی مدرس دانشگاههای برتر استان تهران

درمود Station + Soft AP ، ماژول esp8266 می‌تواند هم بعنوان AP برای رایانه ها و هم بعنوان Station برای AP دیگر باشد.

Client Class

انواع مود در Client Class که دو مود Client و Client Secure دارد

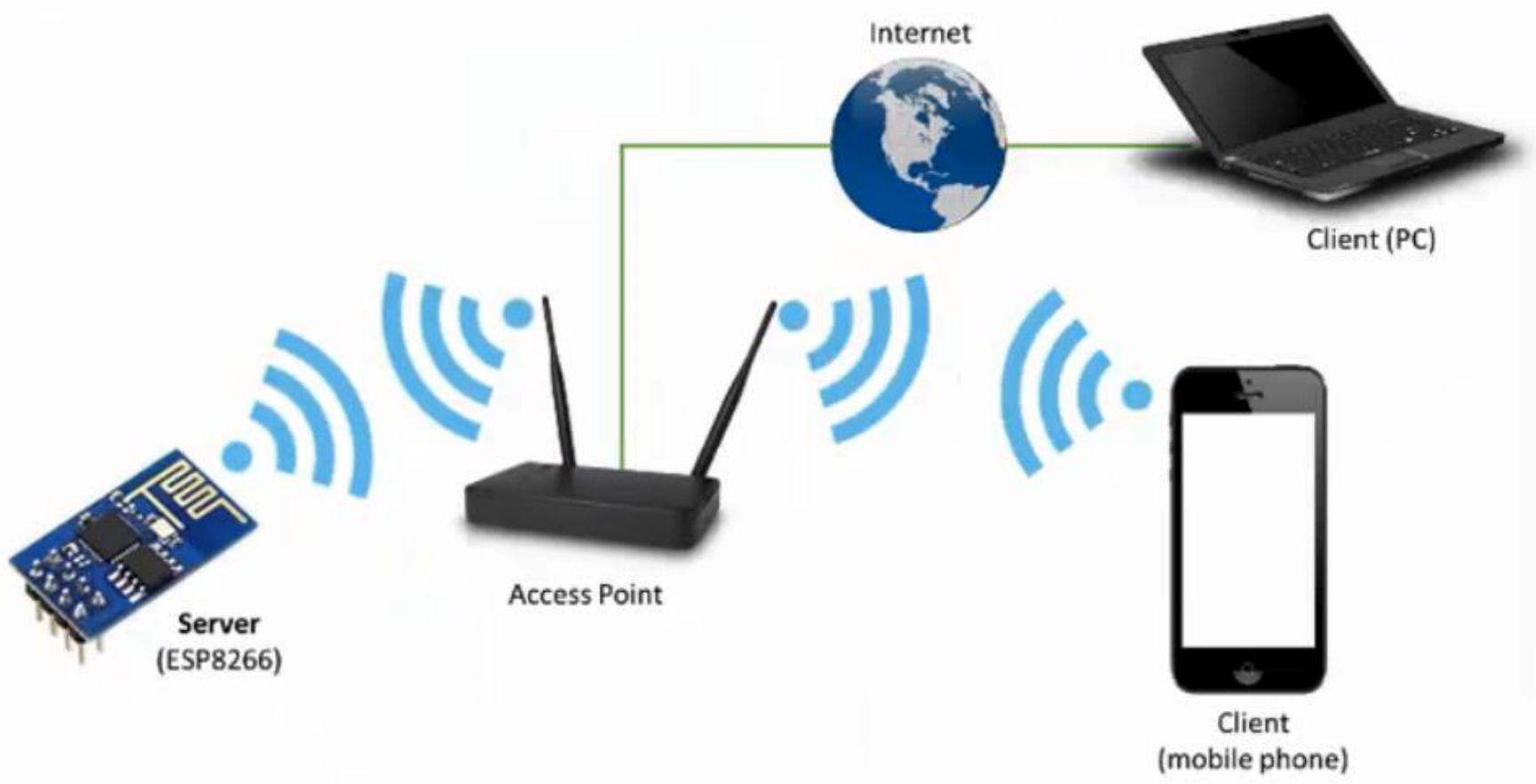


در اینجا سرور دارای IP , Port است که esp8266 می‌تواند از طریق آن به سرور متصل شود و در تصویر سمت راست برای Security از TLS1 استفاده می‌کند که یا از طریق فایل Certificate ای که درون SPIFFS است استفاده کنیم یا اینکه از طریق آرایه ای که ایجاد می‌کنیم ، انجام میدهیم.

استاد : صالحی مدرس دانشگاههای برتر استان تهران

Server Class

در اینجا esp8266 میتواند بعنوان یک سرور عمل کند



استاد : صالحی مدرس دانشگاههای پیام نور استان تهران

توضیح کامل متد begin

Below is the syntax of another overload of `begin` with the all possible parameters:

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

Meaning of parameters is as follows:

- `ssid` - a character string containing the SSID of Access Point we would like to connect to, may have up to 32 characters
- `password` to the access point, a character string that should be minimum 8 characters long and not longer than 64 characters
- `channel` of AP, if we like to operate using specific channel, otherwise this parameter may be omitted
- `bssid` - mac address of AP, this parameter is also optional
- `connect` - a `boolean` parameter that if set to `false`, will instruct module just to save the other parameters without actually establishing connection to the access point

استاد : صالحی مدرس دانشگاههای برتر استان تهران

config

Disable DHCP client (Dynamic Host Configuration Protocol) and set the IP configuration of station interface to user defined arbitrary values. The interface will be a static IP configuration instead of values provided by DHCP.

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

خروجی های WiFi status

status

Return the status of Wi-Fi connection.

```
WiFi.status()
```

Function returns one of the following connection statuses:

- `WL_CONNECTED` after successful connection is established
- `WL_NO_SSID_AVAIL`in case configured SSID cannot be reached
- `WL_CONNECT_FAILED` if password is incorrect
- `WL_IDLE_STATUS` when Wi-Fi is in process of changing between statuses
- `WL_DISCONNECTED` if module is not configured in station mode

استاد : صالحی مدرس دانشگاههای برتر استان تهران

برنامه شماره ۱ : برنامه ای برای نمایش IP ماژول ESP8266 (حالت DHCP)

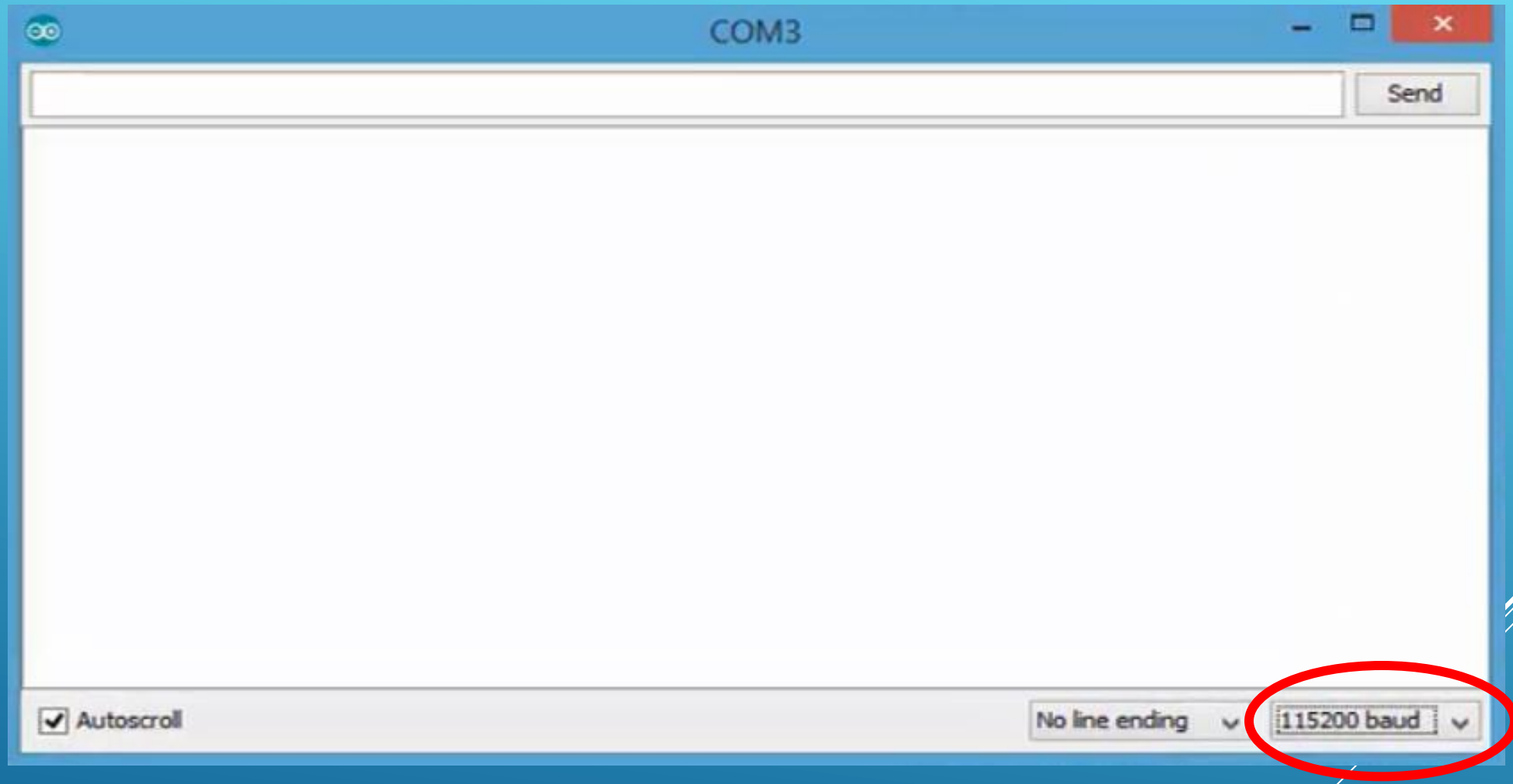
```
1  #include <ESP8266WiFi.h>
2  char* ssid="IRANIOT";
3  char* pass="12345678";
4  void setup() {
5    // put your setup code here, to run once:
6    Serial.begin(115200);
7    WiFi.begin(ssid,pass);
8    While(WiFi.status()!=WL_CONNECTED)
9    {
10     Serial.print(".");
11     delay(500);
12   }
13   Serial.println("");
14   Serial.print("ESP IP address");
15   Serial.print(WiFi.localIP());
16 }
17 void loop() {
18   // put your main code here, to run repeatedly:
19 }
```

در اینجا دستور WiFi.mode(WIFI.STA) را اصلا ننوشتیم بنابراین به طور اتوماتیک esp8266 در مود station قرار می‌گیرد

تا زمانیکه به وای فای متصل نشده ، نقطه چاپ می‌کند.

در حقیقت در اینجا IP ای که از DHCP Server گرفته را چاپ می‌کند.

استاد : صالحی مدرس دانشگاههای برتر استان تهران


COM3
Send
Autoscroll
No line ending
115200 baud

برنامه شماره ۲ : برنامه ای برای نمایش IP ماژول ESP8266 (در حالت بدون DHCP)

```
#include <ESP8266WiFi.h>
char* ssid="IRANIOT";
char* pass="12345678";
IPAddress staticIP(192,168,0,200);
IPAddress subnetmask(255,255,255,0);
IPAddress gateway(192,168,0,1);
IPAddress dns1(4,2,2,4);
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  WiFi.begin(ssid,pass);
  WiFi.config(staticIP,gateway,subnetmask,dns1);
  While(WiFi.status()!=WL_CONNECTED)
  {
    Serial.print(".");
    delay(500);
  }
  Serial.println("");
  Serial.print("ESP IP address");
  Serial.print(WiFi.localIP());
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

این اسامی هر چه میتواند باشد
و مهم نیست که حتما به صورت زیر باشد

ترتیب باید رعایت شود
ضمنا پس از DNS1 ، اگر
DNS2 داشتیم ، آنرا
میتوان نوشت.

اصولا در station هایی که آی پی استاتیک داشته باشند ، سریعتر به شبکه متصل میشوند
بدلیل اینکه اختصاص IP از طریق DHCP کمی زمانبر است.

استاد : صالحی مدرس دانشگاههای برتر استان تهران

برنامه شماره ۳ : برنامه ای برای نمایش IP ماژول ESP8266

```
#include <ESP8266WiFi.h>

const char* ssid = "********";
const char* password = "********";

IPAddress staticIP(192,168,1,22);
IPAddress gateway(192,168,1,9);
IPAddress subnet(255,255,255,0);

void setup(void)
{
  Serial.begin(115200);
  Serial.println();

  Serial.printf("Connecting to %s\n", ssid);
  WiFi.begin(ssid, password);
  WiFi.config(staticIP, gateway, subnet);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.print("Connected, IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {}
```

به متد printf دقت شود

استاد : صالحی مدرس دانشگاههای برتر استان تهران

دستور( )WiFi.disconnect را در ابتدای منطقه Setup برنامه می‌نویسیم تا تنظیمات TCP/IP ریست شود.

disconnect

Sets currently configured SSID and password to `null` values and disconnects the station from an access point.

```
WiFi.disconnect(wifioff)
```

The `wifioff` is an optional `boolean` parameter. If set to `true`, then the station mode will be turned off.

isConnected

Returns `true` if Station is connected to an access point or `false` if not.

```
WiFi.isConnected()
```

setAutoConnect

Configure module to automatically connect on power on to the last used access point.

```
WiFi.setAutoConnect(autoConnect)
```

The `autoConnect` is an optional parameter. If set to `false` then auto connection functionality up will be disabled. If omitted or set to `true`, then auto connection will be enabled.

getAutoConnect

This is "companion" function to `setAutoConnect()`. It returns `true` if module is configured to automatically connect to last used access point on power on.

```
WiFi.getAutoConnect()
```

If auto connection functionality is disabled, then function returns `false`.

setAutoReconnect

Set whether module will attempt to reconnect to an access point in case it is disconnected.

```
WiFi.setAutoReconnect(autoReconnect)
```

If parameter `autoReconnect` is set to `true`, then module will try to reestablish lost connection to the AP. If set to `false` then module will stay disconnected.
Note: running `setAutoReconnect(true)` when module is already disconnected will not make it reconnect to the access point. Instead `reconnect()` should be used.

waitForConnectResult

Wait until module connects to the access point. This function is intended for module configured in station or station + soft access point mode.

```
WiFi.waitForConnectResult()
```

Function returns one of the following connection statuses:

- `WL_CONNECTED` after successful connection is established
- `WL_NO_SSID_AVAIL`in case configured SSID cannot be reached
- `WL_CONNECT_FAILED` if password is incorrect
- `WL_IDLE_STATUS` when Wi-Fi is in process of changing between statuses
- `WL_DISCONNECTED` if module is not configured in station mode

Configuration

macAddress

Get the MAC address of the ESP station's interface.

```
WiFi.macAddress(mac)
```

Function should be provided with `mac` that is a pointer to memory location (an `uint8_t` array the size of 6 elements) to save the mac address. The same pointer value is returned by the function itself.
Example code:

```
if (WiFi.status() == WL_CONNECTED)
{
  uint8_t macAddr[6];
  WiFi.macAddress(macAddr);
  Serial.printf("Connected, mac address: %02x:%02x:%02x:%02x:%02x:%02x\n", macAddr[0], macAddr[1],
macAddr[2], macAddr[3], macAddr[4], macAddr[5]);
}
```

Example output:

```
Mac address: 5C:CF:7F:08:11:17
```

If you do not feel comfortable with pointers, then there is optional version of this function available. Instead of the pointer, it returns a formatted `String` that contains the same mac address.

```
WiFi.macAddress()
```

Example code:

```
if (WiFi.status() == WL_CONNECTED)
{
  Serial.printf("Connected, mac address: %s\n", WiFi.macAddress().c_str());
}
```

localIP

Function used to obtain IP address of ESP station's interface.

```
WiFi.localIP()
```

The type of returned value is IPAddress. There is a couple of methods available to display this type of data. They are presented in examples below that cover description of `subnetMask`, `gatewayIP` and `dnsIP` that return the IPAdress as well.

Example code:

```
if (WiFi.status() == WL_CONNECTED)
{
  Serial.print("Connected, IP address: ");
  Serial.println(WiFi.localIP());
}
```

Example output:

```
Connected, IP address: 192.168.1.10
```

subnetMask

Get the subnet mask of the station's interface.

```
WiFi.subnetMask()
```

Module should be connected to the access point to obtain the subnet mask.

Example code:

```
Serial.print("Subnet mask: ");
Serial.println(WiFi.subnetMask());
```

Example output:

```
Subnet mask: 255.255.255.0
```

gatewayIP

Get the IP address of the gateway.

```
WiFi.gatewayIP()
```

Example code:

```
Serial.printf("Gataway IP: %s\n", WiFi.gatewayIP().toString().c_str());
```

Example output:

```
Gataway IP: 192.168.1.9
```

dnsIP

Get the IP addresses of Domain Name Servers (DNS).

```
WiFi.dnsIP(dns_no)
```

With the input parameter `dns_no` we can specify which Domain Name Server's IP we need. This parameter is zero based and allowed values are none, 0 or 1. If no parameter is provided, then IP of DNS #1 is returned.

Example code:

```
Serial.print("DNS #1, #2 IP: ");
WiFi.dnsIP().printTo(Serial);
Serial.print(", ");
WiFi.dnsIP(1).printTo(Serial);
Serial.println();
```

Example output:

```
DNS #1, #2 IP: 62.179.1.60, 62.179.1.61
```

hostname

Get the DHCP hostname assigned to ESP station.

```
WiFi.hostname()
```

Function returns `String` type. Default hostname is in format `ESP_24xMAC`where 24xMAC are the last 24 bits of module's MAC address.
The hostname may be changed using the following function:

```
WiFi.hostname(aHostname)
```

Input parameter `aHostname` may be a type of `char*`, `const char*` or `String`. Maximum length of assigned hostname is 32 characters. Function returns either `true` or `false` depending on result. For instance, if the limit of 32 characters is exceeded, function will return `false` without assigning the new hostname.
Example code:

```
  Serial.printf("Default hostname: %s\n", WiFi.hostname().c_str());
  WiFi.hostname("Station_Tester_02");
Serial.printf("New hostname: %s\n", WiFi.hostname().c_str());
```

Example output:

```
Default hostname: ESP_081117
New hostname: Station_Tester_02
```

Returned value is type of `wl_status_t` defined in wl_definitions.h

Example code:

```
#include <ESP8266WiFi.h>

void setup(void)
{
  Serial.begin(115200);
  Serial.printf("Connection status: %d\n", WiFi.status());
  Serial.printf("Connecting to %s\n", ssid);
  WiFi.begin(ssid, password);
  Serial.printf("Connection status: %d\n", WiFi.status());
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.printf("\nConnection status: %d\n", WiFi.status());
  Serial.print("Connected, IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {}
```

Example output:

```
Connection status: 6
Connecting to sensor-net
Connection status: 6
......
Connection status: 3
Connected, IP address: 192.168.1.10
```

Particular connection statuses 6 and 3 may be looked up in wl_definitions.h as follows:

```
3 - WL_CONNECTED
6 - WL_DISCONNECTED
```

Basing on this example, when running above code, module is initially disconnected from the network and returns connection status `6 - WL_DISCONNECTED`. It is also disconnected immediately after running `WiFi.begin(ssid, password)`. Then after about 3 seconds (basing on number of dots displayed every 500ms), it finally gets connected returning status `3 - WL_CONNECTED`.

SSID

Return the name of Wi-Fi network, formally called Service Set Identification (SSID).

```
WiFi.SSID()
```

Returned value is of the `String` type.
Example code:

```
Serial.printf("SSID: %s\n", WiFi.SSID().c_str());
```

Example output:

```
SSID: sensor-net
```

psk

Return current pre shared key (password) associated with the Wi-Fi network.

```
WiFi.psk()
```

Function returns value of the `String` type.

Serial.print(WiFi.psk());

توسعه یافته شده پروژه وای فای

```
#include <ESP8266WiFi.h>

char* ssid="IRANIOT";
char* pass = "12345678";
String str(ssid);  //convert char to string to print easy
IPAddress staticIP(192,168,1,100);
IPAddress subnetmask(255,255,255,0);
IPAddress gateway(192,168,1,1);
IPAddress dns1(4,2,2,4);

void setup() {
  WiFi.disconnect();
  Serial.begin(115200);
 WiFi.begin(ssid,pass);
 WiFi.config(staticIP,gateway,subnetmask,dns1);
  Serial.println(" ");
 Serial.println("connecting to (("+str+")) Access Point");  //connection Notification
  while(WiFi.status()!= WL_CONNECTED)
  {
    Serial.print(".");
    delay(500);
  }
  Serial.println(" ");
   Serial.println("Yeeep connected with these Parameters :"); //connection Notification
   Serial.print("ESP IP Address: ");
   Serial.println(WiFi.localIP());
   Serial.print("Subnet mask: ");
   Serial.println(WiFi.subnetMask());
   Serial.print("Gateway: ");
   Serial.println(WiFi.gatewayIP());
   Serial.print("DNS: ");
   Serial.println(WiFi.dnsIP());
   Serial.print("Host Name: ");
   Serial.println(WiFi.hostname());
  }


void loop() {
  // put your main code here, to run repeatedly:
```

استاد : صالحی مدرس دانشگاههای برتر استان تهران