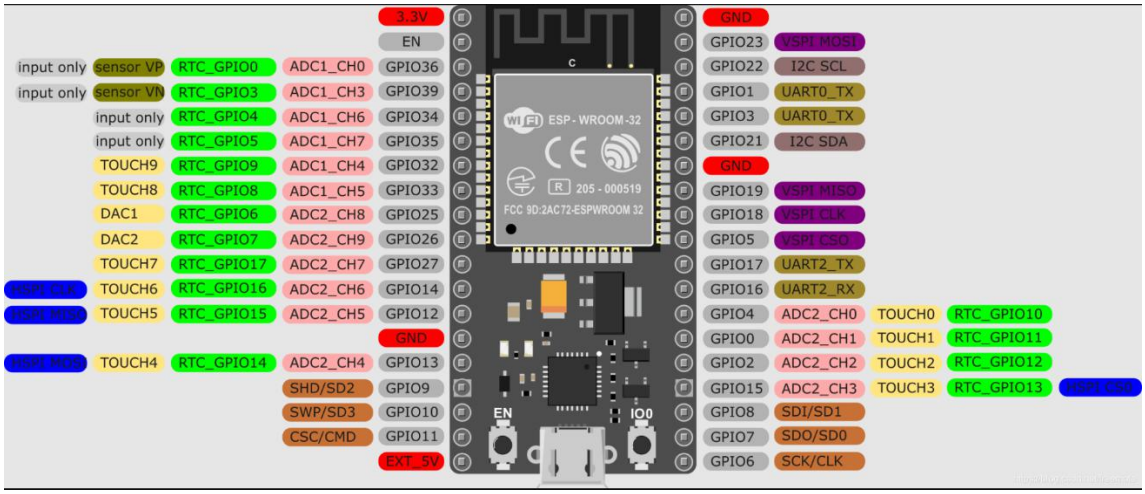
ESP32

一、了解开发板

（一）认识引脚

认识开发板才能不接错线，否则，把线接错了也没有作用。



（二）、点亮开发板 LED

拿到开发板第一步应该是点亮开发板 LED，因为这样可以确定开发板是否能正常工作，如果开发板不能正确工作，再多的想法都不可能实现。

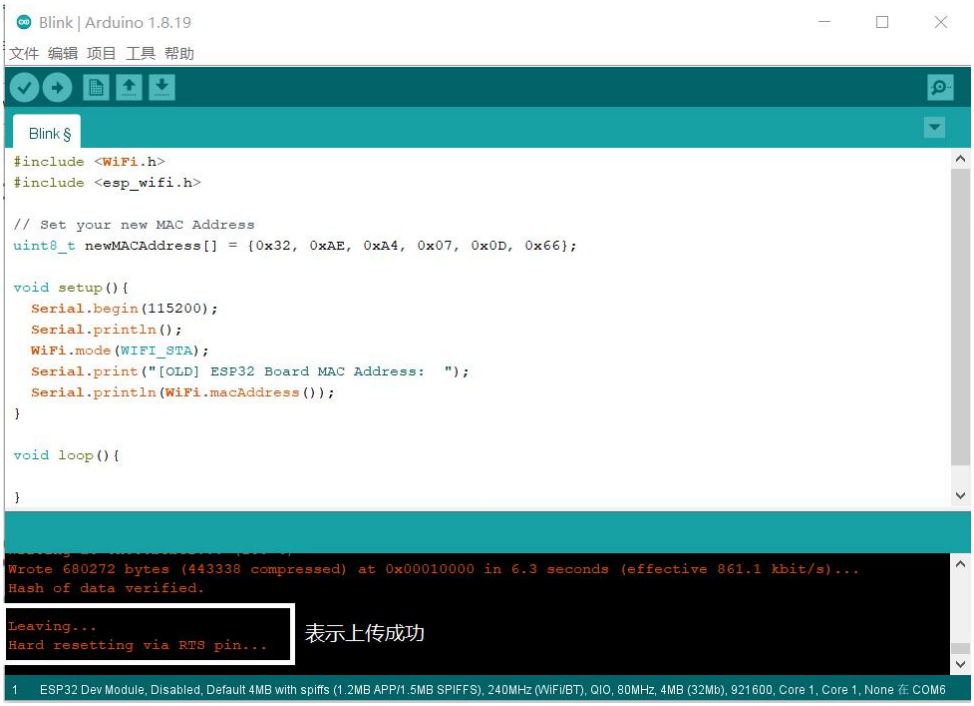
```
int LED=13;

void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED, OUTPUT);
}

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

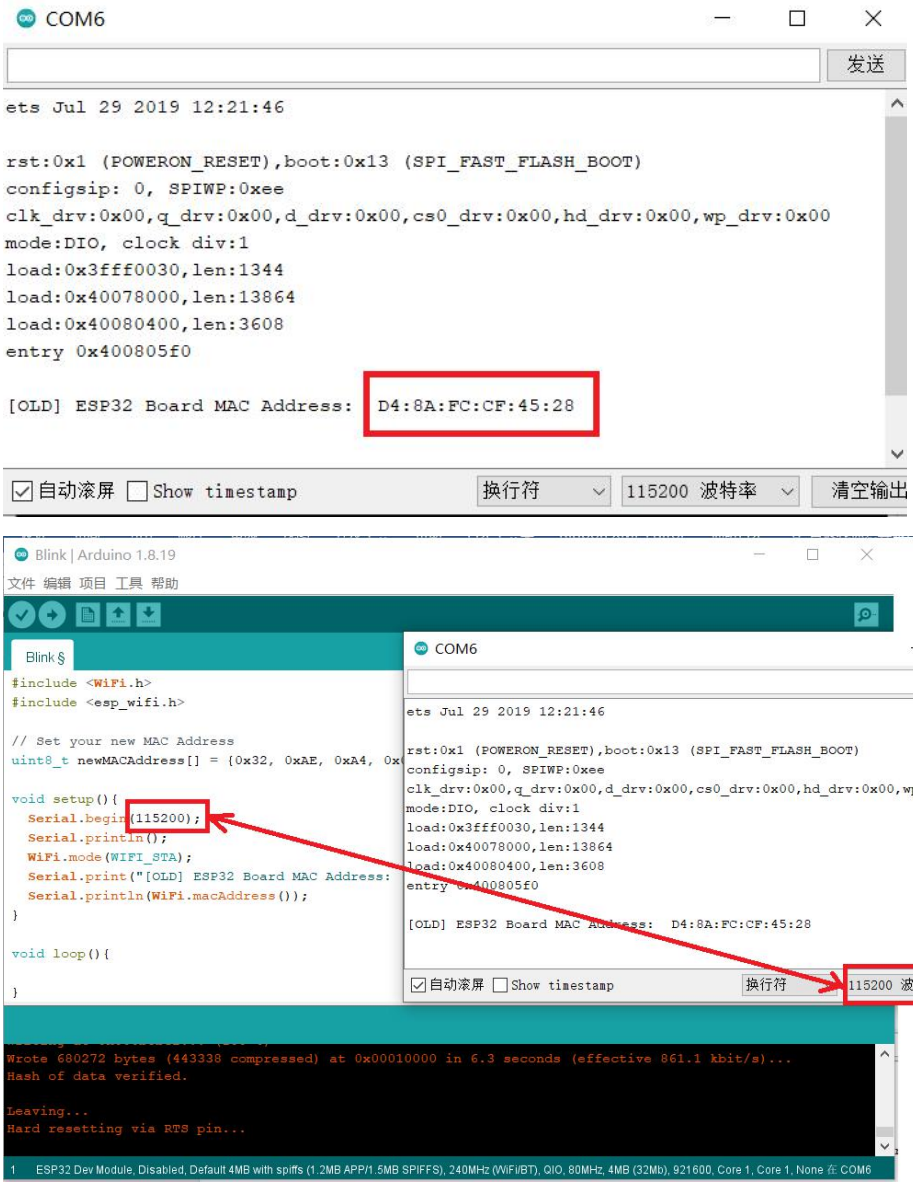
```
  delay(1000);                    // wait for a second
}
```

（三）、查看 ESP32MAC 地址

有的路由器设置了 MAC 地址黑白名单联网要求，这个时候，必须在路由器登录页面里设置 MAC，否则就连不上网，无法进行后面的工作。

```
#include <WiFi.h>
#include <esp_wifi.h>

// Set your new MAC Address
uint8_t newMACAddress[] = {0x32, 0xAE, 0xA4, 0x07, 0x0D, 0x66};

void setup(){
  Serial.begin(115200);
  Serial.println();
  WiFi.mode(WIFI_STA);
  Serial.print("[OLD] ESP32 Board MAC Address:  ");
  Serial.println(WiFi.macAddress());
}

void loop(){

}
```





注意串口值要对应。

二、局域网

局域网在实验教学中基本没什么用，只是可以在办公室里试试各种感应器，但是后面的配置也要用到，所以从局域网开始。

（一）联网

局域网要用到 WiFi.h，默认情况下，arduino IDE 安装好 ESP32 就包含了该库的，直接：

#include <WiFi.h>

就可以使用该库了。

主要的函数有:

一、WiFi.mode();设置配网模式

static bool mode(wifi_mode_t);

wifi_mode_t 的四个选项	
WIFI_OFF	关闭配网模式
WIFI_STA	设置为 STA 模式
WIFI_AP	设置为 AP 模式
WIFI_AP_STA	设置为 AP 和 STA 共存模式

（1）别的设备连接 ESP32——AP 配网

ESP32 的 AP 配网模式可以通过无线 WIFI 连接的方式来连接来控制 ESP32 或获取 ESP32 的数据。

函数：

（1）WiFi.softAP();

设置 AP 的 WIFI 属性

bool softAP(const char* ssid, const char* passphrase = NULL, int channel = 1, int ssid_hidden = 0, int max_connection = 4, bool ftm_responder = false);

ssid	设置 SSID
passphrase	设置密码
channel	设置通道，默认为 1
ssid_hidden	是否隐藏，默认为 0 不隐藏
max_connection	最大连接数量，默认为 4
ftm_responder	测试响应，默认为 false

（2）WiFi.softAPConfig();

设置 AP 的 IP，网关，子网掩码，DHCP

bool softAPConfig(IPAddress local_ip, IPAddress gateway, IPAddress subnet, IPAddress dhcp_lease_start = INADDR_NONE);

local_ip	设置 IP 地址
gateway	设置网关
subnet	设置子网掩码
dhcp_lease_start	设置 DHCP,默认为打开

最简单的配网就这三个函数实现，完整代码：

```
#include <WiFi.h>
IPAddress AP_local_ip(10,0,1,1);              //IP 地址
IPAddress AP_gateway(10,0,1,1);                 //网关地址
IPAddress AP_subnet(255,255,255,0);           //子网掩码
const char* AP_ssid = "shuyang";              //SSID
```

```
const char* AP_password = "12345678";          //密码

void setup() {
   WiFi.mode(WIFI_AP);
   WiFi.softAPConfig(AP_local_ip, AP_gateway, AP_subnet);
   WiFi.softAP(AP_ssid, AP_password);
}

void loop() {
}
```

把以上代码上传到 ESP32 后，打开电脑或手机上的 WIFI 连接界面，就可以看到一个名称为 shuyang 的路由器，这时就可以输入密码"12345678"进行连接测试了。

（2）ESP32 连接路由器——STA 模式

主要的函数有：

一、WiFi.begin();

初始化 WIFI 连接

wl_status_t begin(const char* ssid, const char *passphrase = NULL, int32_t channel = 0, const uint8_t* bssid = NULL, bool connect = true);

ssid	SSID
passphrase	密码
channel	通道
bssid	BSSID，对应 MAC 地址，默认为 NULL
connect	是否连接，默认为 true

二、WIFI.status()

wl_status_t status()

获取 WIFI 的连接状态

参数	无	
返回	WL_CONNECT_FAILED	未连接
	WL_CONNECTED	已连接

三、WIFI.localIP()

wl_status_t localIP()　　-获取 ESP32 的本地 IP 地址

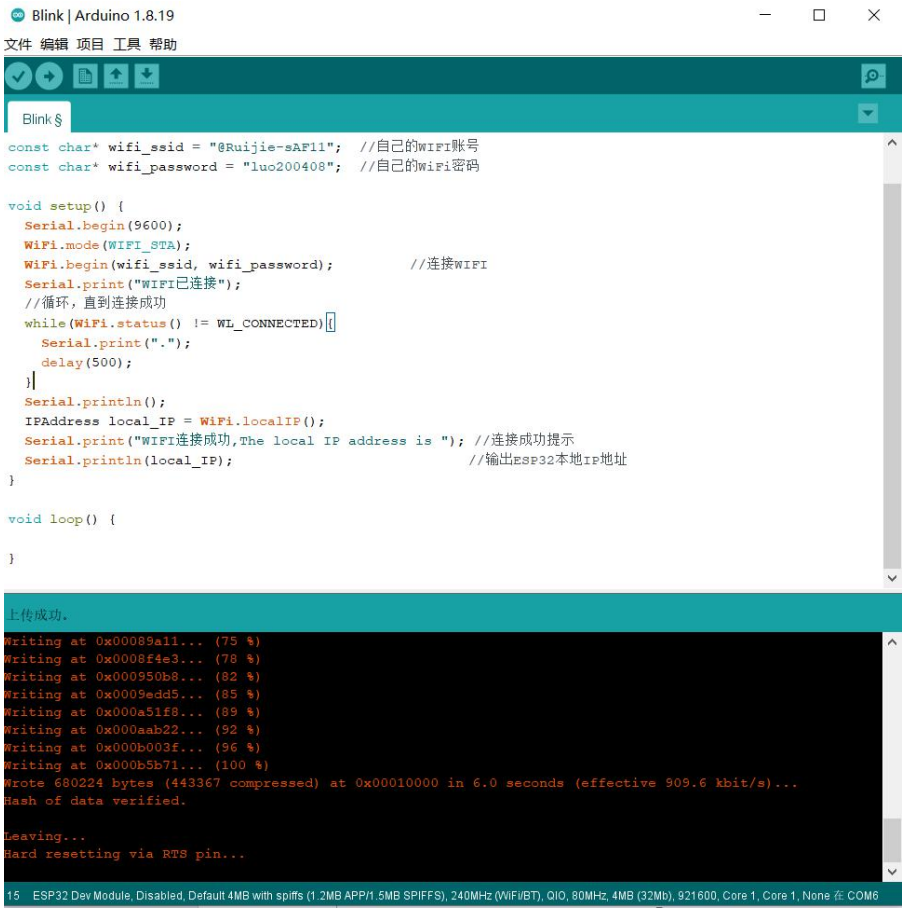
参数	无	
返回	IPAddress	IP 地址

最简单的连接 WIFI 的完整代码：

注意，代码内的，WIFI 连接名称和密码需要根据你的路由名称和密码做出更改

```
#include <WiFi.h>

const char* wifi_ssid = "@Ruijie-sAF11";   //自己的 WIFI 账号
const char* wifi_password = "luo200408";   //自己的 WiFi 密码

void setup() {
```

```
    Serial.begin(9600);
    WiFi.mode(WIFI_STA);
    WiFi.begin(wifi_ssid, wifi_password);              //连接 WIFI
    Serial.print("WIFI 已连接");
    //循环，直到连接成功
    while(WiFi.status() != WL_CONNECTED){
        Serial.print(".");
        delay(500);
    }
    Serial.println();
    IPAddress local_IP = WiFi.localIP();
    Serial.print("WIFI 连接成功,The local IP address is "); //连接成功提示
    Serial.println(local_IP);                                  //输出 ESP32 本地 IP 地址
}

void loop() {

}
```



以上代码上传到 ESP32 前注意打开串口监视器，如果连接成功，将会输出连接成功的提示。

现在没有内容还不能显示

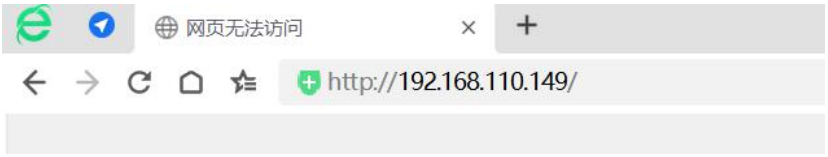
（3）AP 模式与 STA 模式共存

在该模式下，可以同时接受 WIFI 连接到 ESP32,也可以把 ESP32 连接到已有的 WIFI 上。
简单的 AP 模式与 STA 模式共存完整代码：

```
#include <WiFi.h>

IPAddress AP_local_ip(10,0,10,1);                //IP 地址
IPAddress AP_gateway(10,0,10,1);                  //网关地址
IPAddress AP_subnet(255,255,255,0);              //子网掩码
const char* AP_ssid = "shuyang";       //SSID
const char* AP_password = "12345678";     //密码

const char* wifi_ssid = "@Ruijie-sAF11";       //不要与上面的混淆，这是真实的 WIFI 账号
const char* wifi_password = "luo200408";     //不要与上面的混淆，这是真实的 WIFI 密码

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_AP_STA);
  WiFi.softAPConfig(AP_local_ip, AP_gateway, AP_subnet);
  WiFi.softAP(AP_ssid, AP_password);
  WiFi.begin(wifi_ssid, wifi_password);                //连接 WIFI
  Serial.print("Connected");
  //循环，直到连接成功
  while(WiFi.status() != WL_CONNECTED){
```

```
        Serial.print(".");
        delay(500);
    }
    Serial.println();
    IPAddress local_IP = WiFi.localIP();
    Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
    Serial.println(local_IP);                                   //输出本地 IP 地址
}

void loop() {

}
```

测试方法和之前的文章一样，可以用手机或电脑来对 AP 模式进行测试。



浏览器里没内容。

WIFI 里多了一个 shuyang 的 WIFI。

小结

WiFi.mode(WIFI_STA、WIFI_AP、WIFI_AP_STA);	WiFi.mode(WIFI_AP);	softAP(const char* ssid, const char* passphrase = NULL, int channel = 1, int ssid_hidden = 0, int max_connection = 4, bool ftm_responder = false);
		softAPConfig(IPAddress local_ip, IPAddress gateway, IPAddress subnet, IPAddress dhcp_lease_start = INADDR_NONE);
	WiFi.mode(WIFI_STA);	WiFi.begin();
		WIFI.status()
		WIFI.localIP()
	WiFi.mode(WIFI_AP_STA);	

都是先设置 WIFI 模式，根据不同模式有不同的函数。

（二）、上传内置网页

（1）库的安装

1)、必备库

常用的 EPS32 可用于 WEB 服务器的库有两个：

WebServer

ESPAsyncWebServer

默认情况下，arduino IDE 安装好就包含了 WebServer 这个库的，直接:

#include <WebServer.h>

就可以使用该库了。

2)、安装另外的必须库

1.下载 ESPAsyncWebServer、AsyncTCP 库

ESPAsyncWebServe 库依赖 AsyncTCP 库,所以，这两个库都需要另行添加，这两个库的项目地址分别为：

https://github.com/me-no-dev/ESPAsyncWebServer

https://github.com/me-no-dev/AsyncTCP

进入项目地址后，点击 Clone=》Download ZIP

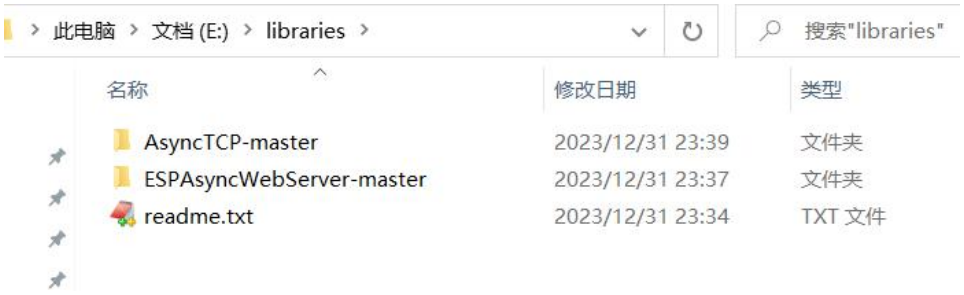
（百度网盘链接）

2.下载好后把这两个库放在任意文件夹都可以

3.安装

打开 arduino IDE,菜单=》项目=》加载库=》添加.ZIP 库...，重复两次，分别选择上述的两个库进行添加。

安装好后，可以在项目文件夹里看到好像是解压的文件



所以，前面的项目文件不要随便设置。
为了方便测试，该文是选择使用 STA 模式。

（2）内置文本

（1）引入需要用到的库：

```
#include <WiFi.h>
#include "ESPAsyncWebServer.h"
AsyncWebServer server(80);
```

（2）连接 WIFI

```
void connect_wifi(){
    const char* wifi_ssid = "@Ruijie-sAF11";       //不要与上面的混淆，这是真实的 WIFI 账号
    const char* wifi_password = "luo200408";     //不要与上面的混淆，这是真实的 WIFI 密码
    Serial.begin(9600);
    WiFi.begin(wifi_ssid, wifi_password);              //连接 WIFI
    Serial.print("Connected");
    //循环，直到连接成功
    while(WiFi.status() != WL_CONNECTED){
      Serial.print(".");
      delay(500);
    }
    Serial.println();
    IPAddress local_IP = WiFi.localIP();
    Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
    Serial.println(local_IP);
}
```

在 setup()里调用该函数。

（3）创建一个 WebServer 对象

```
AsyncWebServer server(80);
```

这行代码是调用了 ESPAsyncWebServer.h 库文件(详细在下面的 ESPAsyncWebServer.h 文件片断中)中的 AsyncWebServer(uint16_t port)，来创建一个 server 对象，port 参数为连接端口。

（4）ESPAsyncWebServer.h 解析

【1】

```
class AsyncWebServer {
```

```
protected:
    AsyncServer _server;
    LinkedList<AsyncWebRewrite*> _rewrites;
    LinkedList<AsyncWebHandler*> _handlers;
    AsyncCallbackWebHandler* _catchAllHandler;

public:
    AsyncWebServer(uint16_t port);
    ~AsyncWebServer();

    void begin();
    void end();
```

【2】

然后是需要注册一个响应链接"/"上的 GET 请求的处理回调函数

```
server.on("/",HTTP_GET,call_back);
```

这行代码是当收到根目录"/"的 GET 请求时，调用 call_back 这个回调函数来响应请求,这个函数内容为:

```
void call_back(AsyncWebServerRequest *request){
   request->send(200,"text/plain","课堂小实验");
}
```

这个回调函数默认需要传入一个 AsyncWebServerRequest 对象，该对象包含了客户端的请求
send()函数会在收到请求后，发送一个字符串("text/plain")内容("hello esp32 web server")的基本(200)响应。

最后需要初始化服务器:

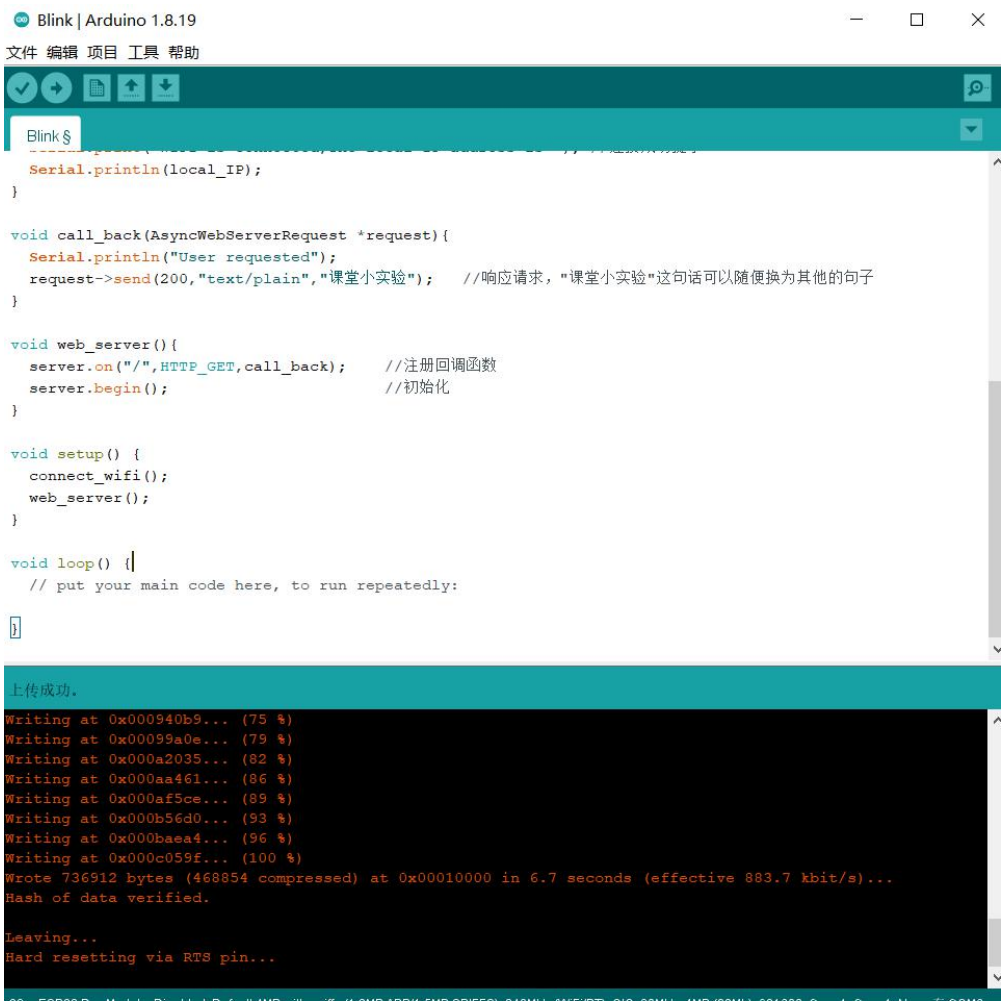
```
server.begin();
```

该方法将启动服务器。

完整代码:

```
#include <WiFi.h>
#include "ESPAsyncWebServer.h"
AsyncWebServer server(80);

//连接 WIFI
void connect_wifi(){
  const char* wifi_ssid = "@Ruijie-sAF11";       //自己的 WIFI 账号
  const char* wifi_password = "luo200408";     //自己的 WIFI 密码
  Serial.begin(9600);
  WiFi.begin(wifi_ssid, wifi_password);              //连接 WIFI
  Serial.print("Connected");
  //循环，直到连接成功
  while(WiFi.status() != WL_CONNECTED){
    Serial.print(".");
```

```
        delay(500);
    }
    Serial.println();
    IPAddress local_IP = WiFi.localIP();
    Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
    Serial.println(local_IP);
}

void call_back(AsyncWebServerRequest *request){
    Serial.println("User requested");
    request->send(200,"text/plain","课堂小实验");     //响应请求，"课堂小实验"这句话可以随便换为其他的句子
}

void web_server(){
    server.on("/",HTTP_GET,call_back);       //注册回调函数
    server.begin();                                  //初始化
}

void setup() {
    connect_wifi();
    web_server();
}

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

}
```



把代码上传到 ESP32 后，完成 WIFI 的连接后，串口会输出一个本地 IP 地址，把该地址复制到浏览器打开，浏览器将输出"hello esp32 web server"

（3）内置网页

1）网页植于代码

正常的网页，有 HTML、CSS、JavaScript。
对于一个中文网页，需要使用
<meta charset="utf-8">
基本的 HTML 页面已经有了，接下来就是要在 ESP32 的 web 服务器 的基础上，把响应浏览器的请求换成上面的 HTML 代码。
首先，把这段 HTML 代码定义为一个 String 的变量:

```
const String indexHtml PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>EPS32 教程</title>
  </head>
  <body>
    <h1>ESP32 HTML 标题</h1>
    <p>ESP32 HTML 段落</p>
  </body>
</html>
)rawliteral";
```

在这个变量里，使用了 PROGMEM 关键字，这个关键字的目的是将数据存储在 Flash(闪存)中而不是 RAM(运行内存)
同时 R"rawliteral 是一种引入原始字符串的方法。引用该方法，就无需对比如斜杠"\"或换行符"\n"之类的进行转义,非常方便。
然后，在响应请求的回调函数里发送这个变量,同时，发送的数据类型也相应得应该换成"text/html":

```
void call_back(AsyncWebServerRequest *request){
  Serial.println("User requested");
  request->send(200,"text/html",indexHtml);     //响应请求
}
```

至此，就创建了一个基本的 HTML 服务器，完整的代码:

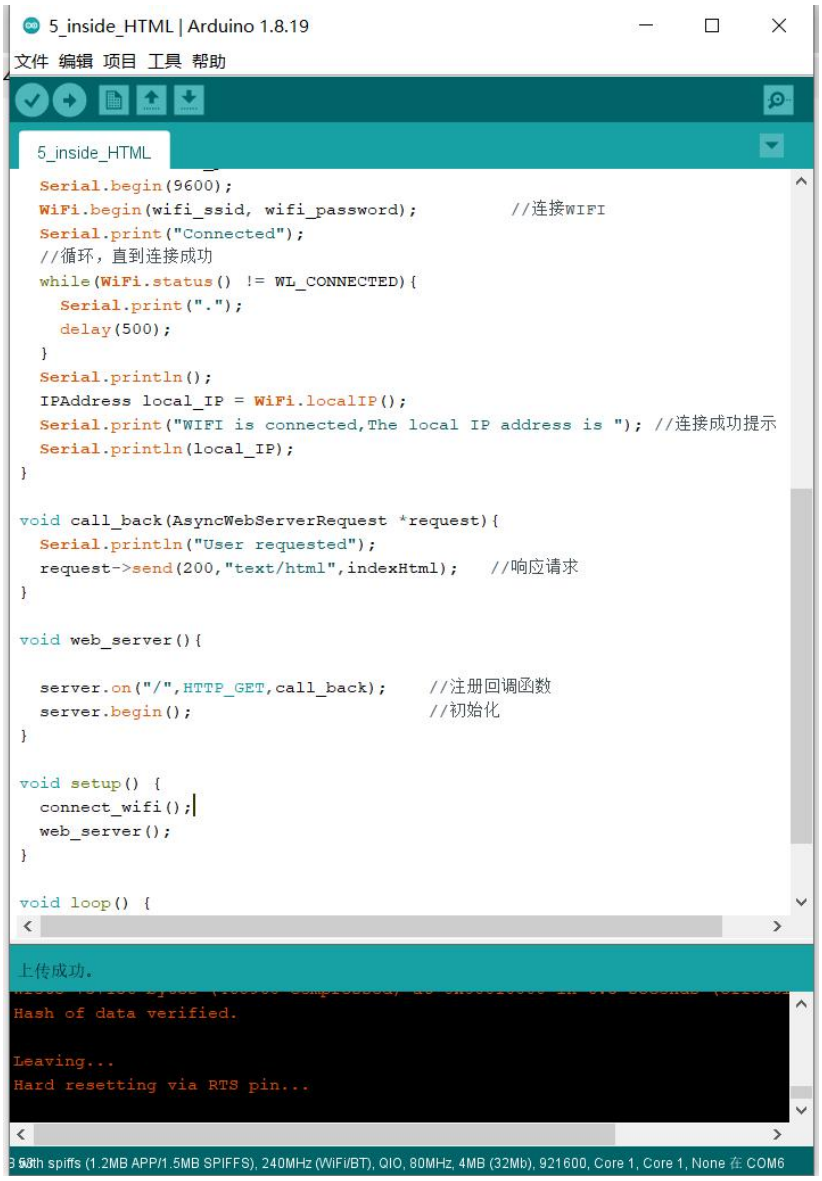
```
#include <WiFi.h>
#include "ESPAsyncWebServer.h"

  AsyncWebServer server(80);
```

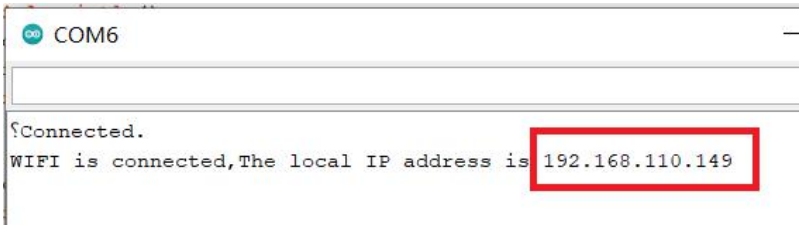
```
const String indexHtml PROGMEM = R"rawliteral(
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>罗孔均</title>
<script type="text/javascript">
alert("阳阳");
</script>
</head>
      <body>
      <h1>课堂物联网实验</h1>
      <h2>用物联网改进课堂教学</h2>
      </body>
</html>
)rawliteral";

//连接 WIFI
void connect_wifi(){
   const char* wifi_ssid = "@Ruijie-sAF11";       //不要与上面的混淆，这是真实的 WIFI 账号
   const char* wifi_password = "luo200408";     //不要与上面的混淆，这是真实的 WIFI 密码
   Serial.begin(9600);
   WiFi.begin(wifi_ssid, wifi_password);              //连接 WIFI
   Serial.print("Connected");
   //循环，直到连接成功
   while(WiFi.status() != WL_CONNECTED){
      Serial.print(".");
      delay(500);
   }
   Serial.println();
   IPAddress local_IP = WiFi.localIP();
   Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
   Serial.println(local_IP);
}

void call_back(AsyncWebServerRequest *request){
   Serial.println("User requested");
   request->send(200,"text/html",indexHtml);     //响应请求
}

void web_server(){
```

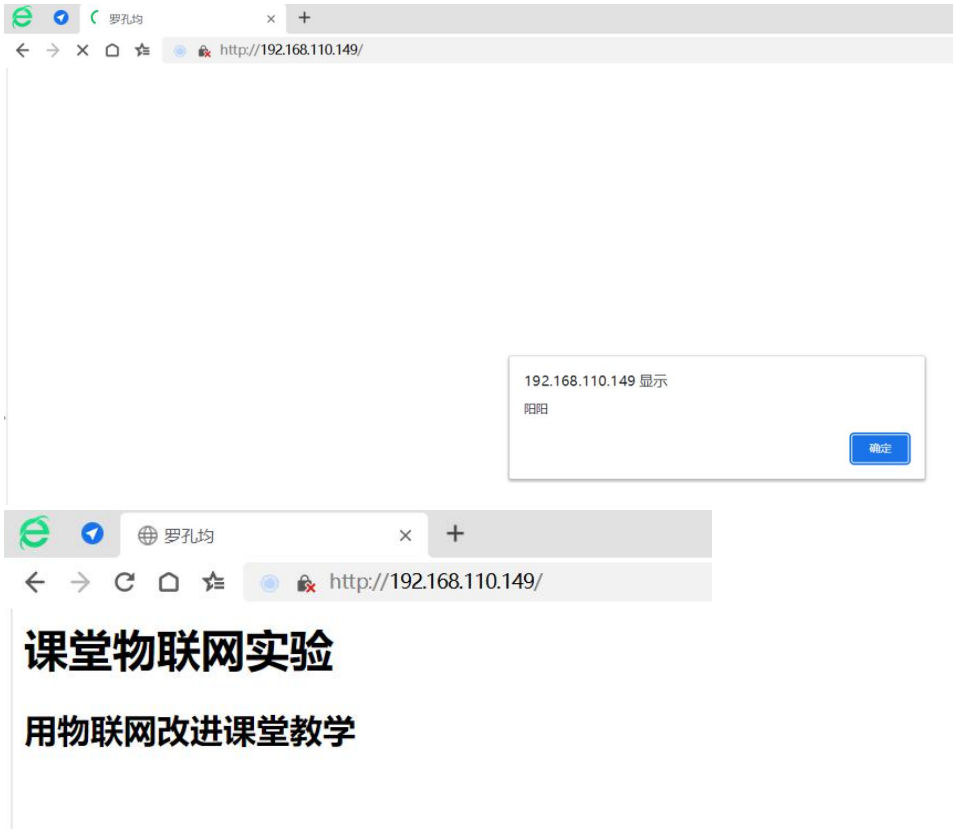
```
    server.on("/",HTTP_GET,call_back);      //注册回调函数
    server.begin();                         //初始化
}

void setup() {
    connect_wifi();
    web_server();
}

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

}
```

```
const String indexHtml PROGMEM = R"rawliteral(
<!DOCTYPE>
<html>

          void call_back(AsyncWebServerRequest *request){
            Serial.println("User requested");
            request->send(200,"text/html",indexHtml);    //响应请求
          }
```

名字要对应，可以随便取英文、拼音，不可以是中文


5_inside_HTML | Arduino 1.8.19
文件 编辑 项目 工具 帮助
5_inside_HTML
Serial.begin(9600);
WiFi.begin(wifi_ssid, wifi_password);
//连接WIFI
Serial.print("Connected");
//循环，直到连接成功
while(WiFi.status() != WL_CONNECTED){
Serial.print(".");
delay(500);
}
Serial.println();
IPAddress local_IP = WiFi.localIP();
Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
Serial.println(local_IP);
}
void call_back(AsyncWebServerRequest *request){
Serial.println("User requested");
request->send(200,"text/html",indexHtml);
//响应请求
}
void web_server(){
server.on("/",HTTP_GET,call_back);
//注册回调函数
server.begin();
//初始化
}
void setup() {
connect_wifi();
web_server();
}
void loop() {
上传成功。
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None 在 COM6


COM6
Connected.
WIFI is connected,The local IP address is 192.168.110.149



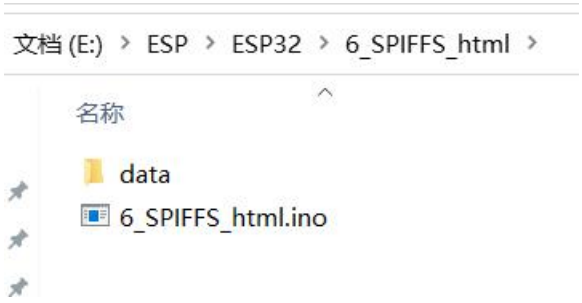
2）SPIFFS 存放前端

【1】SAT 模式——电脑直接访问

SPIFFS 相当于 ESP32 中的一个硬盘分区。

首先，建一个 html 文件(index.html)，文件需要保存在项目文件夹内的"data"文件夹内（不是设置时的项目，而是指 Arduino 代码的文件夹。比如这里的 Arduino 保存为 6_SPIFFS_html，就在这个文件夹里新建一个 data 文件夹）。

注意:，




```
<!DOCTYPE html>
<html>
  <head>
```

```
    <meta charset="utf-8">
    <link href="yilian.css" rel="stylesheet" type="text/css" >
    <script src="yangyang.js"></script>
    <title>罗孔均</title>
  </head>
  <body>
    <h1>课堂物联网</h1>
    <h2>用物联网改进课堂实验</h2>
    <input type="button" value="点击" onclick="shuyang()">
  </body>
</html>
```

然后新建一个 CSS 文件(yilian.css):

```
h1{
color:red;
}
h2{
color:green;
}
```

然后，把文件上传到 ESP32 的 SPIFFS 分区中。

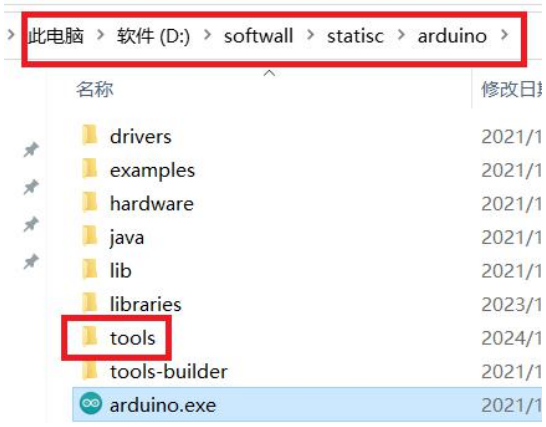
先要在 arduion IDE 中安装一个插件(ESP32FS)，该插件的项目地址在:

https://github.com/me-no-dev/arduino-esp32fs-plugin

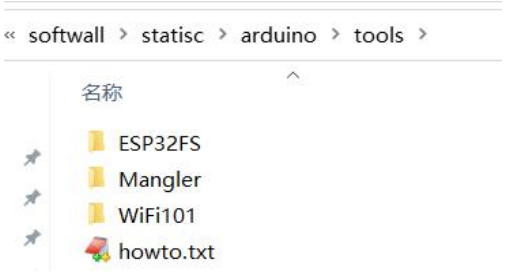
找到 releases page 下载这个插件。

点击下载 ESP32FS-1.1.zip

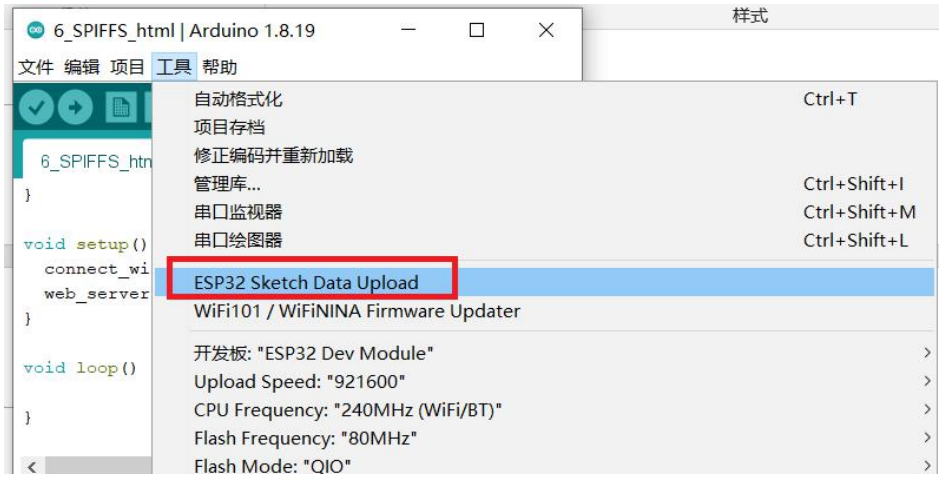
解压缩该插件到 arduino IDE ***安装文件夹***下的 tools 文件夹。



（注意：不是解压后复制最后的文件，而是解压后是一个文件夹，把这个文件夹复制进去。）



重启 arduino IDE 在菜单=>工具 下会出现新的选项菜单:ESP32 Sketch Data Upload



点击后，该插件会把前面提到的"data"文件夹内的所有文件上传到 SPIFFS 分区。

上传之前，一定要注意先把"串口监视器"关闭，关闭对上传端口的占用。
上传完成会输出提示信息:

```
Connecting....
Chip is ESP32-D0WD-V3 (revision 3)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Schem
Crystal is 40MHz
MAC: d4:8a:fc:cf:45:28
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 921600
Changed.
Configuring flash size...
Auto-detected Flash size: 4MB
Flash will be erased from 0x00290000 to 0x003fffff...
Compressed 1507328 bytes to 3318...
Writing at 0x00290000... (100 %)
Wrote 1507328 bytes (3318 compressed) at 0x00290000 in 4.5 seconds (effective
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

至此，文件就上传至 SPIFFS 分区
要使用 ESP32 的 SPIFFS,要先引入 SPIFFS.h,在 arduion IDE 中，如果已经搭建好 ESP32 环境，默认是自带这个库的。
同时需要做的是引入 WiFi.h 和 ESPAsyncWebServer.h 这两个库

```
#include <WiFi.h>
#include <SPIFFS.h>
#include "ESPAsyncWebServer.h"
```

ESPAsyncWebServer 库中，send()结构体提供了直接读取 SPIFFS 中文件的方法。
首先需要对 SPIFFS 进行初始化,为了方便理解，还是和之前一样把一些功能做成函数:

```
void call_back(AsyncWebServerRequest *request){
  if(!SPIFFS.begin(true)){
    Serial.println("An Error has occurred while mounting SPIFFS");
    return;
  }
  request->send(SPIFFS,"/index.html");
}

void web_server(){
  if(!SPIFFS.begin(true)){
    Serial.println("An Error has occurred while mounting SPIFFS");
    return;
```

```
    }
    server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
    server.begin();                                  //初始化
}
```

完整代码:

```
#include <WiFi.h>
#include <SPIFFS.h>
#include "ESPAsyncWebServer.h"


AsyncWebServer server(80);

//连接 WIFI
void connect_wifi(){
    const char* wifi_ssid = "ESP32";
    const char* wifi_password = "12345678";
    Serial.begin(9600);
    WiFi.begin(wifi_ssid, wifi_password);              //连接 WIFI
    Serial.print("Connected");
    //循环，直到连接成功
    while(WiFi.status() != WL_CONNECTED){
        Serial.print(".");
        delay(500);
    }
    Serial.println();
    IPAddress local_IP = WiFi.localIP();
    Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
    Serial.println(local_IP);
}

void web_server(){
    if(!SPIFFS.begin(true)){
        Serial.println("An Error has occurred while mounting SPIFFS");
        return;
    }
    server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
    server.begin();                                  //初始化
}
```

```
void setup() {
   connect_wifi();
   web_server();
}

void loop() {

}
```

【2】以 AP_STA 和域名连接

用 DNSServer 库可以实现用域名，也就是用比较方便记忆的网址的形式来对 ESP32 进行访问。
在 arduion IDE 中，如果已经搭建好 ESP32 环境，默认是自带这个库的:
以前文中的代码为基础，我们增加这个库的引用:

```
#include <WiFi.h>
#include <SPIFFS.h>
#include <DNSServer.h>
#include "ESPAsyncWebServer.h"
```

因为 DNSServer 只能在 AP 模式下才能实现，所以需要开启 AP 和 STA 共存的方式。
之后需要创建一个 DNSServer 实例:

```
DNSServer dnsserver;
```

实现的主要方法：
一、.start();启动 DNSServer 服务

```
bool start(const uint16_t &port, const String &domainName, const IPAddress &resolvedIP);
```

参数：
1:&port 服务端口，DNS 默认端口 53
2:&domainName 域名，如"esp32_ap.com",域名最好不要设置为互联网存在的网址
3:&resolvedIP 解析域名所对应的 IP 地址

二、.processNextRequest()方法，监测客户端的 DNS 请求，需要放到 loop 循环里。
完整代码:

```
#include <WiFi.h>
#include <SPIFFS.h>
```

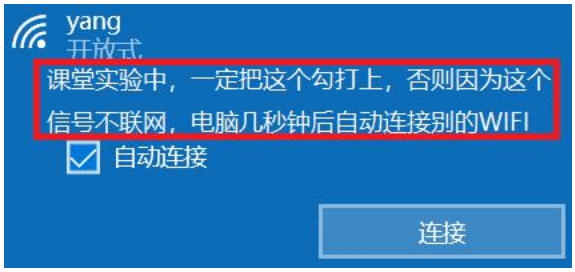
```
#include <DNSServer.h>
#include "ESPAsyncWebServer.h"

DNSServer dnsserver;
AsyncWebServer server(80);

//连接 WIFI
void connect_wifi(){
  const byte DNS_PORT = 53;                          //DNS 端口
  const String url = "yang.com";              //域名
  IPAddress APIp(10,0,10,1);                         //AP IP
  IPAddress APGateway(10,0,10,1);                    //AP 网关
  IPAddress APSubnetMask(255,255,255,0);             //AP 子网掩码
  const char* APSsid = "yang";                    //AP SSID
  const char* APPassword = "";                //AP wifi 密码

  const char* wifi_ssid = "@Ruijie-sAF11";       //不要与上面的混淆，这是真实的 WIFI 账号
  const char* wifi_password = "luo200408";     //不要与上面的混淆，这是真实的 WIFI 密码
  Serial.begin(9600);
  WiFi.mode(WIFI_AP_STA);                                  //打开 AP 和 STA 共存模式
  WiFi.softAPConfig(APIp, APGateway, APSubnetMask);     //设置 AP 的 IP 地址，网关和子网掩码
  WiFi.softAP(APSsid, APPassword, 6);                    //设置 AP 模式的登陆名和密码
  dnsserver.start(DNS_PORT, url, APIp);                 //设置 DNS 的端口、网址、和 IP
  WiFi.begin(wifi_ssid, wifi_password);                 //连接 WIFI
  Serial.print("Connected");
  //循环，直到连接成功
  while(WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  IPAddress local_IP = WiFi.localIP();
  Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
  Serial.println(local_IP);
}

void web_server(){
  if(!SPIFFS.begin(true)){
    Serial.println("An Error has occurred while mounting SPIFFS");
    return;
  }
```

```
    server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
    server.begin();                                    //初始化
}

void setup() {
    connect_wifi();
    web_server();
}

void loop() {
    dnsserver.processNextRequest();
}
```

这个时候，因为电脑已经连接了 ESP32，就无法访问“WIFI is connected,The local IP address is 192.168.110.149”路由器分配给电脑的地址，如图

课堂应用

把实验文件传至 ESP32。
注意：班班通不再连接路由器，而是连接 ESP32 发出的信号，就可以访问了。
电脑连接的时候，注意把自动连接打上勾。



问题解决：

上课不可能带着路由器走，去上课，但是可以带手机走，把手机开热点，让 ESP32 连上手机就可以了；也就是说，把代码里的 WIFI 账号和密码都设为手机的就可以了。

注意：

（1）如果为了简单，在前面设置的时候可以设置为 1,2,3,4——但是要和网关一起变，否则连不上。

（2）在 ESP32 运行代码的时候不能断开手机的 WIFI 连接。

```
//连接WIFI
void connect_wifi(){
  const byte D                                    //DNS端口
                                                  //域名
  IPAddress APIp(1,2,3,4);                        //AP IP
  IPAddress APGateway(1,2,3,4);                   //AP网关
  IPAddress APSubnetMask(255,255,255,0);          //AP子网掩码
```

两个要一样，否则连不上

用上面的技术，基本上可以在课堂上演示用的试验了。