# 使用 **Johnny-Five** 通过 **WiFi** 将 **ESP8266** 连接到 **Blynk** 中（**Firmata WiFi**）

Phodal Huang

October 24, 2017

# 目录

原文链接：https://www.wandianshenme.com/play/raspberrypi-blynk-server-control-esp8266-johnn

在上一个玩法中，我们探索了 Johnny-Five 框架，它允许基于 Arduino、Raspberry Pi（或者 Orange Pi ）上通过 JavaScript 创建连接对象（或者更多）。对于使用 Arduino Nano v3 替代 Orange Pi 的 GPIO，Johnny-Five 也是一个非常好的解决方案。在以前的教程中，我们使用了连接到 USB 端口的 Arduino。而在本玩法中，我们将使用 Arduino IDE 提供的 WiFi Firmata（StandardFirmataWiFi），能让您与 WiFi 中的 ESP8266 进行通信。

## 为什么不使用 **Firmware Serial UDP**

如果您使用 ESP8266 Johnny-Five 关键字进行互联网搜索，您还将找到此 Serial UDP 固件项目（GitHub Gist）。此固件可以以二进制的形式，通过 espressif 提供的 esptool.py 安装。首先，您将需要在计算机上安装 Python 2.7。如果在连接 ESP8266 时遇到困难，这可能是配置错误。由于 GitHub（或另一个平台）不提供源代码，因此建议您使用 firmwareWiFi 固件。它更安装起来更简单，并且完全可配置。

## 步骤 **0**：材料清单

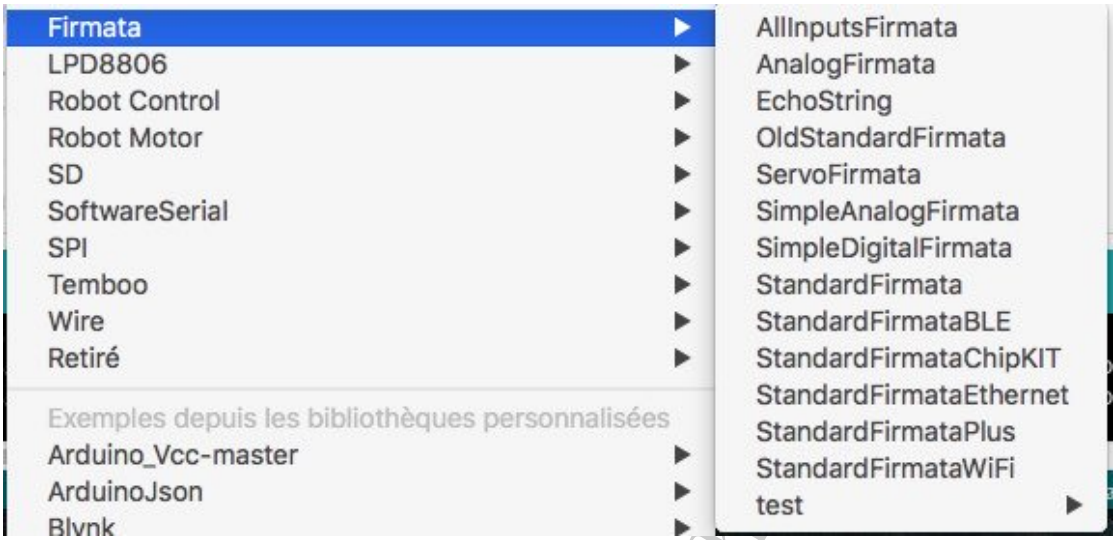
在这里，我们所需要的材料有：

- Wemos D1 Mini
- 5V/3A micro-usb 电源
- BMP180
- LED
- 220Ω 电阻
- 杜邦线
- 面包板

## 步骤 **1**：在 **ESP8266** 上安装 **StandardFirmataWiFi** 固件

StandardFirmataWiFi 固件，可以通过 TCP 串行链接访问所有的 Arduino 功能。它与 Arduino 扩展板、Shield 101、MKR1000 扩展板和所有的 ESP8266 兼容。例如，您可以将 ESP-01 添加到 Arduino Uno 中，这比 WiFi 扩展板便宜得多。

将 ESP8266 连接到计算机，并打开 Arduino IDE。在示例（examples）菜单中，您将找到一个名为 Firmata 的子菜单。从中选择 StandardFirmataWiFi 固件。如下图所示：




缺省/默认情况下，禁止调试。如果要打开调试功能，需要查找包含字符串 #define SERIAL_DEBUG 的那一行，并删除注释（//）。

然后转到 wifiConfig.h 文件配置连接参数：

- 步骤 3（或查找 ssid）：指定 ESP8266 必须要连接的 WiFi 网络
- 步骤 4：指定固定 IP（STATIC_IP_ADDRESS）。在这种情况下，需要设置：子网掩码（SUBNET_MASK）和路由器的 IP（GATEWAY_IP_ADDRESS）
- 步骤 5：SERVER_PORT，服务器连接端口
- 步骤 6：WiFi 网络的安全类型。默认为 WPA（WIFI_WPA_SECURITY），再输入与安全类型相对应的密码（wpa_passphrase for WPA）。

配置完成。上传程序，结束！

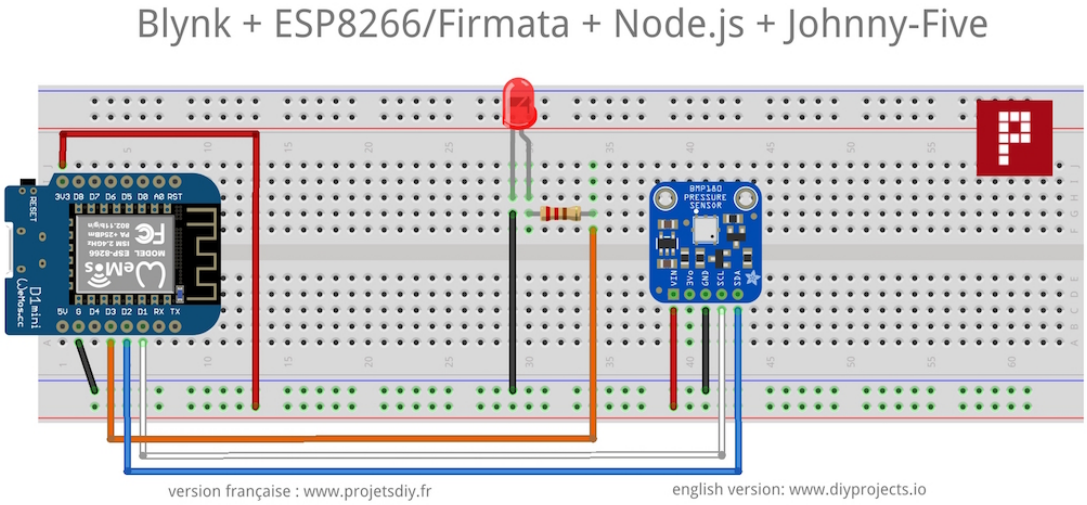
要验证一切正确，请用 9600 波特率打开串行监视器。您还可以检索 ESP8266 的 IP 地址（如果尚未设）。

示例：driving a led, reading a BMP180 in WiFi

现在去进行测试，我们将采用前面的例子，并适应新配置。

## 步骤 3：电路连接

接着，我们就可以进行电路连接。将 LED 连接到 D3 上的（GPIO-0）。在 Wemos D1 mini 上，I2C 总线（SDA - D2，SCL - D1），如下图所示：




## 步骤 4：JavaScript 代码

要建立与 ESP8266 的通信，我们将安装 etherport-client 软件包：

```
sudo npm install firmata etherport-client
```

安装完成后，创建一个新的文件：

```
cd ../..
cd nodebot
nano j5ESPDemo.js
```

在脚本开始时，我们将初始化，并允许我们使用 ESP8266 在 WiFi 中建立串行连接的对象：

```
var EtherPortClient = require("etherport-client").EtherPortClient;
```

要连接到 ESP8266，只需创建一个 EtherPortClient 对象。该对象需要两个参数，即 ESP 和端口的 IP 地址。

```
var board = new five.Board({
  port: new EtherPortClient({
    host: "xxx.xxx.xxx.xxx",  // IP ESP8266
    port: 3030
  }),
  timeout: 10000,
  repl: false
});
```

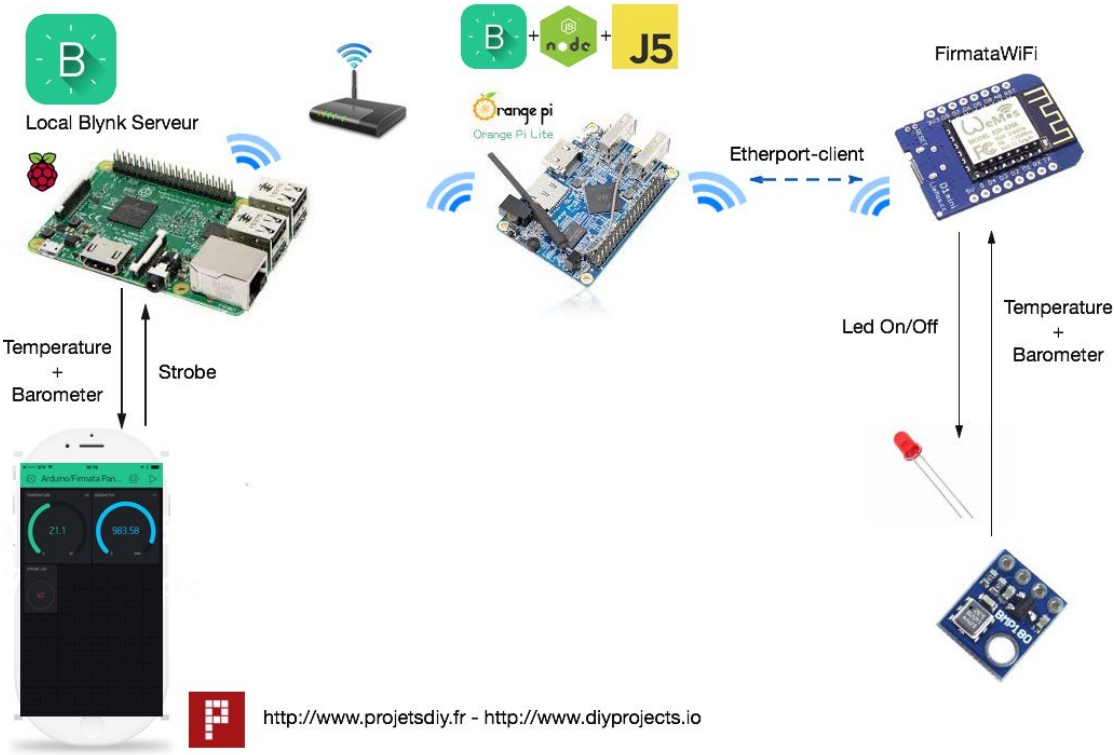
其余的编程，与先前的教程完全相同。

为什么不使用 **npm Firmata** 软件包?

在互联网上，您会发现许多教程通过 Firmata 软件包来建立连接。该软件包允许您在 JavaScript 中控制 Arduino 的 GPIO。这也是 Johnny-Five 框架的目标。在其官方指南（http://johnny-five.io/api/board/）上，详细介绍了管理地图的完整 API。在我看来，没有必要在你的项目中加上这个 “over-layer”。在 I2C 与 BMP180 通信中并不会遇到任何困难，并且可以从 Blynk 应用程序驱动 GPIO。

## 步骤 5：ESP8266 + Johnny-Five + Blynk

现在，让我们进一步将 ESP8266 整合到 Blynk 中。感谢有了 Johnny-Five，我们将不需要在 ESP8266 上进行任何编程。以下是系统的架构图——这一切都发生在 WiFi 网络中：




**Blynk** 部分

打开 Blynk，创建一个新的项目，并添加：

- 2 个测量表（V0 为温度，V1 为气压计）
- 1 个切换式开关（V2）

### JavaScript 代码

将下面的代码粘贴到新脚本中，并修改如下的参数：

- Token：通过邮件收到的 Blynk 项目的 Token
- 本地 Blynk 服务器的 IP（请搜索：如何在 Raspberry Pi 上安装本地 Blynk 服务）
- ESP8266 的 IP
- 端口

代码如下：

```
1 var Blynk = require('blynk-library');
2 var five = require("johnny-five");
3 var EventEmitter = require('events').EventEmitter;
4 var EtherPortClient = require("etherport-client").EtherPortClient;
5
6 var AUTH = 'b065eb0a6e36434da42367b3fa7c3340';  // Remplacer par votre
      Token Blynk - Replace by your Blynk Token
7
8 var event = new EventEmitter();                 // Evenements javascript -
      Javascript Events
9 var DEBUG = true; //false;                      // Active les messages de
      mise au point - Activate debug message
10
11 // Setup Blynk
12 var blynk = new Blynk.Blynk(AUTH, options = {
13   // Connecteur au serveur Blynk local - Local Blynk server connector
14   connector : new Blynk.TcpClient( options = { addr:"IP_SERVEUR_BLYNK",
         port:8442 } )
15 });
16
17 var V0 = new blynk.VirtualPin(0);  // Temperature
18 var V1 = new blynk.VirtualPin(1);  // Barometer
19 var V2 = new blynk.VirtualPin(2);  // On/Off led
```

```
20 var temp, pa;
21 blynk.on('connect', function() { console.log("Blynk ready.");
       });blynk.on('disconnect', function() { console.log("DISCONNECT"); });
22
23 // update host to the IP address for your ESP board
24 var board = new five.Board({
25   port: new EtherPortClient({
26     host: "192.168.1.73",
27     port: 3030
28   }),
29   timeout: 10000,
30   repl: true,
31   debug: true
32 });
33
34 board.on("ready", function() {
35   console.log("READY!");
36   var led = new five.Led(0);
37
38   var bmp180 = new five.Multi({
39     controller: "BMP180",
40     freq: 5000
41   });
42
43   bmp180.on("change", function() {
44     console.log("Temperature BMP180   ", this.thermometer.celsius, "°C");
45     console.log("Atm. Pressure BMP180 ", this.barometer.pressure * 10 ,
           "hPa");
46     temp = this.thermometer.celsius;
47     pa = this.barometer.pressure * 10;
48   });
49
50   board.on("exit", function(){
51     led.stop();
52     led.off();
53   });
```

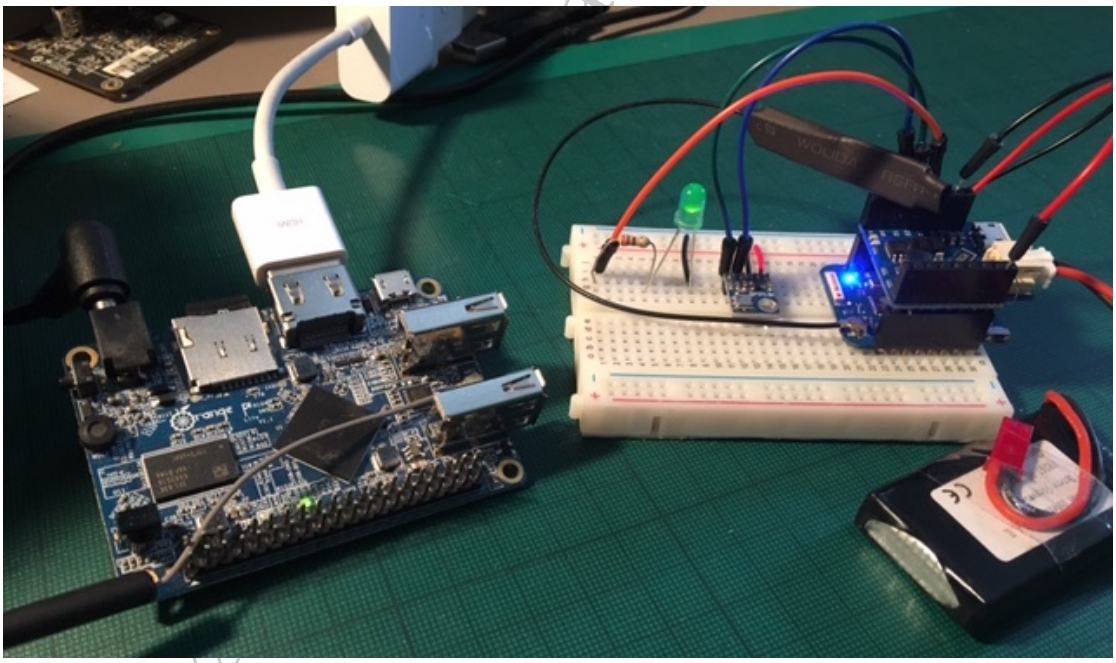
```
54   event.on('V2', function(param){
55 if ( param == 1 || param == true ) {      if ( DEBUG ) { console.log("Start
         strobe led"); }
56       led.strobe(500);
57     } else {
58       if ( DEBUG ) { console.log("Stop strobe Led"); }
59       led.stop();
60       led.off();
61     }
62   });
63
64 });
65
66 setInterval(function() {
67   if ( temp != undefined ) {
68     if ( DEBUG ) { console.log('Temperature:', temp + ' C'); }
69     V0.write(temp);
70   }
71   if ( pa != undefined ) {
72     if ( DEBUG ) { console.log('Humidity:   ', pa    + ' hPa'); }
73     V1.write(pa);
74   }
75 }, 5000);
76
77 V2.on('write', function(param){
78   if ( DEBUG ) { console.log("V2 ", param); }
79   event.emit('V2',param);
80 });
```

保存文件。接着，运行脚本 `node j5ESPDemo.js`。

```
1 node j5ESPBlink.js
2 Connecting to TCP: 192.168.1.24 8442
3 1491226899573 SerialPort Connecting to host:port: 192.168.1.73:3030
4 1491226899605 Connected Connecting to host:port: 192.168.1.73:3030
5 Connected
6 Authorized
```

```
Blynk ready.
1491226905821 Repl Initialized>> READY!
Temperature BMP180    20.9 °C
Atm. Pressure BMP180  983.7 hPa
V0  [ '1' ]
Start strobe led
Temperature BMP180    20.9 °C
Atm. Pressure BMP180  983.67 hPa
V0  [ '0' ]
Stop strobe Led
Temperature BMP180    20.9 °C
Atm. Pressure BMP180  983.63 hPa
```

正如你在下图所见，Wemos 使用电池进行工作（1100 mAh）。这不是一个理想的情况，因为您无法从 ESP8266 待机中获益。但是，这是设置开发远程控制的一个很好的架构示例。



在这里，通过安装 StandardFirmataWiFi 固件，我们在电脑（或迷你 PC Raspberry Pi ...）和 ESP8266 之间通信。在这几行代码中，通过 WiFi 中的 ESP8266 来驱动或恢复测量非常简单。

原文链接：https://diyprojects.io/connecting-esp8266-blynk-johnny-five-firmata-wifi/

原文链接：https://www.wandianshenme.com/play/raspberrypi-blynk-server-control-esp8266-johnn