

ESP8266 กับการควบคุม RGB LED แบบเรียลไทม์

ESP8266 กับการควบคุม RGB LED แบบเรียลไทม์

<http://www.ioxhop.com/article/44/esp8266->

<http://www.ioxhop.com/article/44/esp8266-%E0%B8%81%E0%B8%B1%E0%B8%9A%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%84%E0%B8%A7%E0%B8%9A%E0%B8%84%E0%B8%B8%E0%B8%A1-rgb-led-%E0%B9%81%E0%B8%9A%E0%B8%9A%E0%B9%80%E0%B8%A3%E0%B8%B5%E0%B8%A2%E0%B8%A5%E0%B9%84%E0%B8%97%E0%B8%A1%E0%B9%8C>

HTML Content



ESP8266 ถือเป็นโมดูล WiFi ยอดนิยมมากที่สุดในขนาดนี้ โดยมีความสามารถเป็นไมโครคอนโทรลเลอร์ที่มาพร้อมกับ WiFi ใช้ได้ทั้งในโหมด AP และ STA ในราคาโมดูลเปล่าเพียงร้อยละบาทเท่านั้น ถึงแม้ ESP32 จะออกมาใหม่แล้วในตอนนี้ แต่ ESP8266 ยังคงเหมาะสมที่นำมาใช้กับงานเล็ก ๆ อยู่เหมือนเดิม

ในบทความนี้จะเป็นการนำ ESP8266 มาใช้งานในโหมด AP และทดลองใช้การสื่อสารแบบเรียลไทม์ผ่านโปรโตคอล TCP รวมถึงจะเป็นการทดลองใช้บอร์ด Witty Cloud สำหรับการพัฒนาอีกด้วย

รู้จักกับ RGB LED

ก่อนที่จะมาทดลองใช้งานกับ ESP8266 เราควรที่จะมาทำความรู้จักกับหลอด LED RGB กับก่อนครับ โดยหลอด RGB LED จะแบ่งได้ออกเป็น 2 ประเภท คือ

1. หลอด LED RGB ธรรมดา

สามารถผ่านรายละเอียดได้ในบทความเก่า [Arduino กับการใช้งาน RGB LED](#)

2. หลอด LED RGB แบบไอซี

เนื่องจากหลอด LED RGB แบบธรรมดานั้นต้องการควบคุมถึง 3 ขา ต่อ 1 ดวง ทำให้การนำมาต่อกันแล้วควบคุมให้แต่ละดวงมีสีต่างกันทำได้ยาก ทำให้ RGB LED แบบไอซีนิยมใช้งานกันมากกว่า แต่ข้อเสียคือต้องมี Controller ในการควบคุม ซึ่งตัว Arduino เอง หรือ ESP8266 ก็ทำตัวเป็น Controller ได้

RGB LED แบบไอซีเบอร์ที่นิยมใช้งานกันตอนนี้คือเบอร์ WS2812 โดยมีข้อดีคือใช้สายสัญญาณควบคุมเพียง 1 เส้น และสามารถต่อไปได้อีก โดยมีขั้ว DIO สำหรับจ่ายสัญญาณเข้า และมี DOUT สำหรับจ่ายสัญญาณออกไปให้ตัวถัดไป

WS2812 นิยมใช้งานมาก โดยมีการนำไปทำเป็นเส้นยาว ๆ นำไปติดบน PCB แบบเป็นแถวเรียง และแบบวงกลมอีกด้วย



LED RGB WS2812 แบบเส้น (ขอบคุณที่มา www.tweaktown.com)



LED RGB WS2812 แบบแถวเรียง ([NeoPixel Stick](#))



LED RGB WS2812 แบบวงกลม (ขอบคุณที่มา www.adafruit.com)

การใช้งานโปรโตคอล TCP บน ESP8266

หากกลับไปดูในบทความเก่า [การใช้งาน ESP8266 ในโหมด AP และการรับส่งข้อมูลผ่าน TCP](#) ก็ได้ใช้โปรโตคอล TCP เช่นเดียว แต่ในบทความเก่านี้ได้ใช้วิธีเปิด Socket พอรับข้อมูลมาแล้วก็ปิด Socket ไปเลย เพื่อป้องกันอาการค้าง ซึ่ง ESP8266 ที่คุณภาพราคาจะมีปัญหานี้ในบางครั้ง (ให้สมมติว่านำไปติดกับโดรนแล้ว ESP8266 ค้างในขณะที่โดรนกำลังบินอยู่จะเกิดอะไรขึ้น)

และเนื่องจากในบทความนี้ได้นำมาควบคุม LED RGB ซึ่งไม่มีส่วนใดที่หนีเราไปได้ กรณีที่ค้าง ก็สามารถกดปุ่ม Reset ได้เลย ระบบก็กลับมาเป็นปกติ ดังนั้นในบทความนี้จะเลือกใช้การเชื่อมต่อแบบค้างไว้เลย กล่าวคือมีการเปิด Socket ค้างไว้ตลอดเวลา ทางฝั่ง ESP8266 สามารถตอบสนองต่อคำสั่งได้ทันทีโดยไม่ต้องรอการเชื่อมต่อใหม่

ในบทความเก่าหากลบบรรทัดที่ 26 ออก ก็จะสามารถใช้งานแบบเชื่อมต่อค้างไว้ได้เลย

ทดลองใช้ Witty Cloud

บอร์ด Witty Cloud ที่ทาง IOXhop พึ่งนำมาจัดจำหน่าย มีหลอด LED RGB แบบธรรมดาติดอยู่ในบอร์ดด้วย ทำให้ง่ายต่อการทดลองในครั้งนี้

ตัวหลอด LED RGB ที่ต่ออยู่บนบอร์ด Witty Cloud เป็นแบบคอมมอนแคโทด และมีขาการต่อควบคุมแต่ละสีดังนี้

- RED :: GPIO15
- GREEN :: GPIO12
- BLUE :: GPIO13

การควบคุมให้แต่ละสีมีความสว่างแตกต่างกันสามารถทำได้โดยใช้ฟังก์ชัน analogWrite() ซึ่งเป็นฟังก์ชันสำหรับสร้างสัญญาณแบบ PWM โค้ดที่ได้จากในบทความเก่า เมื่อนำมารวมกันแล้วจึงเป็นไปตามด้านล่างนี้

```
#include <ESP8266WiFi.h>
```

```
// ----- sscanf() function -----
```

```
#include <stdio.h>
```

```
#include <stdarg.h>
```

```
#include <string.h>
```

```
#include <ctype.h>
```

```
#define MAXLN 200
```

```
#define ISSPACE " \t\n\r\f\v"
```

```
// -----
```

```
#define pinRED 15
```

```
#define pinGREEN 12
```

```
#define pinBLUE 13
```

```
WiFiServer server(88);
```

```
void setup() {
```

```
  pinMode(pinRED, OUTPUT);
```

```
  pinMode(pinGREEN, OUTPUT);
```

```
  pinMode(pinBLUE, OUTPUT);
```

```
  Serial.begin(115200);
```

```
  WiFi.mode(WIFI_AP);
```

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

```
}
```

```
void loop() {
```

```
  WiFiClient client = server.available();
```

```
  if (client) {
```

```
    Serial.println("New client");
```

```
    while (client.connected()) {
```

```
      if (client.available()) {
```

```
        String line = client.readStringUntil('\r');
```

```
        client.readStringUntil('\n');
```

```
        Serial.println(line);
```

```
        int r = 0, g = 0, b = 0;
```

```
        if (sscanf(line.c_str(), "%d,%d,%d", &r, &g, &b) > 0) {
```

```
          analogWrite(pinRED, r);
```

```
          analogWrite(pinGREEN, g);
```

```
analogWrite(pinBLUE, b);
```

```
}
```

```
}
```

```
delay(1);
```

```
}
```

```
delay(1);
```

```
client.stop();
```

```
Serial.println("Client disconnect");
```

```
}
```

```
delay(1);
```

```
}
```

```
size_t strcspn (const char *p, const char *s) {
```

```
int i, j;
```

```
for (i = 0; p[i]; i++) {
```

```
for (j = 0; s[j]; j++) {
```

```
if (s[j] == p[i])
```

```
break;
```

```
}
```

```
if (s[j])
```

```
break;
```

```
}
```

```
return (i);
```

```
}
```

```
char * _getbase(char *p, int *basep) {
```

```
if (p[0] == '0') {
```

```
switch (p[1]) {
```



```
case 'x':
```

```
*basep = 16;
```

```
break;
```

```
case 't': case 'n':
```

```
*basep = 10;
```

```
break;
```

```
case 'o':
```

```
*basep = 8;
```

```
break;
```

```
default:
```

```
*basep = 10;
```

```
return (p);
```

```
}
```

```
return (p + 2);
```

```
}
```

```
*basep = 10;
```

```
return (p);

}

/*

*_atob(vp,p,base)

*/

int _atob (uint32_t *vp, char *p, int base) {

uint32_t value, v1, v2;

char *q, tmp[20];

int digit;


if (p[0] == '0' && (p[1] == 'x' || p[1] == 'X')) {

base = 16;

p += 2;

}

}
```

```
if (base == 16 && (q = strchr (p, '.')) != 0) {
```

```
if (q - p > sizeof(tmp) - 1)
```

```
return (0);
```

```
strncpy (tmp, p, q - p);
```

```
tmp[q - p] = '\0';
```

```
if (!_atob (&v1, tmp, 16))
```

```
return (0);
```

```
q++;
```

```
if (strchr (q, '.'))
```

```
return (0);
```

```
if (!_atob (&v2, q, 16))
```

```
return (0);
```

```
*vp = (v1 << 16) + v2;
```

```
return (1);
```

```
}
```

```
value = *vp = 0;
```

```
for (; *p; p++) {
```

```
if (*p >= '0' && *p <= '9')
```

```
digit = *p - '0';
```

```
else if (*p >= 'a' && *p <= 'f')
```

```
digit = *p - 'a' + 10;
```

```
else if (*p >= 'A' && *p <= 'F')
```

```
digit = *p - 'A' + 10;
```

```
else
```

```
return (0);
```

```
if (digit >= base)
```

```
return (0);
```

```
value *= base;
```

```
value += digit;
```

```
}
```

```
*vp = value;
```

```
return (1);
```

```
}
```

```
/*
```

```
* atob(vp,p,base)
```

```
* converts p to binary result in vp, rtn 1 on success
```

```
*/
```

```
int atob(uint32_t *vp, char *p, int base) {
```

```
uint32_t v;
```

```
if (base == 0)
```

```
p = _getbase (p, &base);
```

```
if (_atob (&v, p, base)) {
```

```
*vp = v;
```

```
return (1);
```

```
}
```

```
return (0);
```

```
}
```

```
/*
```

```
* vsscanf(buf,fmt,ap)
```

```
*/
```

```
int vsscanf (const char *buf, const char *s, va_list ap) {
```

```
uint32_t count, noassign, width, base, lflag;
```

```
const char *tc;
```

```
char *t, tmp[MAXLN];
```

```
count = noassign = width = lflag = 0;
```

```
while (*s && *buf) {

while (isspace (*s))

s++;

if (*s == '%') {

s++;

for (; *s; s++) {

if (strchr ("dibouxcsefg%", *s))

break;

if (*s == '*')

noassign = 1;

else if (*s == 'l' || *s == 'L')

lflag = 1;

else if (*s >= '1' && *s <= '9') {

for (tc = s; isdigit (*s); s++);

strncpy (tmp, tc, s - tc);

tmp[s - tc] = '\0';
```

```
atob (&width, tmp, 10);

s--;

}

}

if (*s == 's') {

while (isspace (*buf))

buf++;

if (!width)

width = strcspn (buf, ISSPACE);

if (!noassign) {

strncpy (t = va_arg (ap, char *), buf, width);

t[width] = '\0';

}

buf += width;

} else if (*s == 'c') {

if (!width)
```



```
width = 1;

if (!noassign) {

    strncpy (t = va_arg (ap, char *), buf, width);

    t[width] = '\0';

}

buf += width;

} else if (strchr ("dobxu", *s)) {

    while (isspace (*buf))

        buf++;

    if (*s == 'd' || *s == 'u')

        base = 10;

    else if (*s == 'x')

        base = 16;

    else if (*s == 'o')

        base = 8;

    else if (*s == 'b')
```

```
base = 2;

if (!width) {

if (isspace (*(s + 1)) || *(s + 1) == 0)

width = strcspn (buf, ISSPACE);

else

width = strchr (buf, *(s + 1)) - buf;

}

strncpy (tmp, buf, width);

tmp[width] = '\0';

buf += width;

if (!noassign)

atob (va_arg (ap, uint32_t *), tmp, base);

}

if (!noassign)

count++;

width = noassign = lflag = 0;
```

```
s++;
```

```
} else {
```

```
while (isspace (*buf))
```

```
buf++;
```

```
if (*s != *buf)
```

```
break;
```

```
else
```

```
s++, buf++;
```

```
}
```

```
}
```

```
return (count);
```

```
}
```

```
int sscanf (const char *buf, const char *fmt, ...) {
```

```
int count;
```

```
va_list ap;
```

```
va_start(ap, fmt);
```

```
count = vsscanf (buf, fmt, ap);
```

```
va_end(ap);
```

```
return count;
```

```
}
```

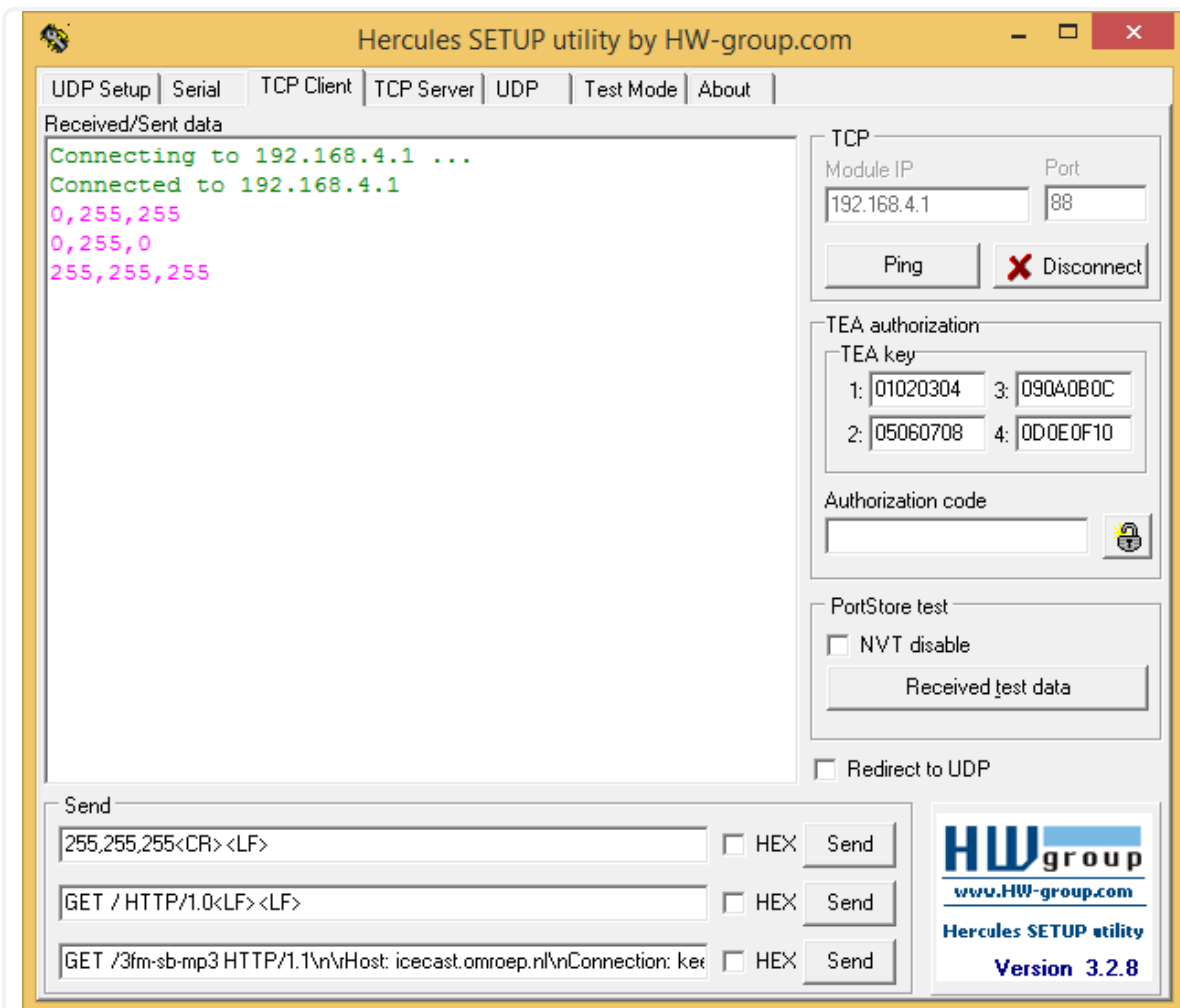
[WittyRealtimeRGB.ino](#) hosted with ♥ by [GitHub](#)

[view raw](#)

การทดสอบสามารถทำได้เชื่อมต่อ WiFi ไปที่ SSID ชื่อ WiFi_xxx ใช้โปรแกรม hercules
เชื่อมต่อไปที่ IP 192.168.4.1 และใส่พอร์ตเป็น 88 แล้วส่งข้อความในรูปแบบ

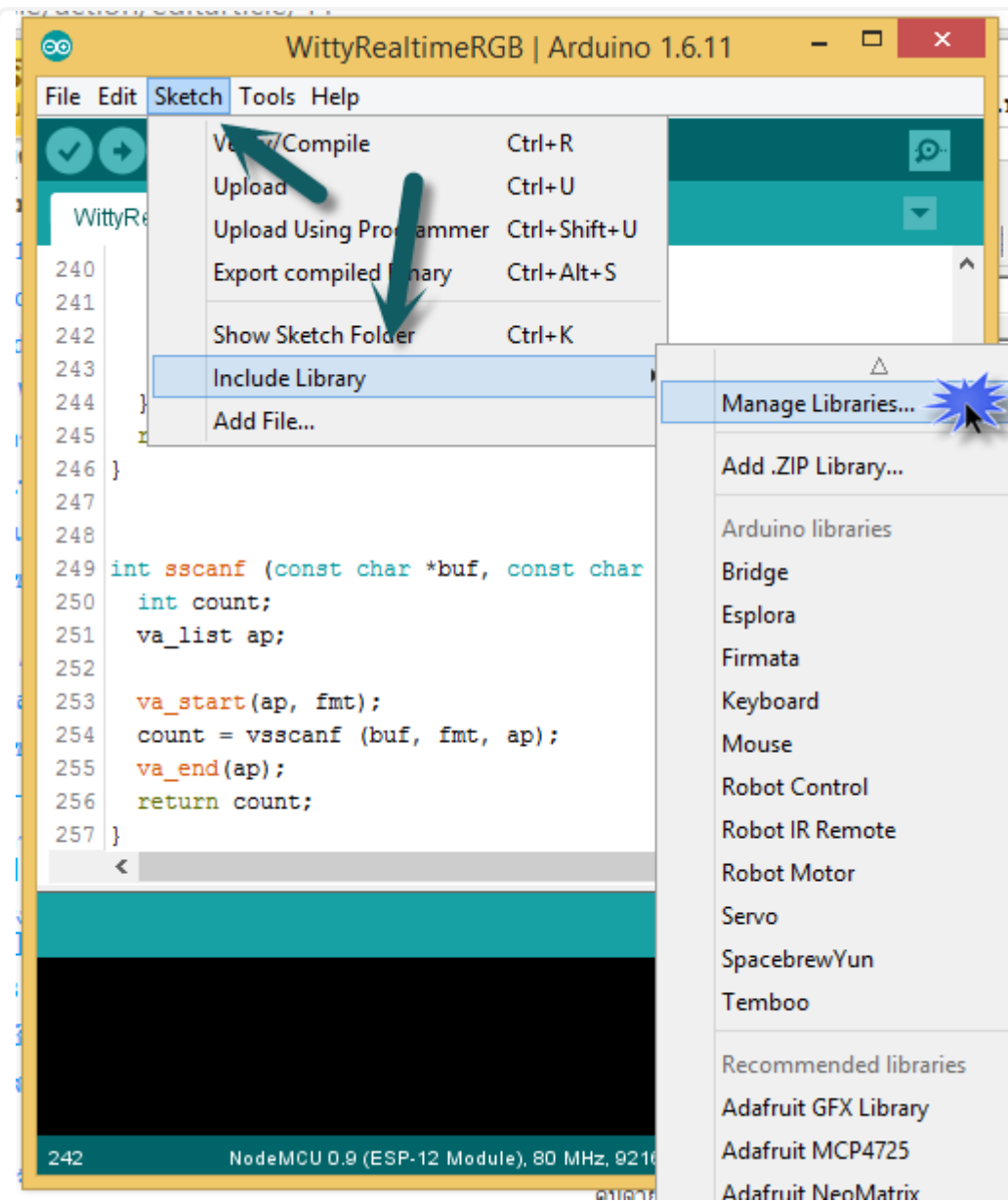
ค่าสีแดง,ค่าเขียว,ค่าสีน้ำเงิน

ผลที่ได้สีของหลอด LED จะเปลี่ยนไปตามค่าสีที่ส่งไป

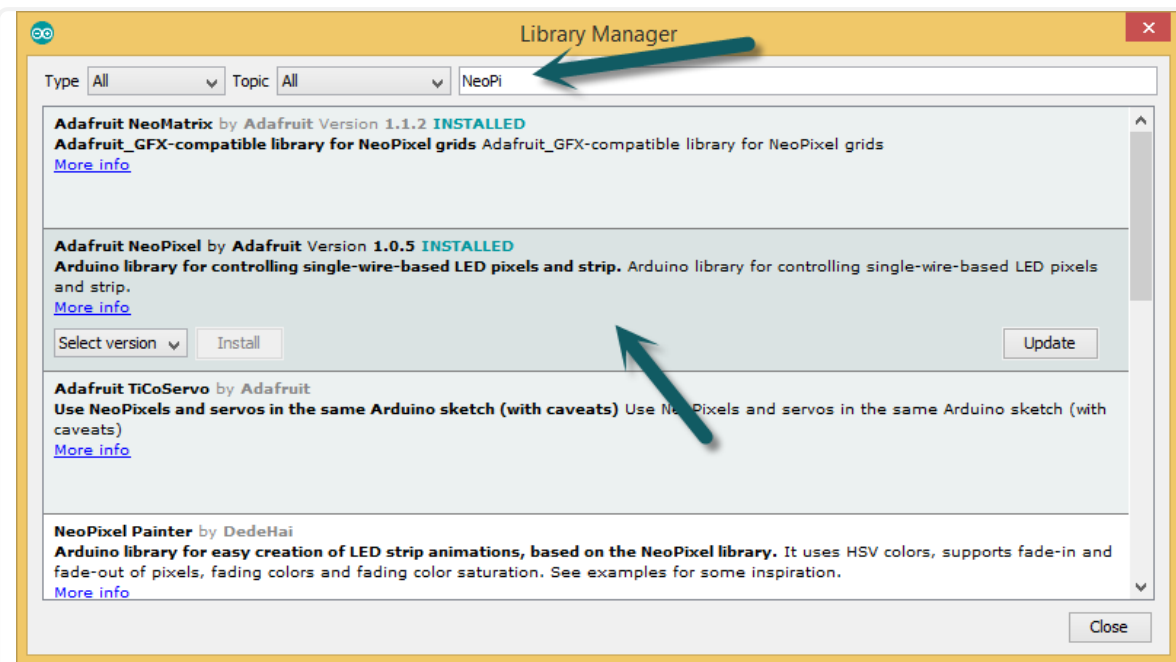


ESP8266 + NeoPixel

ส่วน NeoPixel ที่ใช้ไอซีเบอร์ WS2812 ก็สามารถนำมาใช้งานได้เช่นเดียวกัน โดยจะต้องลงไลบรารี NeoPixel ซึ่งทำได้โดยคลิกไปที่เมนู File > Include Library > Manage Libraries



ในช่องค้นหา ใส่ NeoPixel เลือก Adafruit NeoPixel แล้วกด Install ได้เลย (ในรูปได้ติดตั้งไว้แล้ว)



ก๊อบโค้ดด้านล่างไปอัปโหลด

```
#include <ESP8266WiFi.h>
```

```
// ----- sscanf() function -----
```

```
#include <stdio.h>
```

```
#include <stdarg.h>
```

```
#include <string.h>
```

```
#include <ctype.h>
```

```
#define MAXLN 200
```

```
// -----
```

```
#include <Adafruit_NeoPixel.h>
```

```
#define PIN D1
```

```
#define NUMPIXELS 8
```

```
WiFiServer server(88);
```

```
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB +  
NEO_KHZ800);
```

```
void setup() {
```

```
  Serial.begin(115200);
```

```
  WiFi.mode(WIFI_AP);
```

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



```
pixels.begin();
```

```
show(0, 0, 0);
```

```
}
```

```
void loop() {
```

```
WiFiClient client = server.available();
```

```
if (client) {
```

```
Serial.println("New client");
```

```
while (client.connected()) {
```

```
if (client.available()) {
```

```
String line = client.readStringUntil('\r');
```

```
client.readStringUntil('\n');
```

```
Serial.println(line);
```

```
int r = 0, g = 0, b = 0;
```

```
if (sscanf(line.c_str(), "%d,%d,%d", &r, &g, &b) > 0)
```

```
show(r, g, b);
```

```
}
```

```
delay(1);
```

```
}
```

```
delay(10);
```

```
client.stop();
```

```
Serial.println("Client disconnect");
```

```
}
```

```
delay(10);
```

```
}
```

```
void show(int r, int g, int b) {
```

```
for (int i=0;i<NUMPIXELS;i++)
```

```
pixels.setPixelColor(i, pixels.Color(r, g, b));
```

```
pixels.show();
```

```
}
```

```
size_t strcspn (const char *p, const char *s) {
```

```
    int i, j;
```

```
    for (i = 0; p[i]; i++) {
```

```
        for (j = 0; s[j]; j++) {
```

```
            if (s[j] == p[i])
```

```
                break;
```

```
        }
```

```
        if (s[j])
```

```
            break;
```

```
    }
```

```
    return (i);
```

```
}
```

```
char *_getbase(char *p, int *basep) {  
  
    if (p[0] == '0') {  
  
        switch (p[1]) {  
  
            case 'x':  
  
                *basep = 16;  
  
                break;  
  
            case 't': case 'n':  
  
                *basep = 10;  
  
                break;  
  
            case 'o':  
  
                *basep = 8;  
  
                break;  
  
            default:  
  
                *basep = 10;  
  
                return (p);  
        }  
    }  
}
```

```
}
```

```
return (p + 2);
```

```
}
```

```
*basep = 10;
```

```
return (p);
```

```
}
```

```
/*
```

```
* _atob(vp,p,base)
```

```
*/
```

```
int _atob (uint32_t *vp, char *p, int base) {
```

```
uint32_t value, v1, v2;
```

```
char *q, tmp[20];
```

```
int digit;
```

```
if (p[0] == '0' && (p[1] == 'x' || p[1] == 'X')) {
```

```
base = 16;
```

```
p += 2;
```

```
}
```

```
if (base == 16 && (q = strchr (p, '.')) != 0) {
```

```
if (q - p > sizeof(tmp) - 1)
```

```
return (0);
```

```
strncpy (tmp, p, q - p);
```

```
tmp[q - p] = '\0';
```

```
if (!_atob (&v1, tmp, 16))
```

```
return (0);
```

```
q++;
```

```
if (strchr (q, '.'))
```

```
return (0);
```

```
if (!_atob (&v2, q, 16))
```

```
return (0);
```

```
*vp = (v1 << 16) + v2;
```

```
return (1);
```

```
}
```

```
value = *vp = 0;
```

```
for (; *p; p++) {
```

```
if (*p >= '0' && *p <= '9')
```

```
digit = *p - '0';
```

```
else if (*p >= 'a' && *p <= 'f')
```

```
digit = *p - 'a' + 10;
```

```
else if (*p >= 'A' && *p <= 'F')
```

```
digit = *p - 'A' + 10;
```

```
else
```

```

return (0);

if (digit >= base)

return (0);

value *= base;

value += digit;

}

*vp = value;

return (1);

}

/*

* atob(vp,p,base)

* converts p to binary result in vp, rtn 1 on success

*/

int atob(uint32_t *vp, char *p, int base) {

```



```

uint32_t v;

if (base == 0)

p = _getbase (p, &base);

if (_atob (&v, p, base)) {

*vp = v;

return (1);

}

return (0);

}

/*

* vsscanf(buf,fmt,ap)

*/

int vsscanf (const char *buf, const char *s, va_list ap) {

uint32_t count, noassign, width, base, lflag;

```

```
const char *tc;
```

```
char *t, tmp[MAXLN];
```

```
count = noassign = width = lflag = 0;
```

```
while (*s && *buf) {
```

```
while (isspace (*s))
```

```
s++;
```

```
if (*s == '%') {
```

```
s++;
```

```
for (; *s; s++) {
```

```
if (strchr ("dibouxcsefg%", *s))
```

```
break;
```

```
if (*s == '*')
```

```
noassign = 1;
```

```
else if (*s == 'l' || *s == 'L')
```

```
lflag = 1;
```

```
else if (*s >= '1' && *s <= '9') {

    for (tc = s; isdigit (*s); s++);

    strncpy (tmp, tc, s - tc);

    tmp[s - tc] = '\0';

    atob (&width, tmp, 10);

    s--;

}

}

if (*s == 's') {

    while (isspace (*buf))

        buf++;

    if (!width)

        width = strcspn (buf, ISSPACE);

    if (!noassign) {

        strncpy (t = va_arg (ap, char *), buf, width);

        t[width] = '\0';
```

```
}

buf += width;

} else if (*s == 'c') {

if (!width)

width = 1;

if (!noassign) {

strncpy (t = va_arg (ap, char *), buf, width);

t[width] = '\0';

}

buf += width;

} else if (strchr ("dobxu", *s)) {

while (isspace (*buf))

buf++;

if (*s == 'd' || *s == 'u')

base = 10;

else if (*s == 'x')
```

```
base = 16;

else if (*s == 'o')

base = 8;

else if (*s == 'b')

base = 2;

if (!width) {

if (isspace (*(s + 1)) || *(s + 1) == 0)

width = strcspn (buf, ISSPACE);

else

width = strchr (buf, *(s + 1)) - buf;

}

strncpy (tmp, buf, width);

tmp[width] = '\0';

buf += width;

if (!noassign)

atob (va_arg (ap, uint32_t *), tmp, base);
```

```
}

if (!noassign)

count++;

width = noassign = lflag = 0;

s++;

} else {

while (isspace (*buf))

buf++;

if (*s != *buf)

break;

else

s++, buf++;

}

}

return (count);

}
```

```
int sscanf (const char *buf, const char *fmt, ...) {  
  
int count;  
  
va_list ap;  
  
va_start(ap, fmt);  
  
count = vsscanf (buf, fmt, ap);  
  
va_end(ap);  
  
return count;  
}
```

 HTML Content

}

[NeoPixelRealtimeRGB.ino](#) hosted with ♥ by [GitHub](#)

[view raw](#)

ใช้โปรแกรม hercules ตามวิธีด้านบนในการทดสอบ

แนะนำแอปพลิเคชัน ESPColor



```
}
```

[NeoPixelRealtimeRGB.ino](https://github.com/NeoPixelRealtimeRGB/ino) hosted with ❤ by [GitHub](https://github.com)

[view raw](#)

ใช้โปรแกรม hercules ตามวิธีด้านบนในการทดสอบ

แนะนำแอปพลิเคชัน ESPColor



ร้าน IOXhop ได้จัดทำแอปพลิเคชันสำหรับใช้งานร่วมกับโค้ดในบทความนี้ โดยสามารถเชื่อมต่อ จากนั้นควบคุมสีได้เลย ตัวแอปพลิเคชันถูกอัปโหลดไว้บน Google Play แล้ว สามารถดาวน์โหลดได้โดยพิมพ์ในช่องค้นหาว่า [EPSColor](#)

