2022-07-24

Python 快速搭建本地服务器

进入终端后，选定合适的文件夹，执行以下命令，即可快速搭建本地服务器

```
python3 -m http.server
```

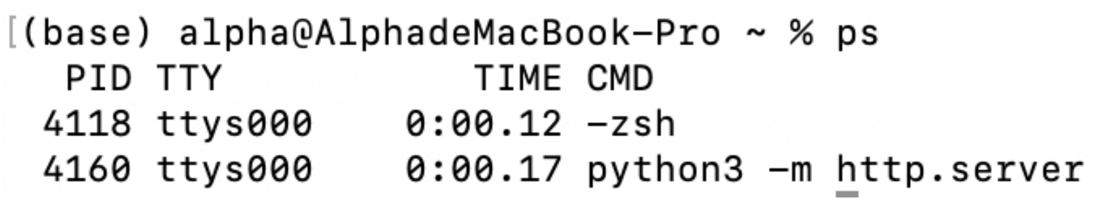
可以指定端口

```
python3 -m http.server 8001
```

加 & 可以后台运行（ctrl+c 无法关闭服务）

```
python3 -m http.server &
```

若要关闭服务，可以使用 ps 命令查看 PID



```
[(base) alpha@AlphadeMacBook-Pro ~ % ps
  PID TTY           TIME CMD
 4118 ttys000    0:00.12 -zsh
 4160 ttys000    0:00.17 python3 -m http.server
```

再使用 kill PID 以结束进程，如

```
kill 4160
```

ESP32CAM

摄像头需要使用 Arduino IDE 烧录程序

```
/*
```

```
 分辨率默认配置：config.frame_size = FRAMESIZE_UXGA;
 其他配置：
 FRAMESIZE_UXGA （1600 x 1200）
 FRAMESIZE_QVGA （320 x 240）
 FRAMESIZE_CIF （352 x 288）
 FRAMESIZE_VGA （640 x 480）
 FRAMESIZE_SVGA （800 x 600）
 FRAMESIZE_XGA （1024 x 768）
 FRAMESIZE_SXGA （1280 x 1024）
 config.jpeg_quality = 10;（10-63）越小照片质量最好
 数字越小表示质量越高，但是，如果图像质量的数字过低，尤其是在高分辨率时，可能会导致
ESP32-CAM崩溃
*/
#include "esp_http_client.h"
#include "esp_camera.h"
#include <WiFi.h>
#include <ArduinoJson.h>

/*********************需要修改的地方**********************/
const char* ssid = "114514";           //WIFI名称
const char* password = "1919810";   //WIFI密码
int capture_interval = 5 * 1000;     //默认5秒上传一次，可更改（本项目是自动上传，如需条件触发上
传，在需要上传的时候，调用take_send_photo()即可）
const char*  post_url = "http://192.168.0.104/test"; //上传地址
/*******************************************************/

static String httpResponseString;//接收服务器返回信息
long current_millis;
long last_capture_millis = 0;

/*定义引脚*********************************/
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
```

```
#define Y7_GPIO_NUM      39
#define Y6_GPIO_NUM      36
#define Y5_GPIO_NUM      21
#define Y4_GPIO_NUM      19
#define Y3_GPIO_NUM      18
#define Y2_GPIO_NUM       5
#define VSYNC_GPIO_NUM   25
#define HREF_GPIO_NUM    23
#define PCLK_GPIO_NUM    22
/*****************************************/
void setup()
{
  Serial.begin(115200);
  if (init_wifi()) { // Connected to WiFi
    Serial.println("Internet connected");
  }
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
  if (psramFound()) { //判断缓存容量是否充足
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
}
/********初始化WIFI*********/
bool init_wifi(){
  int connAttempts = 0;
  Serial.println("\r\nConnecting to: " + String(ssid));
  WiFi.begin(ssid, password);
  while (WiFi.status()!=WL_CONNECTED){
    delay(500);
    Serial.print(".");
    if (connAttempts > 10) return false;
    connAttempts++;}
  return true;
}
/********http请求处理函数*********/
esp_err_t _http_event_handler(esp_http_client_event_t *evt){
  if (evt->event_id == HTTP_EVENT_ON_DATA){
    httpResponseString.concat((char *)evt->data);}
  return ESP_OK;
}
/********推送图片*********/
static esp_err_t take_send_photo()
{
  Serial.println("Taking picture...");
  camera_fb_t * fb = NULL;
  esp_err_t res = ESP_OK;

  fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Camera capture failed");
    return ESP_FAIL;
  }
  httpResponseString = "";
  esp_http_client_handle_t http_client;
  esp_http_client_config_t config_client = {0};

  config_client.url = post_url;
  config_client.event_handler = _http_event_handler;
  config_client.method = HTTP_METHOD_POST;

  http_client = esp_http_client_init(&config_client);
  esp_http_client_set_post_field(http_client, (const char *)fb->buf, fb->len);//设置http发送的内容和长
度
  esp_http_client_set_header(http_client, "Content-Type", "image/jpg"); //设置http头部字段
  esp_err_t err = esp_http_client_perform(http_client);//发送http请求
  if (err == ESP_OK) {
    StaticJsonDocument<200> doc;
    DeserializationError error = deserializeJson(doc, httpResponseString);
    if (error) {
      Serial.print(F("deserializeJson() failed: "));
      Serial.println(error.c_str());
    }
    String url = doc["url"];
    Serial.println(url);
  }
  esp_http_client_cleanup(http_client);
  esp_camera_fb_return(fb);
}

void loop()
{
  current_millis = millis();
  if (current_millis - last_capture_millis > capture_interval) {
    last_capture_millis = millis();
    take_send_photo();
  }
}
```

本地服务器接受程序

```
from flask import request, Flask, jsonify
import time
app = Flask(__name__)
app.debug = True
@app.route('/test', methods=['POST'])
def add_stu():
    with open(""+str(time.localtime().tm_min)+"."+str(time.localtime().tm_sec)+".jpg","wb") as f:
        f.write(request.data)
    return jsonify({"url":"ok"})
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=80)
```