**ESP8266常用库总结**

*——该库总结只针对Arduino平台*

**总结人：陈浩**

**总结参考：**[**www.taichi-maker.com**](http://www.taichi-maker.com)

**总结时间：2022-1-20——2022-1-29**

一、基本功能（无需引入库）

**1、引脚控制**

* pinMode //设置引脚工作模式

pinMode(pin,mode)

pin：为引脚名称，板上蓝色LED引脚为LED\_BUILTIN，其他引脚可直接用板上的引脚标注

mode：为工作模式，OUTPUT：输出模式；INPUT：输入模式；INPUT\_PULLUP：输入模式且打开上拉电阻

* digitaWrite //输出

digitalWrite(pin,level)

pin：为引脚名称，板上蓝色LED引脚为LED\_BUILTIN，其他引脚可直接用板上的引脚标注

level：输出电平，HIGH：为高电平，LOW：为低电平

* digitalRead //读取引脚电平状态

digitalRead(pin)

pin：为引脚名称，板上蓝色LED引脚为LED\_BUILTIN，其他引脚可直接用板上的引脚标注

返回值为HIGH 或 LOW

**2、Serial库——串口（stream数据类型）**

* begin //初始化波特率并配置波特率

Serial.begin(9600)

初始化波特率并配置波特率为9600

* print 与 println //串口输出，换行串口输出

Serial.print(val) Serial.println(val)

串口输出val，val可以是字符也可以是函数返回的参数

* readString //从ESP8266接收到数据中读取数据信息，读取到的信息将以字符串格式

返回

Serial.readString()

**3、其他**

* endsWith //判断字符串是否以某段字符串结尾

string\_1.endsWith(string\_0)

若字符串string以string\_0结尾，则返回真，否则返回假

例：String string\_1 = “chenhao”

value = string\_1.endsWith(“hao”) //返回值value为真

二、网络功能

**1、ESP8266WiFiAP库——接入点模式库（包含与ESP8266WiFi中）**

* softAP //设置接入点模式WiFi名称与密码

WiFi.softAP(wifi\_name,wifi\_password)

wifi\_name：创建的wifi网络名称，格式为字符串

wifi\_password：wifi密码，若wifi密码为空字符串则连接wifi无需密码，格式为字符串

* softAPIP //返回esp8266所建立WiFi局域网的默认网关IP地址

WiFi.softAPIP()

返回esp8266所建立WiFi局域网的默认网关IP地址

**2、ESP8266WiFiSTA库——无线终端模式库（包含与ESP8266WiFi中）**

* begin //连接wifi

WiFi.begin(wifi\_name,wifi\_name)

wifi\_name：被连接wifi名称，格式为字符串指针或字符串，例：char \*wifi\_name = “TP-LINK\_1CD4” 或 String wifi\_name = “TP-LINK\_1CD4”

wifi\_password：被连接wifi密码，格式为字符串指针或字符串

* status //获取wifi连接状态

WiFi.status()

当返回参数为WL\_CONNECTED时，说明wifi连接成功

* localIP //获取返回esp8266IP地址

WiFi.localIP()

返回无线终端分配的IP地址，即esp8266的IP地址

**3、ESP8266WiFiGeneric库——通用库（包含与ESP8266WiFi中）**

* mode //设置esp8266工作模式

WiFi.mode(val)

val = WIFI\_OFF //关闭wifi

val = WIFI\_STA //无线终端模式

val = WIFI\_AP //接入点模式

val = WIFI\_AP\_STA //接入点-无线终端双模式，即混合模式

**4、WiFiClient库——TCP客户端库（stream数据类型）**

* WiFiClient http\_client //定义WiFiClient对象，用于调用库中函数
* connect // ESP8266开发板通过TCP协议连接网络服务器

http\_client.connect(web,port)

web：为被连接的网站，定义参数web的时候可使用String、char。如：

const char \* ip = "47.92.129.18"；String ip = [www.ranye-iot.com](http://www.ranye-iot.com)

port：所要连接服务器端口号，默认为80

若连接成功返回1，否则返回0

* stop //停止ESP8266连接TCP服务器，即断开连接该网站

http\_client.stop()

* connected //检查设备是否成功连接服务器，若成功连接返回1，否则返回0

http\_client.connected()

返回值为bool型，若连接则返回1，否则返回0

* print 和 println //发送数据到已连接的服务器

http\_client.print(val) 或 http\_client.println(val)

val：为所要发送的数据可以是字符串字符或数值

* readString //从ESP8266接收到数据中读取数据，读取到的信息将以字符串格式返回

http\_client.readString()

返回值为读取到的字符串

* readStringUntil //从ESP8266接收到数据中读取数据信息，读取到的信息将以字符串

格式返回，当读到指定终止字符后会停止函数执行

http\_client.readStringUntil(terminator)

terminator：为终止字符

* available //用于检查设备是否接收到数据

http\_client.available()

该函数返回值为等待读取的数据字节数（int型）

* find //寻找指定字符或其他基本类型数据

http\_client.find(“\r\n”)

寻找转意字符“\r\n\r\n”，例：被找寻数据若为“ch\r\nchenhao“，算法采取的是边找边删除的效果：

ch\r\nchenhao

h\r\nchenhao

\r\nchenhao

r\nchenhao

\nchenhao

nchenhao

chenhao——至此为止找到，就像水流一样，因此此种数据类型为stream

**5、ESP8266WebServer库——HTTP服务器库**

* ESP8266WebServer //建立ESP8266WebServer对象，以便在程序中利用对象实现物

//联网网络服务器

ESP8266WebServer esp8266\_server(port)

port用于设置端口号，网络服务器标准http端口号为80，如不设置默认也为80

* begin //启动esp8266建立的网络服务器

esp8266\_server.begin()

esp8266\_server为ESP8266WebServer建立的对象

* close //停止esp8266建立的网络服务器

esp8266\_server.close()

esp8266\_server为ESP8266WebServer建立的对象

* on //当ESP8266开发板建立网络服务器后，每当有客户端向服务器发送该uri（网

页）HTTP请求时，可利用on函数来设置HTTP请求回调函数。通过HTTP请求回调函数，我们可以让ESP8266服务器生成响应信息并发送给HTTP请求客户端。

esp8266\_server.on(uri,uri\_handle)

esp8266\_server.on(uri,method,uri\_handle)

uri：HTTP请求客户端所请求的uri即网页，例：“/”代表请求首页

uri\_handle：HTTP请求该网页，需要回调uri\_handle函数处理

method：此参数用于设置向客户端发送响应信息时所使用的HTTP方法，常用方法有

HTTP\_GET、HTTP\_POST

* onNotFound //当ESP8266开发板建立网络服务器以后，每当有客户端向服务器发送

HTTP请求时，我们可以利用onNotFound函数来设置HTTP请求无效

地址的回调函数。

esp8266\_server.onNotFound(uri\_handle\_notFound)

uri\_handle\_notFound为处理无效地址请求的回调函数

* handleClient //处理客户端请求，由于该函数需要时刻被调用即时刻检查是否有客户

端请求，因此需放到loop函数中

esp8266\_server.handleClient()

* uri //获取客户端发送的HTTP请求行中的请求资源路径信息，例如请求首页则返回

字符串“/”

esp8266\_server.uri()

* send //在ESP8266开发板所建立的物联网服务器在运行过程中，如果有客户端向

ESP8266服务器发送HTTP请求，ESP8266服务器可使用本函数向客户端发送

响应信息

esp8266\_server.send(response\_code,content\_type,response\_connect)

response\_code：为响应状态码，响应成功即存在该网页则发送200，否则发送404

connect\_type：为响应内容类型，例纯文本类型为”text/plain”，html网页类型为

“text/html”

response\_connect：为响应内容

* sendHeader //本函数可用于向响应头信息中添加自定义键值对

esp8266\_server.sendHeader(header\_name,header\_value)

header\_name：自定义的响应头信息的名称，可使用字符串类型

header\_value：自定义的响应头值，可使用字符串类型

例：esp8266\_server.sendHeader(“Location”, “/”)

**6、ESP8266HTTPClient库——HTTP客户端库**

* HTTPClient http\_client //建立ESP8266HTTPClient对象，以便在程序中利用对象实网

络客户端操作

* begin //设置esp8266发送HTTP请求目标的URL（网址）

http\_client.begin(url)

url：url为http请求网址

* GET // ESP8266使用HTTP协议通过网络向服务器发送GET请求

http\_client.GET()

返回值为HTTP状态码，若返回HTTP\_CODE\_OK即请求成功过相当于状态码200

* POST // ESP8266使用HTTP协议通过网络向服务器发送POST请求

http\_client.POST(payload)

payload：通过POST请求所发送的数据信息

* sendRequest //ESP8266使用HTTP协议通过网络向服务器发送请求，本函数可以自

定义请求方法类型，如 GET、POST

http\_client.sendClient(type,payload)

type：发送请求的类型，如“GET”、”POST”

payload：POST请求所发送的数据信息，若为GET请求此参数可省略

* end //当ESP8266发送HTTP请求结束后，我们应该调用此函数来清除ESP8266的

接收缓存以便设备再次接收服务器发来的响应信息

http\_client.end()

* getString //可用于获取服务器响应中的响应体信息，响应体信息将以字符串的形式

进行返回。

http\_client.getString()

返回服务器HTTP响应中的响应体数据，即html代码，可用一字符串变量接受

如：String respond\_code = http\_client.getString()

三、闪存功能

**1、FS库——闪存文件系统操作库（stream数据类型）**

* begin //启动闪存（SPIFFS）

SPIFFS.begin()

对闪存文件系统进行操作之前，需要启动闪存即SPIFFS.begin()，若启动成功返回真，否则返回假

* format //闪存文件系统格式化

SPIFFS.format()

注：无需每次实用闪存都对其格式化

* File date\_file //建立文件操作对象，以便接下来的程序对文件进行操作
* open //对闪存文件进行操作

date\_file = SPIFFS.open(file\_name, “val”)

file\_name：为被操作文件名，格式为字符串格式

val：为操作方法，“w”为写如果文件系统没有file\_name文件，此操作将会在文件系统中建立该文件，如果文件系统有该文件，则程序将会重新建立该文件，即原有文件信息将会被覆盖。“r“为读，一次只能读file\_name中的一个字符，因此需要配合

data\_file.size()进行操作。“a”为追加。

* print 与 println //用于追加或写入字符

SPIFFS.print() //添加字符

SPIFFS.println()//换行添加字符

一般用于写或追加命令之后，如：

file date\_file = SPIFFS.open(file\_name, “w”) //”w”换成“a“即是追加

SPIFFS.println()

* readString //读取字符信息

date\_file.readString()

返回值为读取到的字符串数据

* close //关闭操作文件

date\_file.close()

对文件操作完毕后，需关闭文件

* remove //删除文件

SPIFFS.remove(file\_name)

file\_name格式为字符型。若删除成功则返回真，否则返回假

四、JSON

**1、JSON基础**

* JSON数据：JSON数据以“名”“值”对呈现。数据“名”“值”由冒号分隔。JSON数据的书写格式是：“JSON数据名”：JSON数据值

例如：“Year“：2022

* JSON数据名称

JSON数据名称需要放在双引号中。以下示例都是合法的JSON数据名：

“Value”、”信息1”

* JSON数据值

JSON数据值可以是以下内容：

* 数字（整数或浮点数）

1. "value" : 21

* 字符串 、

1."name" : "陈浩"

* 逻辑值（true 或 false）

1. ” bool\_value”：True

* 数组（在中括号中[ ]）

1. "info": [
2. {
3. "name" : "taichi-maker",
4. "website" : "www.taichi-maker.com"
5. },
6. {
7. "year": 2020,
8. "month": 12,
9. "day": 30
10. }
11. ]

* 对象（在大括号中{}）

1. "info": {
2. "name" : "taichi-maker",
3. "website" : "www.taichi-maker.com"
4. }

* null（表示数据为空）

1. “value” : null

注意：一个JSON数据名称只能对应一个值

* JSON语法规则要点

1）、数据以“名”“值”对呈现

2）、数据“名”和“值”之间由冒号分隔

3）、大括号{}用于标注对象内容

4）、中括号[]用于标注数组内容

5）、逗号用于分隔数据、对象、数组

* JSON对象：JSON对象在大括号{}中书写，对象可以包含单个或者多个JSON数据。

含有单个数据的JSON对象示例：

1. {
2. "name" : "taichi-maker"
3. }

含有多个数据的JSON对象示例

1. {
2. "name" : "taichi-maker",
3. "website" : "www.taichi-maker.com"
4. }

* 注意：
* 对象不能直接存放对象
* 对象也不能直接存放数组
* JSON数组：数组（array） 是相同元素的有序集合。
* 数组可包含一个对象。

1. [
2. {
3. "name" : "taichi-maker",
4. "website" : "www.taichi-maker.com"
5. }
6. ]

* 数组可包含多个对象

1. [
2. {
3. "name" : "taichi-maker",
4. "website" : "www.taichi-maker.com"
5. },
6. {
7. "year": 2020,
8. "month": 12,
9. "day": 30
10. }
11. ]

* 数组也可以包含单个或多个数组

1. [
2. [
3. {
4. "name" : "taichi-maker",
5. "website" : "www.taichi-maker.com"
6. },
7. {
8. "year": 2020,
9. "month": 12,
10. "day": 30
11. }
12. ],
13. [
14. {
15. "temperature" : 15,
16. }
17. ]
18. ]

* 注意：数组不能直接存放JSON数据
* JSON 对象与数组混合存放示例

1. {
2. "results": [
3. {
4. "location": {
5. "name": "Beijing",
6. "country": "CN"
7. },
8. "now": {
9. "text": "Clear",
10. "code": "1",
11. "temperature": "3"
12. },
13. "last\_update": "2020-03-01T20:10:00+08:00"
14. }
15. ]
16. }

* 总结
* 首先注意以下符号的含义：

：用于分隔数据的“名”和“值”

{} 标注对象内容

[ ]标注数组内容

，分隔数据、对象和数组

* 另外注意：

对象（{ }）用于存放数据（名值对）

对象（{ }）不能直接存放对象或数组

数组（[ ]）存放元素有序号（序号起始值0）

数组（[ ]）不能直接存放数据（名值对）

* JSON在线解析工具：<https://c.runoob.com/front-end/53/>

2**、ArduinoJson库**

* const size\_t capacity = JSON\_ARRAY\_SIZE(1) + JSON\_OBJECT\_SIZE(1) + 30

依据json大小创建变量名为capacity的变量用于对json进行操作，后面的+30

是所留裕量，注意：具体大小以及裕量由实际json大小决定，且不好计算

* DynamicJsonDocument doc(capacity) //依据json大小建立DynamicJsonDocument 的doc对象，用于对json进行反序列化（解析）
* deserialize //反序列化数据（解析json）

deserialize(doc,json)

json为被解析的json字符，类型为字符串型

* String name\_str = doc[“name”].as<String>()

String number\_int = doc[“number”].as<int>()

将解析完成的json中名为name的值转换为字符串赋值给name\_str

将解析完成的json中名为number的值转换为整型赋值给number\_int

* 实际json解析较繁琐，这里推荐实用开发ArduinoJson库作者网站中的在线json解析代码生成工具（<https://arduinojson.org/v5/assistant/>），在线解析由V5和V6两个版本,

应依据ArduinoJson库版本选择，我实用的是V5版本的库，故选择V5版在线解析工具。



五、多任务处理

**1、Ticker库——多任务处理**

* Ticker ticker //建立Ticker对象，以便对函数进行调用
* attach //定时调用函数
* ticker.attach(1,say\_hi) //每间隔1s，定时调用say\_hi函数
* ticker.attach(1,say\_hi,6) //每间隔1s，定时调用say\_hi函数并将6传递给say\_hi函数，say\_hi函数需定义为void say\_hi(int count)即多定义一个形参接受6，注意：attach函数所能传递的参数最多只有一个。另外该参数仅能是以下类型中的一种：char, short, int, float, void\*, char\*
* attach\_ms //定时调用函数，与上函数类似，不过定时时间基准为ms级
* detach //停止ticker对象调用函数

ticker.detach()

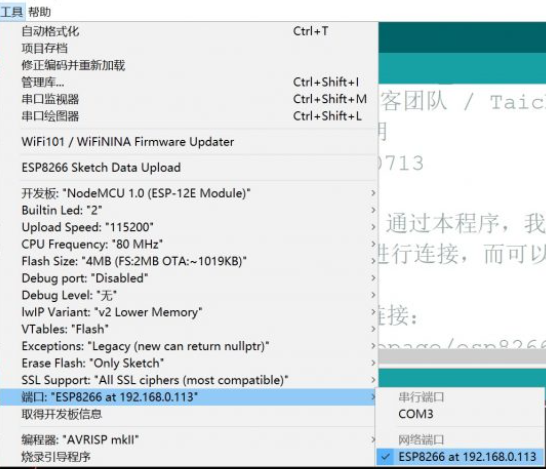
六、隔空传输

**1、ArduinoOTA库——只需ESP8266与电脑在同意WiFi下，即可无需数据线传输程序**

* setHostname //设置主机名，即在选择端口下的名

ArduinoOTA.setHostname(“ESP8266”)

效果如下：



* setPassword //设置连接ESP8266时密码

ArduinoOTA.setPassword(“12345678”)

设置密码为12345678

* begin //启动OTA

ArduinoOTA.begin()

* handle //处理OTA请求（即通过同一wifi下载程序至ESP8266无需数据线），应时刻调用此函数

ArduinoOTA.handle()

七、第三方库

**1、ESP8266\_Seniverse库——ESP8266获取心知天气信息库（针对免费版）**

**1）、获取实时天气信息**

* WeatherNow weather\_now //建立对象名为weather\_now的WeatherNow对象，以便接下来的函数调用
* config //配置心知天气请求信息

weather\_now.config(user\_key,location,unit)

配置用户私钥、位置、温度单位信息

* updata //检验esp8266是否连接到心知天气服务器

weather\_now.updata()

ESP8266成功连接心知天气服务器并且服务器返回响应状态码为200时，此函数将会返回真

* getWeatherText //获取天气信息

weather\_now. getWeatherText()

返回值为天气情况，字符串型

* getWeatherCode //获取天气情况对应代码

weather\_now. getWeatherCode()

返回值为天气情况对应代码，int型

* getDegree //获取温度信息

weather\_now.getDegree()

返回值为温度信息，int型

* getLastUpdate //获取上次更新时间

weather\_now.getLastUpdata()

返回值为天气上次更新时间

**2）、获取近三天（今天、明天、后天）天气预报信息**

* Forecast weather\_forecast //建立Forecast对象,以便接下来函数调用
* config //配置心知天气请求信息

weather\_forecast.config(user\_key,location,unit)

配置用户私钥、位置、温度单位信息

* updata //检验esp8266是否连接到心知天气服务器

weather\_forecast.updata()

ESP8266成功连接心知天气服务器并且服务器返回响应状态码为200时，此函数将会返回真

* getDayText //获取白天天气信息

weather\_forecast.getDayText(i)

参数i为第几天的白天天气信息，i = 0为今天白天天气信息，i = 1为明天白天天气信息，i = 2为后天白天天气信息，返回值为白天天气情况，字符串型

* getDayCode //获取白天天气信息对应的代码

weather\_forecast.getDayCode(i)

参数i为第几天的白天天气信息对应的代码，i = 0为今天白天天气信息对应的代码，i = 1为明天白天天气信息对应的代码，i = 2为后天白天天气信息对应的代码，返回值为白天天气情况对应的代码，int型

* getNightText //获取夜晚天气信息

weather\_forecast.getNightText(i)

参数i为第几天的夜晚天气信息，i = 0为今天夜晚天气信息，i = 1为明天夜晚天气信息，i = 2为后天夜晚天气信息，返回值为夜晚天气情况，字符串型

* getNightCode //获取夜晚天气信息对应的代码

weather\_forecast.getNightCode(i)

参数i为第几天的夜晚天气信息对应的代码，i = 0为今天夜晚天气信息对应的代码，i = 1为明天夜晚天气信息对应的代码，i = 2为后天夜晚天气信息对应的代码，返回值为夜晚天气情况对应的代码，int型

* getHigh(i) //获取对应天数最高温

weather\_forecast.getHigh(i)

返回值为int型

* getLow(i) //获取对应天数最低温

weather\_forecast.getLow(i)

返回值为int型

* getRain(i) //获取对应天数降水概率

weather\_forecast.getRain(i)

返回值为float型

* getWindDirection(i) //获取对应天数风向信息

weather\_forecast.getEWindDirection(i)

返回值为字符串型

* getWindSpeed(i) //获取对应天数风速信息

weather\_forecast.getWindSpeed(i)

返回值为float型

* getWindScale(i) //获取对应天数风力等级信息

weather\_forecast.getWindScale(i)

返回值为int型

* getHumidity(i) //获取对应天数湿度信息

weather\_forecast.getHumidity(i)

返回值为int型

* getLastUpdate(i) //获取对应天数上次更新时间信息

weather\_forecast.getLastUpdate(i)

返回值为字符串型

**2）、获取实时生活指数信息**

* LifeInfo life\_info //建立LifeInfo对象，以便接下来函数的调用
* config //配置心知天气请求信息

life\_info.config(user\_key,location,unit)

* update //检验esp8266是否连接到心知天气服务器

life\_info.update()

若已连接到心知天气服务器，返回真，否则返回假

* getCarWash //获取洗车建议

life\_info.getCarWash()

返回值为字符串型

* getDressing //获取穿衣建议

life\_info. getDressing()

返回值为字符串型

* getFactorFlu //获取流感建议

life\_info. getFactorFlu()

返回值为字符串型

* getExercise //获取锻炼建议

life\_info. getExercise()

返回值为字符串型

* getTravel //获取旅游建议

life\_info. getTravel()

返回值为字符串型

* getUV //获取紫外线建议

life\_info. getUV()

返回值为字符串型

**2、WiFiManager库——ESP8266自动配网库**

**此库**

* WiFiManager wifi\_manager //建立WiFiManager对象，以便调用库中函数
* resetSettings //清除ESP8266的闪存中所存储的WiFi连接信息

WiFiManager.resetSettings()

* autoConnect //自动连接WiFi

WiFiManager.autoConnect(“autoConnectAP”)

WiFiManager.autoConnect(“autoConnectAP”,”12345678”)

说明：autoConnect函数工作流程：



ESP8266首先会读取存在闪存中的WiFi信息进行联网，若闪存中没有WiFi信息或者闪存信息连接不上WiFi信息，ESP8266会启动AP模式（接入点模式），其中“autoConnectAP”与”12345678”中内容分别是ESP8266建立WiFi的名称与密码，其他无线终端（手机或电脑）可以连接该WiFi来配置ESP8266所要连接周围的WiFi名称与密码，之后ESP8266会将本次连接的WiFi名称与密码保存至闪存中，以便下次连接。