Blinker APP：

添加设备：进入app主界面，点击右上角加号-->独立设备-->wifi接入-->阿里云（可以尝试其他选项）-->记录下Secret Key-->返回我的设备

配置设备：进入app主界面，点击上边新建的设备-->点击右上角的铅笔图标-->点击下边的“按键”（屏幕中会多出一个方框）-->点击方框-->在“数据键名”里边填“btn-switch”-->在”按键类型”中选择“开关按键”-->点击右上角“保存”-->再点击右上角小锁形状的按钮。

硬件连接：

若板子为小型的NODE MCU，自带的有板载LED，硬件不需要做改动。

若板子为稍大些的NODE MCU，则需要外接LED，将LED的正极接到3.3V（推荐在3.3V与LED的正极之间串联一个电阻），将LED的负极接到板子的某个引脚上，并需要在程序中将myLed变量初始化为对应的引脚，板子的引脚的对应关系可参考“nodemcu引脚图.PNG“。

PART A：

功能实现：将以下代码编译运行（将之前记录下的Secret Key、即将要连接的WiFi名称和秘密写到对应的位置）下载成功之后，打开串口监视器，按下复位键，当看到“MQTT Connected!”，说明板子联网成功，打开Blinker APP，进入到新建的设备中，按下手机上刚才新建的方框按键，在串口监视器可以看到“you enter the buttun 'switch'!”，PART A部分功能验证成功。

#define BLINKER\_WIFI

#include <Blinker.h>

const char auth[] = "your own Secret Key";

const char ssid[] = "your wifi name ";

const char pswd[] = "your wifi password";

BlinkerButton Button1("btn-switch");

void button1\_callback(const String & state)

{

Serial.println("you enter the buttun 'switch'! ");

}

void setup()

{

Serial.begin(115200);

BLINKER\_DEBUG.stream(Serial);

Blinker.begin(auth, ssid, pswd);

Button1.attach(button1\_callback);

}

void loop() {

Blinker.run();

}

程序分析：BlinkerButton Button1("btn-switch")为新建一个组件Button1，并通过Button1.attach(button1\_callback)为其注册一个回调函数button1\_callback()，当在Blinker APP上点击“btn-switch”这个按钮时，就会进入到回调函数button1\_callback()中，从而在串口监视器上打印出一条提示信息。Blinker.begin(auth, ssid, pswd)将板子通过路由器接入到网络中。

PART B:

功能实现：将以下程序编译下载，按下复位键，打开串口监视器，可以看到“low“和”heigh“交替输出，说明程序在正常运行，同时看到板载的led灯也在按照程序中的时间间隔交替闪烁。PART B部分程序验证成功。

const int myLed = LED\_BUILTIN;

void setup() {

Serial.begin(115200);

pinMode(myLed, OUTPUT);

}

void loop() {

digitalWrite(myLed, LOW);

Serial.println("low");

delay(1000);

digitalWrite(myLed, HIGH);

Serial.println("heigh");

delay(2000);

}

程序分析：LED\_BUILTIN为板载led的引脚，将其赋值到myLed，myLed的值可以自定义更改；pinMode(myLed, OUTPUT)将myLed引脚配置为输出，通过digitalWrite()函数来将myLed引脚配置为高电平或者低电平，从而控制led的亮或者灭。

PART C：

功能实现：将以下程序编译、下载，依据PART A部分介绍，按下Blinker APP中的按键，即可看到LED由灭变为亮，再按一下按键，LED的状态发生反转，实验一功能验证成功。

#define BLINKER\_WIFI

#include <Blinker.h>

const char auth[] = "your own Secret Key";

const char ssid[] = "your wifi name ";

const char pswd[] = "your wifi password";

BlinkerButton Button1("btn-switch");

const int myLed = LED\_BUILTIN;

void button1\_callback(const String & state)

{

digitalWrite(myLed, !digitalRead(myLed));

if(digitalRead(myLed))

Serial.println("get button state: off");

else

Serial.println("get button state: on");

}

void setup()

{

Serial.begin(115200);

BLINKER\_DEBUG.stream(Serial);

pinMode(myLed, OUTPUT);

digitalWrite(myLed, HIGH);

Blinker.begin(auth, ssid, pswd);

Button1.attach(button1\_callback);

}

void loop() {

Blinker.run();

}

程序分析：将PART A与PART B部分想结合，回调函数button1\_callback()中的函数digitalWrite(myLed, !digitalRead(myLed))的作用为将LED的状态进行反转，其中函数digitalRead(myLed)的作用为读取myLed引脚的电平状态，前边的叹号为取反的作用，并将取反后的结果赋值到myLed引脚上，从而实现了将LED灯的状态取反的功能；然后根据digitalRead(myLed)函数读取myLed的电平状态，来输出对应的提示信息到串口监视器上。