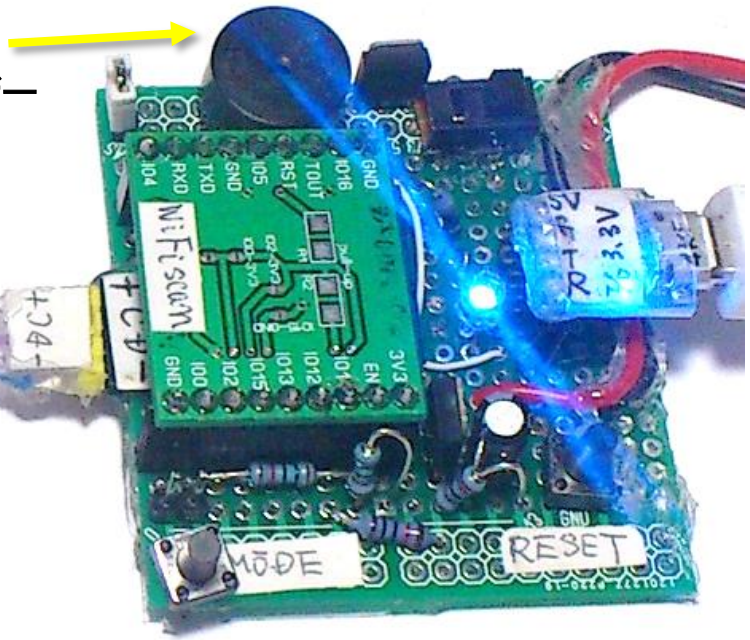
ESP-WROOM-02(ESP8266) httpサーバ設定によるスマホ等への送信

2017.3.8

参考にさせて頂いた他人様のページ　https://www.mgo-tec.com/blog-entry-ss-wroom-howto01.html

Strawberry -linux の1602D1（バックライト付き）も
全く同じスケッチで、動作可能.

スマホ（client）からの
アクセス要求チェックブザー



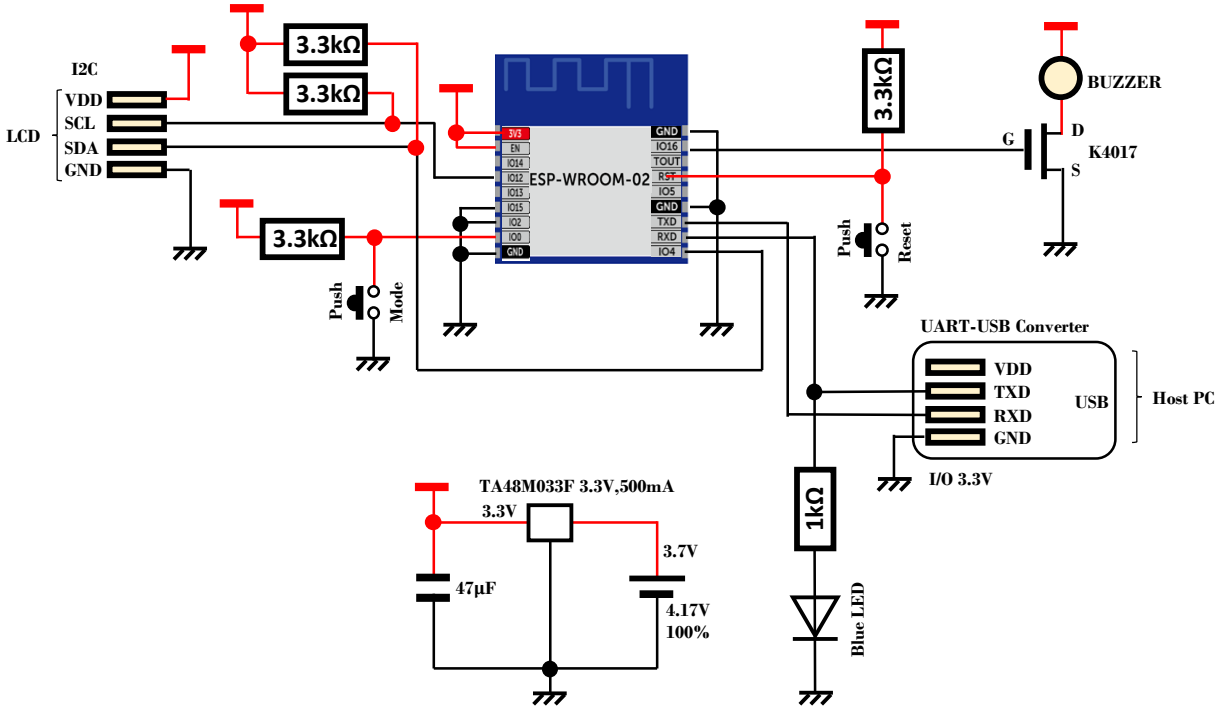
Host PC

送信時画面

http サーバ設定

起動時画面
APへの接続動作

スマホ受信画面



Pushボタン操作なしで電源ON ⇒ 本スケッチが起動する

利用させて頂いた ESP8266他人様の元図 http://qiita.com/umi_kappa/items/ac3d37db44a2dcfe71fd

```
/* 2017.2.25 "WiFi_ESP8266_Trans_Smapho.ino"
   ---- Send "Hello World" to Smartphone ... ----
   WiFi module : WROOM-02 by aitendo
   LCD  : 1602D1 by Strawberry-Linux
      or AQM1602 by Akizuki

   ref :https://www.mgo-tec.com/blog-entry-ss-wroom-howto01.html
   コメント等、そのまま引用させて頂いています
*/

#include "ESP8266WiFi.h"
#include <Wire.h>

#define ADDR   0x3e
#define C_Low  0x70
#define C_High 0x56   //

#define AQM1602

#ifdef  AQM1602
#define Cont     6   // Contrast AQM1602 Akizuki
#else
#define Cont     10   // Contrast 1602D1 Strawberry
#endif

#define BUZZER 16

#define WIFI_SSID "planexuser"
//#define WIFI_PSK  "**********" //
#define WIFI_PSK    "**********" //
//#define WIFI_SSID "**********" //
//#define WIFI_PSK  "**********" // Private

#define DEST_HOST "api.fixer.io"
#define DEST_PORT 80
#define DEST_URL  "/latest?base=USD&symbols=JPY"

/* ********************************************************* */
//ご自分のルーターのSSIDを入力してください
const char* ssid     = WIFI_SSID;
//ご自分のルーターのパスワード
const char* password = WIFI_PSK;

boolean Ini_html_on = false;
//ブラウザからの初回HTTPレスポンス完了したかどうかのフラグ

WiFiServer server(80);
WiFiClient client;

void setup() {
  uint8_t cmd_init1[] = {0x38, 0x39, 0x14};
  uint8_t cmd_init2[] = {C_Low | (Cont & 0x0f), C_High | (Cont >> 4 & 3), 0x6c};
  uint8_t cmd_init3[] = {0x38, 0x0c, 0x01};  // 0x0d => 0x0c ?
  uint8_t mess_1[]    = "Started server !";
  uint8_t point[]     = {0x2e};

  pinMode(BUZZER, OUTPUT);
  digitalWrite(BUZZER, LOW);

  Serial.begin(115200);
  // Connect to WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  delay(10);
  // Wire.begin(4, 14);    // (SDA,  SCL)
  Wire.begin(4, 12);       // (SDA,  SCL)
  delay(40);
  /*  LCD init */
  command(cmd_init1, sizeof(cmd_init1));
  command(cmd_init2, sizeof(cmd_init2));
  delay(300);
  command(cmd_init3, sizeof(cmd_init3));
  delay(2);
  clear_LCD();
  delay(1000);
  write(mess_1, sizeof(mess_1));

  CRLF_LCD();
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
    write(point, sizeof(point));
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.println(WiFi.localIP());

  delay(1000);
  clear_LCD();
}
```

```
uint8_t  ssi[]   = WIFI_SSID;
uint8_t  hello[] = "Send Hello World";
// ************メインループ************
void loop() {
  Disp_LCD();

  while (1) {
    if (Ini_html_on == false) {
      Ini_HTTP_Response();
    } else if (client.available()) {
      Serial.println("client.available()");
      Serial.print(client.read());
    }
    delay(100);//これは重要かも。これがないと動作しないかも。
  }
}

//      *****  初回ブラウザからのGET要求による
//      HTMLタグ吐き出しHTTPレスポンス  *******
void Ini_HTTP_Response()
{
  client = server.available();//クライアント生成
  delay(1);
  String req;

  while (client) {

    if (client.available()) {

      req = client.readStringUntil('¥n');
      Serial.println(req);
      if (req.indexOf("GET / HTTP") >= 0 || req.indexOf("GET /favicon") >= 0) {
        //ブラウザからリクエストを受信したらこの文字列を検知する
        //Google Chromeの場合faviconリクエストが来るのでそれも検出する
        Serial.println("-----from Browser FirstTime HTTP Request---------");
        Serial.println(req);

        //ブラウザからのリクエストで空行（¥r¥nが先頭になる）まで読み込む
        while (req.indexOf("¥r") != 0) {
          req = client.readStringUntil('¥n');
          //¥nまで読み込むが¥n自身は文字列に含まれず、捨てられる
          Serial.println(req);
        }
        req = "";
        delay(10);//10ms待ってレスポンスをブラウザに送信
        //メモリ節約のため、Fマクロで文字列を囲う
        //普通のHTTPレスポンスヘッダ
        client.print(F("HTTP/1.1 200 OK¥r¥n"));
        client.print(F("Content-Type:text/html¥r¥n"));
        client.print(F("Connection:close¥r¥n¥r¥n"));// 1 行空行が必要
        //ここからブラウザ表示のためのHTML JavaScript吐き出し
        client.println(F("<!DOCTYPE html>"));
        client.println(F("<html>"));
        client.println(F("<font size=30>"));
        client.println(F("Hello World<br>"));
        client.println(F("</html>¥r¥n"));
        delay(1);//これが重要！これが無いと切断できないかもしれない。
        client.stop();//一旦ブラウザとコネクション切断する。

        Serial.println("¥nGET HTTP client stop----------------------");
        req = "";
        Ini_html_on = false;  //一回限りの接の場合、ここをtrueにする

        Blink_LCD();
      }
    }
  }
}

void Blink_LCD() {
  for (int i = 0; i < 4; i++) {
    clear_LCD();
    digitalWrite(BUZZER, HIGH);
    delay(300);
    Disp_LCD();
    digitalWrite(BUZZER, LOW);
    delay(300);
  }
}
void Disp_LCD() {
  write(ssi, sizeof(ssi));
  CRLF_LCD();    // CRLF
  write(hello, sizeof(hello));
}

void CRLF_LCD() {
  uint8_t cmd_cr[] = {0xc0};  // C/R
  command(cmd_cr, sizeof(cmd_cr));
  delay(5);
}

void clear_LCD() {
  uint8_t cmd_cl[] = {0x01};  // clear command
  command(cmd_cl, sizeof(cmd_cl));
  delay(5);
}

void command(uint8_t *cmd, size_t len) {
  size_t i;
  for (i = 0; i < len; i++) {
    Wire.beginTransmission(ADDR);
    Wire.write(0x00);
    Wire.write(cmd[i]);
    Wire.endTransmission();
    delayMicroseconds(30);    // 26.3us
  }
  delay(1);
}

void write(uint8_t *cmd, size_t len) {
  size_t i;
  for (i = 0; i < len; i++) {
    Wire.beginTransmission(ADDR);
    Wire.write(0x40);
    delayMicroseconds(30);    // 26.3us
    Wire.write(cmd[i]);
    Wire.endTransmission();
    delayMicroseconds(30);    // 26.3us
  }
  delay(1);
}
```