

Webサーバー プログラム

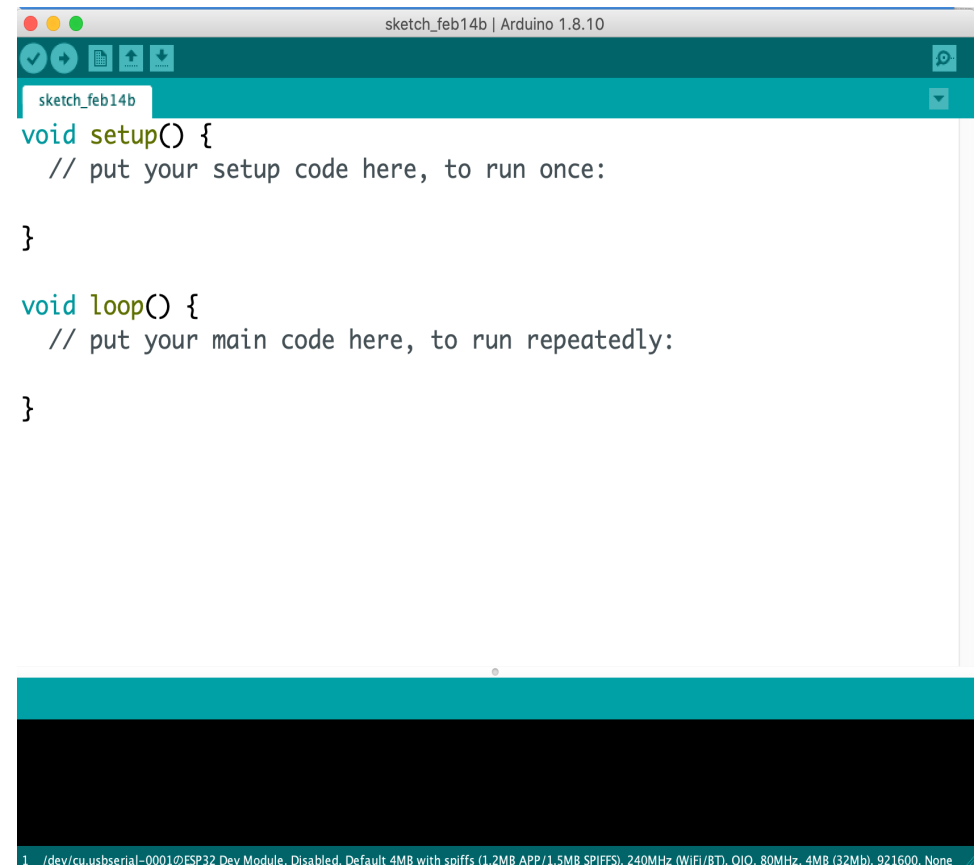
2021.2.15



Arduinoプログラミングの基礎

setupとloopという名前の関数(一連の処理をまとめたもの)がありますね。この2つが最低限必要な関数になっていて、マイコンが起動するとsetup(){から}までに書いてある命令を1回実行します。その後loop(){から}に書いてある命令を繰り返し実行し続けます。

//で始まる行はプログラムの命令ではなく、コメントと言って注意書きのようなものです。ですので、今は中身は何もありません。ここに自分が動作させたい内容を記述していくことになります。



```
sketch_feb14b | Arduino 1.8.10
sketch_feb14b
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}

1 /dev/cu.usbserial-0001のESP32 Dev Module, Disabled, Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None
```

このプログラム（スケッチ）は、
シリアルモニターにIPアドレスを表示し（接続後）、
そのIPアドレスでWebブラウザを開きます
ブラウザ上の” **here**”という文字列をクリックすると
pin23につながれたLEDをオンまたはオフにします。

開発ボードのIPアドレスで呼び出して次の例では：
http:// 192.168.100.xxx / H でLEDをオンにします
http:// 192.168.100.xxx / L でLEDをオフにします

* LEDは、GPIO pin 23に接続する。

無線ルーターのSSIDやパスワードを指定して、ネットワークを特定する。

```
#include <WiFi.h>
```

```
const char* ssid = "SPWH_H33_ED1E59";
```

```
const char* password = "j98mi6i6eq0ygq9 ";
```

```
WiFiServer server(80);
```

プログラムの設定部分

```
void setup()
```

```
{
```

```
  Serial.begin(115200);  
  pinMode(23, OUTPUT);
```

```
  delay(10);
```

//各種設定の部分

// 通信スピードは、115200BPS.

// LEDは、GPIO pin 23に接続する。

//10ms待ちます。

シリアルモニターに通信状況を表示する部分

// WiFi ネットワークに接続する。

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

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

```
Serial.print("Connecting to ");
```

```
Serial.println(ssid);
```

// Serial.println()は、シリアルモニタに表示する命令。

// 空白行の表示

// Connecting to.の文字列表示

// SSIDの内容を文字列表示

```
WiFi.begin(ssid, password);
```

// WiFi接続開始。

```
while (WiFi.status() != WL_CONNECTED) {
```

// 現在の接続状態を返却します。

// アクセスポイントに接続しているときは、

// WL_CONNECTEDが返ってきます。

```
    delay(500);
```

// 500ms待ちます。

```
    Serial.print(".");
```

// wi-fiが接続するまで・・・の表示が続く

```
}
```

シリアルモニターに通信状況を表示する部分

```
Serial.println("");  
Serial.println("WiFi connected."); // WiFi connected.の文字列表示  
Serial.println("IP address: "); // IP address:の文字列表示  
Serial.println(WiFi.localIP()); // 現在割り当てられているIPアドレスを表示します。  
  
server.begin(); // WiFiサーバー機能開始。  
  
}  
  
int value = 0; // valueという変数を初期化。
```

Wi-Fi通信開始直後のシリアルモニタ画面

```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1044
load:0x40078000,len:8896
load:0x40080400,len:5816
entry 0x400806ac

Connecting to SPWH_H33_ED1E59
.
WiFi connected.
IP address:
192.168.100.139
```


プログラムは、この手順を繰り返します。

```
void loop(){
```

```
WiFiClient client = server.available(); // ユーザーのスマホからの接続を取得し、  
// WiFiClient型の変数clientに接続かどうか代入します。  
// ユーザーから接続がない場合は、if (client)が、falseになります。  
// 以降は、clientを用いて、WiFiユーザー要求とのやり取りを行います。  
  
if (client) { // ユーザーの要求が見つかれば  
  Serial.println("New Client."); // シリアルポートにNew Client.のテキストを出力します。  
  String currentLine = ""; // ユーザーからの受信データを保持する文字列を作成します  
  while (client.connected()) { // ユーザーとの接続状態を返します。  
    if (client.available()) { // 読み込み可能なバイト数を取得します。  
      char c = client.read(); // 1文字読みます。  
      Serial.write(c); // 読み込んだテキストをシリアルモニタに出力します。  
      if (c == '\n') { // 改行コードを読んだらシリアルモニタ画面で改行  
        // 現在の行が空白の場合、2つの改行文字が連続して表示されます。  
        // これでクライアントのHTTPリクエストが一旦終了し、応答を送信します。
```

LEDのON-Offを命令したときにシリアルモニタ表示される。



The screenshot shows a serial terminal window titled "/dev/cu.usbserial-0001". The terminal displays the following text:

```
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; POT-LX2J) AppleWebKit/537.36 (
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,i
Referer: http://192.168.100.139/L
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

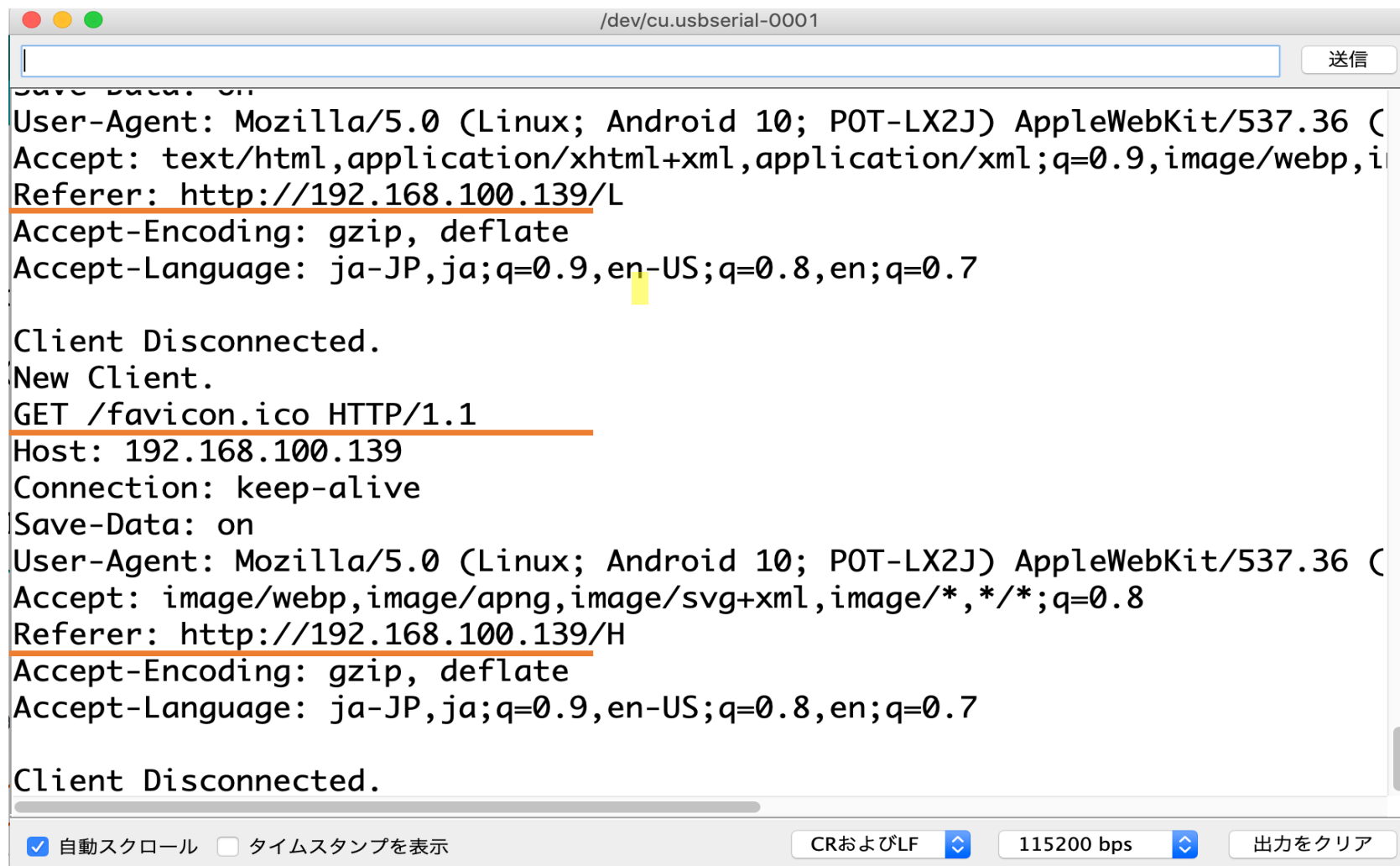
Client Disconnected.
New Client.
GET /favicon.ico HTTP/1.1
Host: 192.168.100.139
Connection: keep-alive
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; POT-LX2J) AppleWebKit/537.36 (
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.100.139/H
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

Client Disconnected.
```

At the bottom of the window, there are several controls: a checked checkbox for "自動スクロール" (Auto scroll), an unchecked checkbox for "タイムスタンプを表示" (Show timestamps), a dropdown menu for "CRおよびLF" (CR and LF) set to "\n\r", a dropdown menu for "115200 bps" (Baud rate), and a button labeled "出力をクリア" (Clear output).

```
if (currentLine.length() == 0) {  
    // HTTPヘッダーは、常に応答コード（HTTP / 1.1 200 OKなど）  
    // とコンテンツタイプで始まるため、  
    // クライアントは何が来るのかを認識し、その後に空白行を表示します。  
    client.println("HTTP/1.1 200 OK");           // HTTPリクエストを書き込みます。  
    client.println("Content-type:text/html");  
    client.println();  
  
    // クライアント（スマホやPC）の画面に下記テキストを表示。  
    client.print("Click <a href=¥"/H¥">here</a> to turn the LED on pin 5 on.<br>");  
    client.print("Click <a href=¥"/L¥">here</a> to turn the LED on pin 5 off.<br>");  
  
    // HTTP応答は、別の空白行で終了します。  
    client.println();  
    // whileループから抜け出します。  
    break;
```

LEDのON-Offを命令したときにシリアルモニタ表示される。



The screenshot shows a serial terminal window titled "/dev/cu.usbserial-0001". The terminal displays the following text:

```
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; P0T-LX2J) AppleWebKit/537.36 (
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,i
Referer: http://192.168.100.139/L
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

Client Disconnected.
New Client.
GET /favicon.ico HTTP/1.1
Host: 192.168.100.139
Connection: keep-alive
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; P0T-LX2J) AppleWebKit/537.36 (
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.100.139/H
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

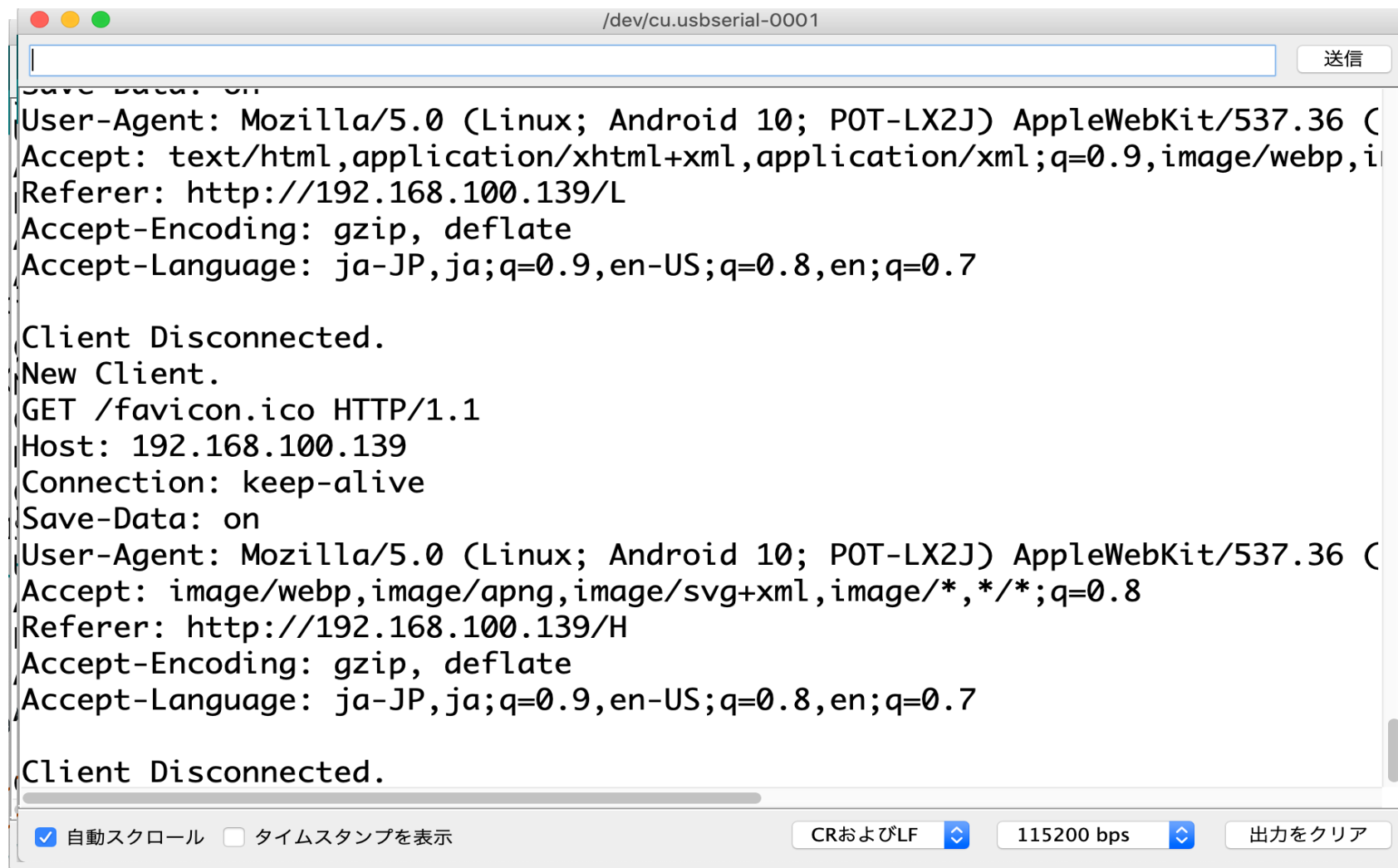
Client Disconnected.
```

At the bottom of the window, there are several controls: a checked checkbox for "自動スクロール" (Auto Scroll), an unchecked checkbox for "タイムスタンプを表示" (Show Timestamps), a dropdown menu for "CRおよびLF" (CR and LF) set to "⌵", a dropdown menu for "115200 bps" (Baud Rate) set to "⌵", and a button labeled "出力をクリア" (Clear Output).

```
} else { // if you got a newline, then clear currentLine:
    currentLine = "";
}
} else if (c != '\r') { //もし改行コードなら
    currentLine += c; // 現在の行にそれを加えます。
}

// ユーザーからの要求が "GET /H" or "GET /L":かどうか調べる。
if (currentLine.endsWith("GET /H")) {
    digitalWrite(23, HIGH); // もしGET /H なら LED 点灯。
}
if (currentLine.endsWith("GET /L")) {
    digitalWrite(23, LOW); // もし GET /L なら LED 消灯。
}
}
}
// 接続終了:
client.stop();
Serial.println("Client Disconnected."); // 接続終了の表示。
}
```

LEDのON-Offを命令したときにシリアルモニタ表示される。



The image shows a serial terminal window titled "/dev/cu.usbserial-0001". The window contains the following text:

```
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; POT-LX2J) AppleWebKit/537.36 (
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,i
Referer: http://192.168.100.139/L
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

Client Disconnected.
New Client.
GET /favicon.ico HTTP/1.1
Host: 192.168.100.139
Connection: keep-alive
Save-Data: on
User-Agent: Mozilla/5.0 (Linux; Android 10; POT-LX2J) AppleWebKit/537.36 (
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.100.139/H
Accept-Encoding: gzip, deflate
Accept-Language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7

Client Disconnected.
```

At the bottom of the window, there are several controls: a checked checkbox for "自動スクロール" (Auto scroll), an unchecked checkbox for "タイムスタンプを表示" (Show timestamps), a dropdown menu for "CRおよびLF" (CR and LF) set to "CRおよびLF", a dropdown menu for "115200 bps" (Baud rate), and a button labeled "出力をクリア" (Clear output).