RaspiOnZumo の Wi-Fi コントロール（WebIOPi）
システム構築の手引き

Ver1.6 2018.01.16

概要:

このプロジェクトは RaspiOnZumo を Wi-Fi を通じてパソコンやスマホ（Android, iPhone）のブラウザから操縦する仕組みを構築するものです。（図 1）Zumo Robot for Arduino にはコントロールボードの ZumoShield が搭載されています。ZumoShield は Arduino に接続されるシールドとして機能します。

① ZumoShield と Arduino Leonardo（以後 Arduino と表記）は 4 本のピンで接続されています。Arduino はこの 4 本のピンを使って ZumoShield 上のモーターコントロール LSI を通じ、左右のモーターの回転方向と速度をコントロールします。

② Zumo のモーターをコントロールするために、Arduino に前もって Pololu 社製の ZumoMotorLiburary を利用した MotorControl.ino プログラムを RaspberryPi3(以後 RasPi と表記)上の ArduinoIDE からコンパイル・アップロードしておく必要があります。このプログラムは RasPi から USB を通じて送られてくる前進、後退などのコマンド文字（'F','B'など）を待ち受け、それに応じて Zumo のモーターをコントロールするものです。

③ RasPi には GPIO ピンや USB シリアル等をネットワークからコントロールできる「WebIOPI」サーバーシステムを導入し、この Web ページに操縦ボタンを作り込み、ボタンが押されたら USB 通信で Arduino にコマンド文字を送るようにします。

④ 全体として、スマホや PC のブラウザからコントローラ Web ページに Wi-Fi 経由でアクセスし、Zumo をコントロールすることになります。

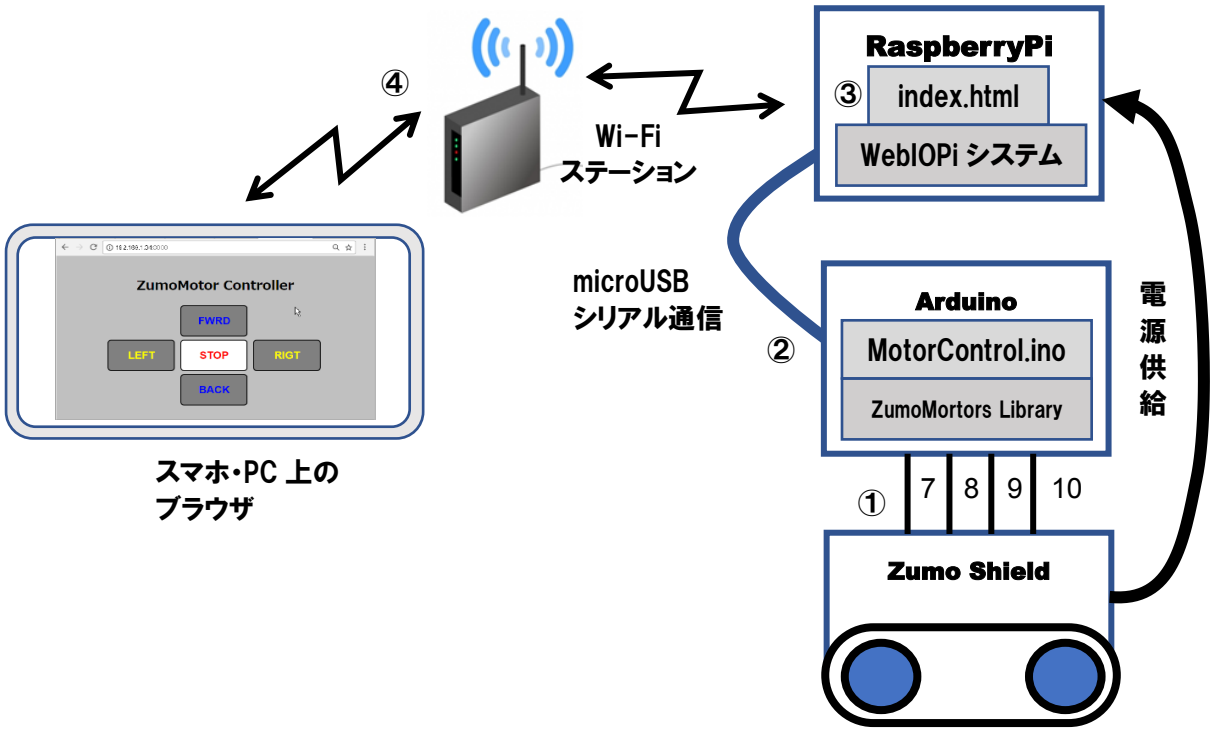


図 1　RaspiOnZumo 全体構成図

用意していただくもの

1．弊社 RasberryPiOnZumo(RaspberryPi3 + Arduino Leonardo + ZumoRobot for Arduino v1.2) すでにその３つが一体として組み立てられていることを前提にしています。またシャットダウン用のピンスイッチも 36、37 ピンに差しておいてください。
（RasPi には Raspbian をインストールした SD カードがセットされているとする）
2．充電済みニッケル水素電池４本（Panasonic Eneloop など）＋バッテリーチャージャー
3．Wi-Fi 環境（Wi-Fi ステーション）（RasPi は Wi-Fi に接続済みであるとする）
4．RasPi 使用に必要な周辺機器（ディスプレイ、キーボード、マウス、電源など）
5．RaspiOnZumo のモーターが駆動したときに走らないように Zumo の下に置く小箱。

図 2　RaspiOnZumo 開発のようす

電源供給について

1．RasPi から Arduino をコントロールする USB ケーブルは RasPi の電源を入れる前に接続をしておいてください。
2．開発時は RaspberryPi への電源供給は AC アダプタを使用し、Zumo 電源 USB ケーブルははずして置いて下さい。（開発には長時間かかるので、Zumo の電池消費を抑えるため）
3．運用時は RaspberryPi へ Zumo 電源ケーブルを接続します。（バッテリーフル充電でおよそ 2 時間の連続運用が可能です）
4．RasPi の電源を切り替える前に、必ず RasPi のシャットダウン処理を忘れないで下さい。（不意にシャットダウンをすると SD カードの内容が壊れる可能性があります）

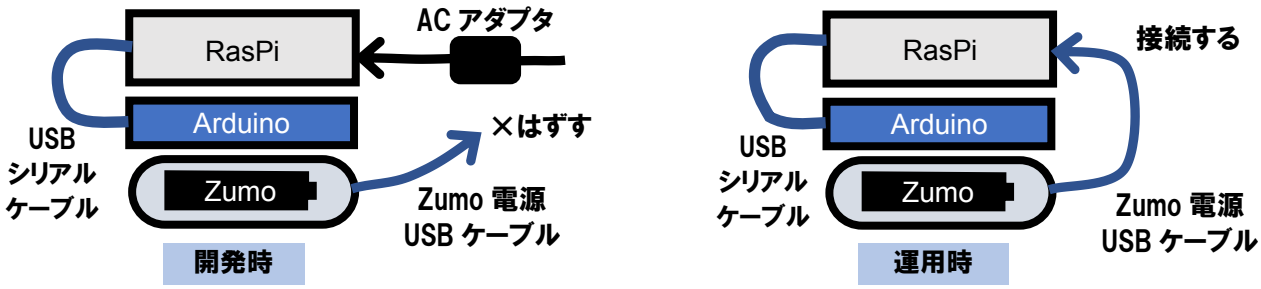


図 3　開発時と運用時の電源供給

構築の手順の概要：

RaspiOnZumo の組み立てについては以下の URL を参照してください。

http://physical-computing-lab.net/raspberry-pi/raspi-on-zumo-kit-man-01.html

1．WebIOPi システムを RasPi へインストールする。（WebIOPi は RasPi 上に Web サーバーを構築し、スマホなどのブラウザに現れるボタンを使って GPIO ピンやシリアルをコントロールする IoT システムを作り上げるサーバーシステムです）

2．Arduino IDE を RasPi へインストールし、Zumo Motors Library をインストールする。

3．Zumo 上の Arduino Leonardo に Raspi からコマンドを受け付け、Zumo を駆動するためのスケッチ（Arduino のプログラムのこと）を書き込む。

4．RasPi 上の Python を使って USB シリアル経由で Zumo のモーターをコントロールしてみる。（これは完成システムには関係ありませんが、RasPi から USB シリアル経由で Zumo がコントロールできるかどうかを確認するためのものです。省略しても構いません）

5．WebIOPi を使って RaspiOnZumo をコントロールする Web ページを作成する。

6．スマホから RaspiOnZumo の Web ページにアクセスしてコントロールする。

7．RasPi に GPIO ピンにシャットダウンボタンを取り付け、シャットダウンのスクリプトを書き込む。

8．RasPi 上に自動起動のスクリプトを書き込む。

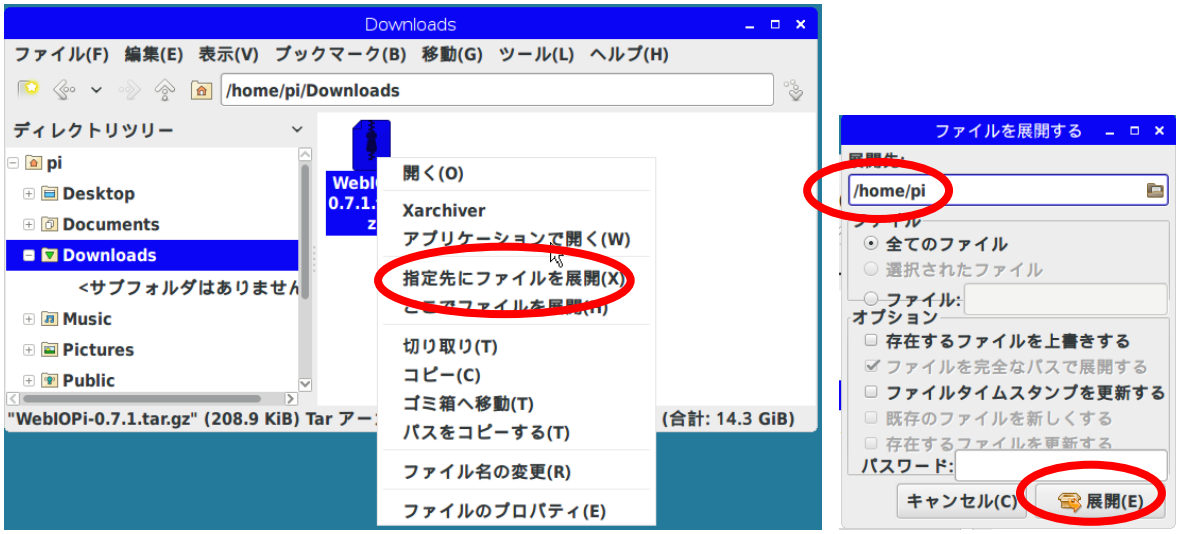
構築の実際：

1. WebIOPiのダウンロード：RasPiデスクトップ上で Chromium Web Brouser から WebIOPi の WebPage（下記 URL）にアクセスし、ページ内の WebIOPi-0.7.1.tar.gz を左クリックしてダウンロードします。

http://webiopi.trouch.com/DOWNLOADS.html

2. WebIOPi のインストール

① **WebIOPI の展開：**ファイルマネージャーを使って、**/home/pi/Downloads** にダウンロードされた **WebIOPi-0.7.1.tar.gz** を右クリックし、「指定先にファイルを展開」を選び、展開先：**/home/pi** と入力して「展開」ボタンを押す。これにより **/home/pi/WebIOPi-0.7.1** フォルダが作成され、



内部に **WebIOPi** 関連ファイルが展開されます。

② **WebIOPi にパッチを当てる（RaspberryPi2, RaspberryPi3 で必要）**

LxdeTerminal を起動し、

$ cd WebIOPi-0.7.1

で **WebIOPi-0.7.1** ディレクトリに入ったのち、

$ wget https://raw.githubusercontent.com/doublebind/raspi/master/webiopi-pi2bplus.patch

にてパッチファイル（修正ファイル）をダウンロードします。

$ patch -p1 -i webiopi-pi2bplus.patch

にてパッチを当てます。

③ **インストールスクリプトの実行**

$ sudo ./setup.sh

にてインストールスクリプトを実行します。最後にインターネットアクセスを n で答えて終了。

Do you want to access WebIOPi over Internet ? [y/n]

n

WebIOPi successfully installed

④ **WebIO サービスのダウンロードとインストール**

$ wget https://raw.githubusercontent.com/neuralassembly/raspi/master/webiopi.service

webiopi.service を /etc/systemd/system/に移動

$ sudo mv webiopi.service /etc/systemd/system/

３．WebIOPi サーバーの起動

ここまでインストールできたら、デバッグモードで WebIOPi の起動と終了をしてみます。(他の起動方法については、この手引きの最後に記載してあります。)

$ sudo webiopi -d -c /etc/webiopi/config

終了は ctrl-C を押します。

RasPi のネットワークアドレスを調べる

次に Wi-Fi ルーターから DHCP で割り振られた Raspi のアドレスを lxdeTerminal から以下のコマンドで調べます。

$ ifconfig

```
eth0      Link encap:イーサネット  ハードウェアアドレス xx:xx:xx:xx:xx:xx
          .........
lo        Link encap:ローカルループバック
          .........
wlan0     Link encap:イーサネット  ハードウェアアドレス xx:xx:xx:xx:xx:xx
          inet アドレス:192.168.xx.xx  ブロードキャスト:192.168.1.255  マスク:255.255.255.0
```

この中の wlan0 の後の inet アドレス：**192.168.xx.xx** が RasPi のアドレスです。これを控えておいてください。
ここで、**inet-address** がない場合、Wi-Fi ルーターが作動していない、RasPi の Wi-Fi が設定されていないなどの可能性があります。

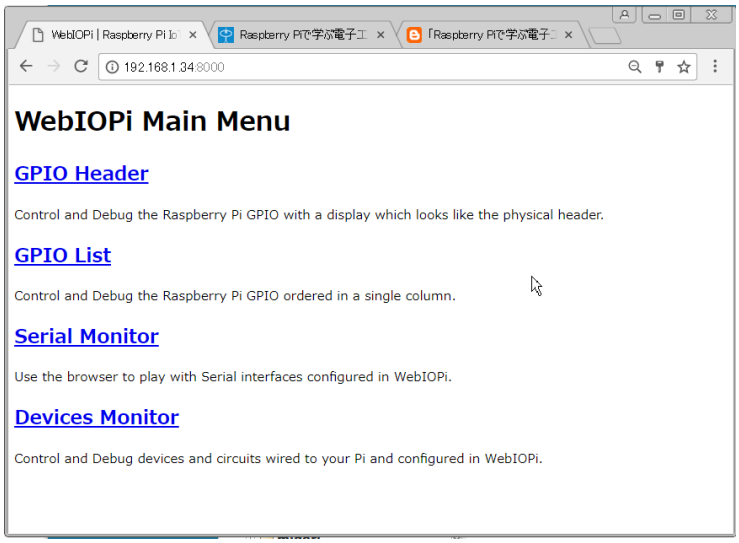
ブラウザから RasPi の WebIOPi にアクセスする

Wi-Fi に接続されたパソコン、スマホなどから

http://192.168.xx.xx:8000/

にアクセスすると下のようなウィンドウが開き WebIOPi へのアクセスの認証を求められます。

ここで、ユーザー名： **webiopi**　パスワード：**raspberry** を入力してログインします。
すると下のようなデモページが表示されれば成功です。



ここで GPIOHeader のページから RaspberryPi の GPIO ピンを遠隔で操作することができますが、ここでは詳しくは触れません。（ここではピンのコントロールではなく、USB の通信を使うため）関心のある方は WebIOPi 関連のホームページをごらんになって下さい。）

4. RaspberryPi 上に ArduinoIDE をインストール

次に Arduino のプログラム統合開発環境 ArduinoIDE(linux 版)を RasPi 上にダウンロード、インストールします。Terminal から次のようにタイプしてください。

```
$ sudo apt-get update
$ sudo apt-get install arduino
```

最後に**[y/n]**を聞いてくるので **y** で答えます。
インストールが完了したら、**Desktop** のメニューから
プログラミング＞arduino IDE
で **arduinoIDE** を起動し、終了しておきます。すると**/home/pi** の下にディレクトリ **sketchbook** と、さらにその下にディレクトリ **libraries** が作られることを確認しておきます。
保存したプログラムは **sketchbook** ディレクトリ内に、取り込んだライブラリは **library** ディレクトリ内に保存されます。

5. ZumoMotors のライブラリを GitHub からダウンロード

```
$ git clone https://github.com/pololu/zumo-shield.git
```

上記コマンドにより**/home/pi/**内に **Zumo** 関連のファイルがダウンロードされます。
/home/pi の下に**ディレクトリ Zumo-shield**、その下に６つのディレクトリが作られるます。

```
Pushbutton  QTRSensors  ZumoBuzzer  ZumoExamples
  ZumoMotors   ZumoReflectanceSensorArray
```

このうちの **ZumoMortors** をフォルダごと **/home/pi/sketchbook/libraries** の中に
コピーします。 **/home/pi** にいるとして

```
$ cp -r zumo-shield/ZumoMotors  sketchbook/libraries
```

これで Zumo Motor Library が ArduinoIDE にセットされました。

6. Arduino Leonardo に RasPi 上の ArduinoIDE からモーターコントロールのプログラムを書き込む

右が ZumoMotor ライブラリを使って RasPi からコマンドを受け取るプログラムです。

これは USB を通じて RasPi から送られてきたコマンド（'S'停止、'F'前進、'L'左、'R'右、'B'後退)を受け取り、ZumoMotor ライブラリを使用してモーターを制御するものです。

Zumo モーターライブラリの関数である **setLeftSpeed()**、**setRightSpeed()**は **-400**（後退）から**+400**（前進）まで（ **0** は停止）スピードを設定できます。ライブラリは Arduino の 4 本のピン **7,8,9,10** に接続されたモーターコントローラーIC を介して左右のモータを制御しています。**7,8** ピンの **HIGH,LOW** デジタル制御で左右の前進後退を決定し、**9,10** ピンのアナログ出力**(PWM)**で左右の回転速度を決めています。

```
#CommandReceiver.ino
#include <ZumoMotors.h>
#define SP 160 //Speed
ZumoMotors motors;

void setup(){
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void setSpeed(int left, int right){
  motors.setLeftSpeed(left);
  motors.setRightSpeed(right);
}

void loop(){
  if(Serial.available()>0){
    int ch = Serial.read();
    switch( ch ){
      case 'S': setSpeed(   0,    0 ); break;
      case 'F': setSpeed(  SP,   SP ); break;
      case 'L': setSpeed(   0,  SP ); break;
      case 'R': setSpeed(  SP,    0 ); break;
      case 'B': setSpeed( -SP, -SP ); break;
      case '1': digitalWrite(13, HIGH); break;
      case '0': digitalWrite(13, LOW ); break;
    }
    delay(2);
  }
}
```

7. ArduinoIDE でマイコンボード、シリアルポートを設定する

プログラムを書き込むときは Arduino IDE のメニューから

ツール＞マイコンボード

＞Arduino Leonardo

ツール＞シリアルポート

＞/dev/ttyACM0

を指定します。

上のプログラムを Arduino IDE に打ち込み、

CommandReceiver の名前で保存します。

保存すると、

/home/pi/sketchbook/CommandReciever.ino

として保存されます。

IDE のメニューからマイコンボードに書き込む（→右矢印）を選んで、Arduino にプログラムを書き込みます。

8. Raspi 上の Python から Serial を通じて Zumo が動くか試す

ここでは RasPi の Python から USB（serial）を使ってコマンドを Arduino に送り、それを受け取って Zumo が動作するか試すことが目的です。

RasPi デスクトップから　メニュー＞プログラミング＞python 2(IDLE)を起動
右の Python プログラムを打ち込み、
ファイル名 **SerialTest.py** で
/home/pi/wifi-project ディレクトリに保存します。

※注意:後から Arduino を接続するとシリアルポートが変化してコマンドを送ることができなくなることがあります。RasPi の電源を投入する前に USB ケーブルで Arduino を接続しておいてください。

成功すれば、Zumo は前進、後退、右旋回、左旋回を繰り返します。

キーボードから **Ctrl-C** で止めてください。

```
# -*- coding: utf-8 -*-
# SerialTest.py

import serial
import time

ser = serial.Serial('/dev/ttyACM0', 9600 )
try:
    while True:
        ser.write('F')
        time.sleep( 1 )
        ser.write('B')
        time.sleep( 1 )
        ser.write('S')
        time.sleep( 3 )
        ser.write('L')
        time.sleep( 1 )
        ser.write('R')
        time.sleep( 3 )

except KeyboardInterrupt:
    print "Ctrl-C Pushed."
    pass

ser.write('S')
print "Finished."
```

9．RaspiOnZumo 用 WebIOPi 作業ディレクトリを作る

/home/pi の下に **wifi-project** を作り、さらにその下に **html** と **python** を作ります。

```
$ mkdir wifi-project
$ cd wifi-project
$ mkdir html
$ mkdir python
cd
$ (/home/pi にもどりました)
```

その結果、次のようなディレクトリができあがります。

```
/home/pi
      |---/wifi-project
            |---/html
            |---/python
```

１０．コントローラーの Web Page 作成

/home/pi/wifi-project の下の **Html** ディレクトリ中にファイル名 **index.html** で右のファイルを作成します。

```
<html>
   <head>
      <meta charset="UTF-8">
      <title>ZUMO CONTROLLER</title>
                  <script type="text/javascript" src="/webiopi.js"></script>
                  <script type="text/javascript">
                           var serial;
                           webiopi().ready(init);
                           function init(){
                                    serial = new Serial("usb1");
                           }
                           function fwrd() {
                                    serial.write("F");
                           }
                           function left() {
                                    serial.write("L");
                           }
                           function stop() {
                                    serial.write("S");
                           }
                           function rigt() {
                                    serial.write("R");
                           }
                           function back() {
                                    serial.write("B");
                           }
                  </script>
                  <style type="text/css">
                           button {
                                    //display: inline;
                                    margin: 5px 5px 5px 5px;
                                    width: 200px;       height:          100px;
        border-radius: 10px;
                                    font-size: 24pt;  font-weight: bold;
                           }
                           body {
                                    background-color:#C0C0C0;
                           }
                           #fwrd{
                                    color: blue;
                                    background-color:#808080;
                           }
                           #left{
                                    color: yellow;
                                    background-color:#808080;
                           }
                           #stop{
                                    color: red;
                                    background-color:#FFFFFF;

                           }
                           #rigt{
                                    color: yellow;
                                    background-color:#808080;
                           }
                           #back{
                                    color: blue;
                                    background-color:#808080;
                           }
                           #on {
                                    color: red;
                                    background-color:#FFFFFF;
                           }
                           #off {
                                    color: black;
                                    background-color:#FFFFFF;
                           }
                  </style>
   </head>

   <body>
          <div align="center">
             </br>
             <p><h1>ZumoMotor Controller</h1></p>
             <button id="fwrd" type="button" onClick="fwrd()">FWRD</button>
             </br>
             <button id="left" type="button" onClick="left()">LEFT</button>
             <button id="stop" type="button" onClick="stop()">STOP</button>
             <button id="rigt" type="button" onClick="rigt()">RIGT</button>
             </br>
             <button id="back" type="button" onClick="back()">BACK</button>
             </br>
          </div>
   </body>
</html>
```

/home/pi/wifi-project/python のディレクトリ内に **script.py** という名の空ファイルを作成します。（本来はここに **python** プログラムを書き込みますが、今回は何もないので空ファイルということになります。）

11. /etc/webiopi/config の書換え

webiopi 起動時に参照される **config** ファイルの設定を書き換えます。

まず、バックアップを作ります。

```
$ sudo cp  /etc/webiopi/config /etc/webiopi/config.bak
```

つぎに内容に変更を加えます。

```
$ sudo nano /etc/webiopi/config
```

または **Desktop** 環境なら **leafpad** エディタを使って

```
$ sudo leafpad /etc/webiopi/config
```

変更する点は以下のとおりです。元の行には#をつけてコメントにし、そのコピーを変更します。

```
[SCRIPTS]
#myscript=/home/pi/myWebIOproject2/python/script.py
myscript=/home/pi/wifi-project/python/script.py    <--スクリプト（中身は空）
[HTTP]
#doc-root = /home/pi/webiopi/examples/scripts/macros
doc-root=/home/pi/wifi-project/html                <--index.html のあるフォルダ
# Use welcome-file to change the default "Welcome" file
welcome-file = index.html                          <--始めに表示する html ファイル
[DEVICES]
#usb1 = Serial device:ttyACM0 baudrate:9600
usb1 = Serial device:ttyACM0 baudrate:9600         <--使用する USB
```

以上変更後上書きをします。

12. webiopi を実行して、ブラウザから RaspiOnZumo をコントロールする。

1）Debug 実行　コマンドラインから起動します

```
$ sudo webiopi -d -c /etc/webiopi/config
```

Debug 実行の停止

キーボード **Ctrl-C** を押す。

2）バックグラウンドで実行します。

```
$ sudo systemctl start webiopi
```

バックグラウンドでの実行停止

```
$ sudo systemctl stop webiopi
```

3）Raspi 起動時に自動実行させます。

```
$ sudo systemctl enable webiopi
```

起動時実行をやめる

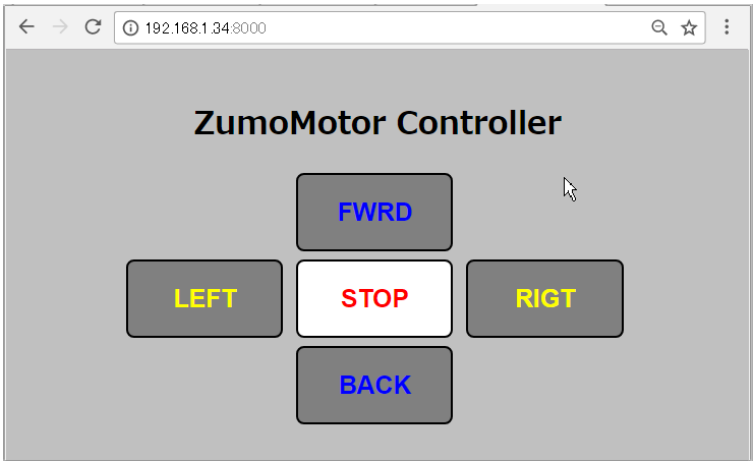
```
$ sudo systemctl disable webiopi
```

WebIOPi が起動中か確認する

```
$ ps ax | grep webiopi
```

上記１）で実行して問題がなければスマホや PC のブラウザから ４．と同様に

http://192.168.xx.xx:8000 にアクセスする。認証を求められたら**ユーザー：webiopi**
パスワード：raspberry で答えると、右に示すコントローラーページが現れます。



13. シャットダウンスイッチ

ディスプレイ、キーボードが接続できない環境で、自動起動にするには１２の２）を使って

```
$ sudo systemctl enable webiopi
```

としておき、シャットダウンは **/home/pi** に以下の **python** ファイル **shutdown.py** として保存します。これは **GPIO26** 番と **GND** 間にプッシュスイッチを差し、**python** にスイッチ押下を検出させて、検出したらシェルのシャットダウンを呼びます。

```
#!/usr/bin/python
# coding:utf-8
# shutdown.py
import time
import serial
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
#GPIO 26pin を入力モードとし、pull up 設定とします
GPIO.setup(26,GPIO.IN,pull_up_down=GPIO.PUD_UP)
GPIO.wait_for_edge(26, GPIO.FALLING)
ser.write("S")
GPIO.cleanup()
os.system("sudo shutdown -h now")
```

この python プログラムを RasPi 起動時に起動しておくには

/etc/rc.local

を開いて、リスト最後のほうの

exit 0 の

前に以下のスクリプトを追加します。これでプッシュスイッチでシャットダウンができるようになります。

```
$ sudo nano /etc/rc.local
```

```
. . . . .
python /home/pi/shutdown.py  #追加
exit 0
```

以上

TechShare 森