```
/*
  WiFi Web Server

 A simple web server that shows the value of the analog input pins.
 using a WiFi shield.

 This example is written for a network using WPA encryption. For
 WEP or WPA, change the WiFi.begin() call accordingly.

 Circuit:
 * WiFi shield attached
 * Analog inputs attached to pins A0 through A5 (optional)

 created 13 July 2010
 by dlf (Metodo2 srl)
 modified 31 May 2012
 by Tom Igoe

 */

#include <SPI.h>
#include <WiFi101.h>
#include <driver/source/nmasic.h>

#define TIMEOUT (0x2000ul)

#include "arduino_secrets.h"
///////please enter your sensitive data in the Secret tab/arduino_secrets.h
char ssid[] = "XXXXXXXXXX";    // your network SSID (name)
char pass[] = "XXXXXXXXXX";    // your network password (use for WPA, or use as
key for WEP)
int keyIndex = 0;              // your network key Index number (needed only for
WEP)
IPAddress ip(192, 168, XX, XX);

int status = WL_IDLE_STATUS;
int led = 2;
int relay = 6;
int count=0;
String command="";
String line="";

WiFiServer server(80);

void setup() {
     pinMode (6, OUTPUT);
//     digitalWrite (6, HIGH);
     pinMode (led, OUTPUT);
     digitalWrite (led, HIGH);
      WiFi.setPins (8,7,4);
//         pinMode(led, OUTPUT);
  //Initialize serial and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }

  // check for the presence of the shield:
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue:
    while (true);
  }
```

```
  WiFi.config(ip);

  // attempt to connect to WiFi network:
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    // Connect to WPA/WPA2 network. Change this line if using open or WEP
network:
    status = WiFi.begin(ssid, pass);

    // wait 10 seconds for connection:
    delay(10000);
  }
  server.begin();
  // you're connected now, so print out the status:
  printWiFiStatus();
}


void loop() {
  // listen for incoming clients
  WiFiClient client = server.available();
  if (client) {
//    digitalWrite(led, HIGH);
    Serial.println("new client");
    // an http request ends with a blank line
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        if (c == '\n')
            line +="\0";
        else
            line += c;
//        Serial.write(c);
        // if you've gotten to the end of the line (received a newline
        // character) and the line is blank, the http request has ended,
        // so you can send a reply
        if (c == '\n' && currentLineIsBlank) {
          // Process the command

          if (command=="temp1") {
              Serial.println("received temp1 command");
              client.println("HTTP/1.1 200 OK");
              client.println("Content-Type: text/html");
              client.println("Connection: close");  // the connection will be
closed after completion of the response
              client.println("Refresh: 10");  // refresh the page automatically
every 5 sec
              client.println();
              client.println("<!DOCTYPE HTML>");
              client.println("<html>");
              // output the value of each analog input pin
              int sensorReading = analogRead(0);
              client.print("analog input 0 (temp1) is ");
              client.print(sensorReading);
              client.println("<br />");
              client.print("Count :");
              client.println(count);
              client.println("</html>");
              count++;
              break;
          }
          else if (command=="temp2") {
```

```
                Serial.println("received temp1 command");
                client.println("HTTP/1.1 200 OK");
                client.println("Content-Type: text/html");
                client.println("Connection: close");  // the connection will be
closed after completion of the response
                client.println("Refresh: 10");  // refresh the page automatically
every 5 sec
                client.println();
                client.println("<!DOCTYPE HTML>");
                client.println("<html>");
                // output the value of each analog input pin
                int sensorReading = analogRead(2);
                client.print("analog input 2 (temp2) is ");
                client.print(sensorReading);
                client.println("<br />");
                client.print("Count :");
                client.println(count);
                client.println("</html>");
                count++;
                break;
            }
             else if (command=="alltemp") {
                Serial.println("received alltemp command");
                client.println("HTTP/1.1 200 OK");
                client.println("Content-Type: text/html");
                client.println("Connection: close");  // the connection will be
closed after completion of the response
                client.println("Refresh: 10");  // refresh the page automatically
every 5 sec
                client.println();
                client.println("<!DOCTYPE HTML>");
                client.println("<html>");
                // output the value of each analog input pin
                int sensorReading = analogRead(0);
                client.print("analog input 0 (temp1) is ");
                client.print(sensorReading);
                client.println("<br />");
                sensorReading = analogRead(2);
                client.print("analog input 2 (temp2) is ");
                client.print(sensorReading);
                client.println("<br />");
                client.print("Count :");
                client.println(count);
                client.println("</html>");
                count++;
                break;
            }         else if (command=="led-off") {
                Serial.println("received led-off command");
                digitalWrite (led, LOW);
            }
            else if (command=="led-on") {
                Serial.println("received led-on command");
                digitalWrite (led, HIGH);
            }
            else if (command=="relay-off") {
                Serial.println("received relay-off command");
                digitalWrite (relay, LOW);
            }
            else if (command=="relay-on") {
                Serial.println("received relay-on command");
                digitalWrite (relay, HIGH);
            }
            else if (command=="status") {
                Serial.println("received status command");
```

```
              int relayState = digitalRead(relay);
              int ledState = digitalRead(led);
              client.println("HTTP/1.1 200 OK");
              client.println("Content-Type: text/html");
              client.println("Connection: close");  // the connection will be
closed after completion of the response
              client.println();
              client.println("<!DOCTYPE HTML>");
              client.println("<html>");
              // output the value of each status
              client.print("led status is ");
              client.print(ledState);
              client.println("<br />");
              client.print("relay status is ");
              client.print(relayState);
              client.println("<br />");
              client.println("</html>");
              break;
          }
          else {
              Serial.println("Unknown command : " + command);
          }
          command="";
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");  // the connection will be closed
after completion of the response
          client.println("Refresh: 10");  // refresh the page automatically
every 5 sec
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
          // output the value of each analog input pin
 //         for (int analogChannel = 0; analogChannel < 4; analogChannel++) {
 //           int sensorReading = analogRead(analogChannel);
 //           client.print("analog input ");
 //           client.print(analogChannel);
 //           client.print(" is ");
 //           client.print(sensorReading);
 //           client.println("<br />");
 //         }
          client.print("Count :");
          client.println(count);
          count++;
          client.println("</html>");
          break;
        }
        if (c == '\n') {
          // you're starting a new line
          currentLineIsBlank = true;

//          Serial.println(line);

//          Serial.println("0:" + (String) line.charAt(0));
//          Serial.println("1:" + (String) line.charAt(1));
//          Serial.println("2:" + (String) line.charAt(2));
//          Serial.println("3:" + (String) line.charAt(3));
//          Serial.println("4:" + (String) line.charAt(4));
//          Serial.println("5:" + (String) line.charAt(5));

          if (line.charAt(0)=='G' &&
              line.charAt(1)=='E' &&
              line.charAt(2)=='T' &&
```

```
              line.charAt(3)==' ' &&
              line.charAt(4)=='/') {
                Serial.println("Found GET ");
                int i=5;
                command="";
                while (line.charAt(i) != 'H' && line.charAt(i) != ' ') {
                    command +=line.charAt(i);
                    i++;
                }
                Serial.println("Command is :" + command);
          }
          else if (line.charAt(1)=='G' &&
              line.charAt(2)=='E' &&
              line.charAt(3)=='T' &&
              line.charAt(4)==' ' &&
              line.charAt(5)=='/') {
                Serial.println("Found GET ");
                int i=6;
                command="";
                while (line.charAt(i) != 'H' && line.charAt(i) != ' ') {
                    command +=line.charAt(i);
                    i++;
                }
                Serial.println("Command is :" + command);
          }
//          else
//              Serial.println("Get not found");

          line="";
        }
        else if (c != '\r') {
          // you've gotten a character on the current line
          currentLineIsBlank = false;
        }
      }
    }
    // give the web browser time to receive the data
    delay(1);

    // close the connection:
    client.stop();
//    digitalWrite(led, LOW);

    Serial.println("client disconnected");
  }
}


void printWiFiStatus() {
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your WiFi shield's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
}
```