This code needs to be run using Ardunio which will upload the code to the ESP8266 Thing via FTDI Breakout.

#include <ESP8266WiFi.h>

//////////////////////

// WiFi Definitions //

//////////////////////

const char WiFiAPPSK[] = "sparkfun";

/////////////////////

// Pin Definitions //

/////////////////////

const int LED\_PIN = 5; // Thing's onboard, green LED

const int ANALOG\_PIN = A0; // The only analog pin on the Thing

const int DIGITAL\_PIN = 12; // Digital pin to be read

WiFiServer server(80);

void setup()

{

initHardware();

setupWiFi();

server.begin();

}

void loop()

{

// Check if a client has connected

WiFiClient client = server.available();

if (!client) {

return;

}

// Read the first line of the request

String req = client.readStringUntil('\r');

Serial.println(req);

client.flush();

// Match the request

int val = -1; // We'll use 'val' to keep track of both the

// request type (read/set) and value if set.

if (req.indexOf("/led/0") != -1)

val = 0; // Will write LED low

else if (req.indexOf("/led/1") != -1)

val = 1; // Will write LED high

else if (req.indexOf("/read") != -1)

val = -2; // Will print pin reads

// Otherwise request will be invalid. We'll say as much in HTML

// Set GPIO5 according to the request

if (val >= 0)

digitalWrite(LED\_PIN, val);

client.flush();

// Prepare the response. Start with the common header:

String s = "HTTP/1.1 200 OK\r\n";

s += "Content-Type: text/html\r\n\r\n";

s += "<!DOCTYPE HTML>\r\n<html>\r\n";

// If we're setting the LED, print out a message saying we did

if (val >= 0)

{

s += "LED is now ";

s += (val)?"on":"off";

}

else if (val == -2)

{ // If we're reading pins, print out those values:

s += "Analog Pin = ";

s += String(analogRead(ANALOG\_PIN));

s += "<br>"; // Go to the next line.

s += "Digital Pin 12 = ";

s += String(digitalRead(DIGITAL\_PIN));

}

else

{

s += "Invalid Request.<br> Try /led/1, /led/0, or /read.";

}

s += "</html>\n";

// Send the response to the client

client.print(s);

delay(1);

Serial.println("Client disonnected");

// The client will actually be disconnected

// when the function returns and 'client' object is detroyed

}

void setupWiFi()

{

WiFi.mode(WIFI\_AP);

// Do a little work to get a unique-ish name. Append the

// last two bytes of the MAC (HEX'd) to "Thing-":

uint8\_t mac[WL\_MAC\_ADDR\_LENGTH];

WiFi.softAPmacAddress(mac);

String macID = String(mac[WL\_MAC\_ADDR\_LENGTH - 2], HEX) +

String(mac[WL\_MAC\_ADDR\_LENGTH - 1], HEX);

macID.toUpperCase();

String AP\_NameString = "ESP8266 Thing " + macID;

char AP\_NameChar[AP\_NameString.length() + 1];

memset(AP\_NameChar, 0, AP\_NameString.length() + 1);

for (int i=0; i<AP\_NameString.length(); i++)

AP\_NameChar[i] = AP\_NameString.charAt(i);

WiFi.softAP(AP\_NameChar, WiFiAPPSK);

}

void initHardware()

{

Serial.begin(115200);

pinMode(DIGITAL\_PIN, INPUT\_PULLUP);

pinMode(LED\_PIN, OUTPUT);

digitalWrite(LED\_PIN, LOW);

// Don't need to set ANALOG\_PIN as input,

// that's all it can be.

}