ESP32 Web Server Tutorial

Welcome to this tutorial! In this tutorial, we will be teaching you how to connect an ESP32 to a WiFi network and then setup a simple web server.

First, lets install the WiFi Library:

`#include <Wifi.h>`

We are now going to tell our board which WiFi network to connect to, lets define two varibles and use `*` to declare a pointer. We are going to name these SSID and Password, and set them to our network credentials.

```
const char*  ssid = "#";
const char*  password = "#";
```

Now we are going to begin our WiFi connection and setup a serve on port 80. We will be declaring an object, `server` here.

```
WiFiServer Server(80);
```

Now in our `setup()` function, lets initialize the web server and all of its components.

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  WiFi.begin(ssid, password);
  server.begin();

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("...");
  }
```

```
  Serial.print("");
  Serial.println(WiFi.localIP());
}
```

Here, the server will try and begin by inputting our ssid and password and attempting to connect to our network. `Wifi.begin` attempts to connect to the WiFi network, taking `ssid` and `password` as parameters. It then attempts to start the web server.

Whilst connecting to the WiFi Network this while loop is running

```
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("...");
  }

  Serial.print("");
  Serial.println(WiFi.localIP());
```

As we are trying to connect, “…” will be printed in the Serial Monitor, once we are connected and the server is up, the IP adress of that server will be printed in the Serial monitor.

Lets now setup the HTTP client code, this will be in our `void loop()` function

```
void loop() {
 WiFiClient client = server.available();
  if (!client) {
    return;
  }
}
```

This code listens for a new client connection on a Wi-Fi server. When a client connects, the message "new client" is printed to a serial interface for debugging. The code then waits for data from the connected client.

Lastly, we need to send our client a response back, this can be in HTML. We will make a String called `htmlResponse` and send that back to the client. We will also have to send

`HTTP/1.1 200 OK"` and `Content-Type: text/html` to inform the client that this is in HTML and the request was recivied alright.

```
void loop() {
 WiFiClient client = server.available();
  if (!client) {
    return;
  }

  Serial.println("new client");
  while (!client.available()) {
    delay(1);
  }

//Start Request
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("");
  client.println("<!DOCTYPE html>");
  String htmlResponse =
    "<html>"
    "<body>"
    " <form>"
    "  <h1>Hello, World!</h1>"
    " </form>"
    "</body>"
    "<style>"
    "body { font-family: Comic Sans MS }"
    "</style>"
    "</html>";

  client.println(htmlResponse);
}
```

And there you go! You have connected an ESP32 to a network and hosted a web server!

Full Code

```
#include <WiFi.h>

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

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

WiFiServer server(80);

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  WiFi.begin(ssid, password);
  server.begin();

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("...");
  }

  Serial.print("");
  Serial.println(WiFi.localIP());
}

void loop() {
 WiFiClient client = server.available();
  if (!client) {
    return;
  }


//Start Request
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("");
  client.println("<!DOCTYPE html>");
  String htmlResponse =
    "<html>"
    "<body>"
    " <form>"
    "  <h1>Hello, World!</h1>"
    " </form>"
    "</body>"
    "<style>"
    "body { font-family: Comic Sans MS }"
    "</style>"
    "</html>";

  client.println(htmlResponse);
}
```