149 lines - 52 Removals

```
import os
import time
import ipaddress
import wifi
import socketpool
import board
import microcontroller
import terminalio
from digitalio import DigitalInOut, Direction
from adafruit_httpserver.server import HTTPServer
from adafruit_httpserver.request import HTTPRequest
from adafruit_httpserver.response import HTTPRespons
e
from adafruit_httpserver.methods import HTTPMethod
from adafruit_httpserver.mime_type import MIMEType

#  onboard LED setup
led = DigitalInOut(board.LED)
led.direction = Direction.OUTPUT
led.value = False

#  function to convert celcius to fahrenheit
def c_to_f(temp):
    temp_f = (temp * 9/5) + 32
    return temp_f

#  connect to network
print()
print("Connecting to WiFi")


#  connect to your SSID
wifi.radio.connect(os.getenv('CIRCUITPY_WIFI_SSID'),
os.getenv('CIRCUITPY_WIFI_PASSWORD'))

print("Connected to WiFi")
pool = socketpool.SocketPool(wifi.radio)
server = HTTPServer(pool, "/static")

# variables for HTML


unit = "F"


#  font for HTML
font_family = "monospace"

```

147 lines + 51 Additions

```
import os
import time
import ipaddress
import wifi
import socketpool
import board
import microcontroller
import terminalio
from digitalio import DigitalInOut, Direction
from adafruit_httpserver.server import HTTPServer
from adafruit_httpserver.request import HTTPRequest
from adafruit_httpserver.response import HTTPRespons
e
from adafruit_httpserver.methods import HTTPMethod
from adafruit_httpserver.mime_type import MIMEType

#  onboard LED setup
led = DigitalInOut(board.LED)
led.direction = Direction.OUTPUT
led.value = False

#  function to convert celcius to fahrenheit
def c_to_f(temp):
    temp_f = (temp * 9/5) + 32
    return temp_f

#  connect to network
print()
print("Connecting to WiFi")

#  set static IP address
ipv4 =  ipaddress.IPv4Address("192.168.1.42")
netmask =  ipaddress.IPv4Address("255.255.255.0")
gateway =  ipaddress.IPv4Address("192.168.1.1")
wifi.radio.set_ipv4_address(ipv4=ipv4,netmask=netmas
k,gateway=gateway)
#  connect to your SSID
wifi.radio.connect(os.getenv('CIRCUITPY_WIFI_SSID'),
os.getenv('CIRCUITPY_WIFI_PASSWORD'))

print("Connected to WiFi")
pool = socketpool.SocketPool(wifi.radio)
server = HTTPServer(pool, "/static")

# variables for HTML
temp_test = str(c_to_f(microcontroller.cpu.temperatu
re))
unit = "F"
i = 0
istr = str(i)
#  font for HTML
font_family = "monospace"

```

```
#  the HTML script
#  setup as an f string
#  this way, can insert string variables from code.py directly
#  of note, use {{ and }} if something from html *actually* needs to be in brackets
#  i.e. CSS style formatting
def webpage():
    temp = str(c_to_f(microcontroller.cpu.temperature))

    html = f"""
    <!DOCTYPE html>
    <html>
    <head>
    <meta http-equiv="Content-type" content="text/html;charset=utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.4/jquery.min.js"></script>
    <style>
    html{{font-family: {font_family}; background-color: lightgrey;
    display:inline-block; margin: 0px auto; text-align: center;}}
      h1{{color: deeppink; width: 200; word-wrap: break-word; padding: 2vh; font-size: 20px;}}
      p{{font-size: 15px; width: 200; word-wrap: break-word;}}
      .button{{font-family: {font_family};display: inline-block;
      background-color: black; border: none;
      border-radius: 4px; color: white; padding: 5px 5px;
      text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}}
      p.dotted {{margin: auto;
      width: 75%; font-size: 15px; text-align: center;}}
    </style>
    </head>
    <body>
    <title>Pico W HTTP Server</title>
    <h1>Pico W HTTP Server</h1>
    <br>
    <p class="dotted">This is a Pico W running an HTTP server with CircuitPython.</p>
    <br>
    <p class="dotted">The current ambient temperature near the Pico W is
    <span style="color: deeppink;"><span id="temp">{temp}</span>°{unit}</span></p><br>


    <h1>Control the LED on the Pico W with these buttons:</h1><br>
    <p><button class="button" name="LED" value="ON" type="submit">LED ON</button></a></p>
    <p><button class="button" name="LED" value="OFF" type="submit">LED OFF</button></a></p>
```

```
#  the HTML script
#  setup as an f string
#  this way, can insert string variables from code.py directly
#  of note, use {{ and }} if something from html *actually* needs to be in brackets
#  i.e. CSS style formatting
def webpage():



    html = f"""
    <!DOCTYPE html>
    <html>
    <head>
    <meta http-equiv="Content-type" content="text/html;charset=utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">


    <style>
    html{{font-family: {font_family}; background-color: lightgrey;
    display:inline-block; margin: 0px auto; text-align: center;}}
      h1{{color: deeppink; width: 200; word-wrap: break-word; padding: 2vh; font-size: 20px;}}
      p{{font-size: 15px; width: 200; word-wrap: break-word;}}
      .button{{font-family: {font_family};display: inline-block;
      background-color: black; border: none;
      border-radius: 4px; color: white; padding: 5px 5px;
      text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}}
      p.dotted {{margin: auto;
      width: 75%; font-size: 15px; text-align: center;}}
    </style>
    </head>
    <body>
    <title>Pico W HTTP Server</title>
    <h1>Pico W HTTP Server</h1>
    <br>
    <p class="dotted">This is a Pico W running an HTTP server with CircuitPython.</p>
    <br>
    <p class="dotted">The current ambient temperature near the Pico W is
    <span style="color: deeppink;">{temp_test}°{unit}</span></p><br>
    <span style="color: black;">i={istr}</span></p><br>
    <h1>Control the LED on the Pico W with these buttons:</h1><br>
    <form accept-charset="utf-8" method="POST">

    <button class="button" name="LED ON" value="ON" type="submit">LED ON</button></a></p></form>
```

```
    </body>
    <script>

    // LED buttons handler
    $(".button").click((el) => {{ $.post("/led", {{[el.target.name]: el.target.value}}) }});
    // 1 sec temperature refresher
    setInterval(() => {{
        $.get("/temp", (data) => {{
            $("#temp").text(data);
        }});
    }}, 1000);
    </script>
    </html>
    """
    return html

#  route default static IP
@server.route("/")
def base(request: HTTPRequest):
    #  serve the HTML f string
    #  with content type text/html
    with HTTPResponse(request, content_type=MIMEType.TYPE_HTML) as response:
        response.send(f"{webpage()}")

# change led state
@server.route("/led", method=HTTPMethod.POST)
def buttonpress(request: HTTPRequest):
    print("button request")
    #  get query params (doesn't handle all cases, use with caution
    query = {x[0] : x[1] for x in [x.split("=") for x in request.body.decode("utf8").split("&")]}

    #  if the led on button was pressed
    if "LED" in query:
        #  turn on or off the onboard LED
        led.value = (query["LED"] == "ON")
    # Acknowledge

    with HTTPResponse(request, content_type=MIMEType.TYPE_HTML) as response:
        response.send()

# get temperature
@server.route("/temp")
def temp(request: HTTPRequest):
```

```
    <p><form accept-charset="utf-8" method="POST">
    <button class="button" name="LED OFF" value="OFF" type="submit">LED OFF</button></a></p></form>
    </body></html>
    """
    return html

#  route default static IP
@server.route("/")
def base(request: HTTPRequest):
    #  serve the HTML f string
    #  with content type text/html
    with HTTPResponse(request, content_type=MIMEType.TYPE_HTML) as response:
        response.send(f"{webpage()}")

#  if a button is pressed on the site
@server.route("/", method=HTTPMethod.POST)
def buttonpress(request: HTTPRequest):
    #  get the raw text
    raw_text = request.raw_request.decode("utf8")

    print("raw text = ")

    print(raw_text)
    #  if the led on button was pressed
    if "ON" in raw_text:
        #  turn on the onboard LED
        led.value = True
    #  if the led off button was pressed
    if "OFF" in raw_text:
        #  turn the onboard LED off
        led.value = False
    #  reload site
    with HTTPResponse(request, content_type=MIMEType.TYPE_HTML) as response:
        response.send(f"{webpage()}")

print("starting server..")
```