Names ____

INTRODUCTION

The amazing Pico W also has a built-in radio module. This module can be used for both WiFi and Bluetooth connectivity. So you can build devices that don't need to be plugged into computers to take and save data.

WIFI FUN – *GETTING STARTED*, 2ND *ED.* CHAPTER 11

- Start reading and working through Ch. 12.
- For a network we will use:
  - SSID: GENEVAGAMING
  - PSK: GenevaGameConsoles
- For the WiF country use "US"

SERVING A WEB PAGE

We are going to do some things a little different. Instead of send a plain text "Hello!" to the web browser, we are going to send HTML. You may be familiar with HTML because it is the formatting language web pages are written in. So, the *response* the Pico W is going to send is in a string. Define this string before any loops in your program. The response is the HTML below:

```
response = """<!DOCTYPE html>
<html>
<head>
  <title>Pico W</title>
</head>
<body>
  <h1>Pico W</h1>
  <p>Hello World from Raspberry Pi Pico W!</p>
</body>
</html>
"""
```

- Remember triple quotes, `"""`, allow strings to span multiple lines.
- The `<!DOCTYPE html>` tells your browser that HTML formatted text is coming.
- The HTML document is sandwiched between `<html>` and `</html>`. In fact almost all HTML content is bracketed between tags with this style.
- The `head` is web page metadata that does not show up in the browser.
- The `title` of the web page is what shows up in your browser tab label above the web page.
- The main part of the web page is between the `body` tags.
- There will be a first level header, `h1`, at the top of the web page. This is usually large and bold.
- Next is a paragraph, `p`, with the welcome message.
- Finally, all of the tags are closed.

If you want to learn more about HTML, there are many resources on the web. I like `https://w3schools.com`.

Change the loop in the code to:

```
while True:
    try:
        client, address = s.accept()
        print("Connection accepted from ", address)
        client_file = client.makefile("rwb", 0)
        while True:
            line = client_file.readline()
            if not line or line == b"\r\n":
                break

        client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
        client.send(response)
        client.close()
        print("Response sent, connection closed.")
    except OSError as e:
        client.close()
        print("Error, connection closed")
```

The main difference is that instead of sending `Content-type: text/plain`, we are sending `text/html`, and our response is the HTML document above in the string `response`.

For sending the temperature, change the triple quoted string to an F-string by putting f in front of the string triple quote, and change the message to something like

```
The temperature of the CPU is {temperature:2f} C.
```

Since it is an f-string the value of the variable is printed as a float with two decimal places of precision.

Controlling an LED from a Web Page

Once the Pico W is set up to serve a web page, you can also use it to send dat from a browser to the Pico W. You can actively control what the Pico W is doing.

Work through the section *Controlling an LED* in Ch. 12.

Sending Data to the Cloud

Serving a web page is nice, but for some applications, you want the data stored permanently *on the cloud*, that is at some web site.

First we need a protocol for sending data to a server. One of the most widely used in the Internet Of Things (IoT) is MQTT. It is a publish-subscribe protocol. This means a device *publishes* its data, or feeds, on the internet, and, optionally, none, one, or many devices or cloud services *subscribe* to the feed. The MQTT protocol is supported by MicroPython, so we need a cloud service.

Sign Up to io.adafruit.com

Fortunately AdaFruit offers a free MQTT cloud service at `https://io.adafruit.com`. Navigate to this link and click *Get Started for Free*. Sign up for a free account. Click on *Feeds* in top navigation bar. On the next page in the top left click on the Key icon. Under *CircuitPython* select and copy your username and key. It should look something like this:

```
ADAFRUIT_AIO_USERNAME = "profhuster"
ADAFRUIT_AIO_KEY      = "aio_GTyf484q6RWzHSdW5vG7SPiQDo7i"
```

Start a new file in Thonny and paste this information in to save it for later.

Get MQTT MicroPython Library

A feature of Thonny we have not used yet is that Thonny can find and install MicroPython libraries. Plug in a Pico W, and click the Stop icon to get its attention and the python prompt.

Names ____

Go to **`https://raw.githubusercontent.com/pycom/pycom-libraries/master/examples/mqtt/mqtt.py`**. Click the *Download* icon and save it as **`mqtt.py`**. Move it to your working folder. In Thonny create a folder lib on the Pico W if you don't already have one. Upload **`mqtt.py`** to the lib folder.

- Get Adafruit.io account
- Create Feed. Go to feed, click Feed Info. Grab MQTT by Key URL and save it
- Back. Create Dashboard. Open dashboard. Click Settings => Create new Block => Line chart => Click on you feed and click Next Step
- Block Settings window. Show history to Live. Click Create Block
- Thonny => Tools => Manage packages. Search for umqtt simple. One choice should show up. Click on it (version 1.4.0) then click Install.
- Check it is installed. In the Pico W lib folder there should be a new folder umqtt and in that, a file simple.py
- Download mqtt_publish.py
- Edit SSID, and Password for your WiFi
- Lower in the code insert your MQTT username, password key, and your MQTT by Key.
-

Write the MQTT Publish Code