INTRODUCTION

Finally, we can break the boundaries of the four walls we are sitting in and connect to the wide world out there through the World Wide Web (WWW.) Once you can connect a Pico W to the internet you can read web pages, serve your own web pages, and store data "in the cloud." We are finally getting to the "W" in Pico W!

Start working through Ch. 12 *WiFi Connectivity with Pico W*. When you get **`Robust_connection.py`** working, pause and work through the section below.

SAVING WIFI INFORMATION IN A FILE

It is a good idea **not** to put sensitive information in your code. Instead save it in a separate file. This way you can safely share your code without sharing your sensitive information. There is a simple format for saving system settings called *TOML* (Tom's Obvious Minimal Language, if you were curious.) The code below reads simple TOML settings you need for WiFi.

The `settings.toml` *File*

Create a file named **`settings.toml`** like the one below:

```
WIFI_SSID = "GENEVAGAMING"
WIFI_PASSWORD = "GenevaGameConsoles"
```

This should work whereever you can access the Geneva gaming Wifi. If you want to connect to a different WiFi, make the abvious changes in the SSID and PASSWORD lines.

If you are are taking the Pico W home to a completely differents WiFi, I would make two different setting files named something like **`settings_geneva.toml`** and **`setting_home.toml`** and copy the one I want to **`settings.toml`** when I change locations.

Reading the `.toml` *File*

Since this is a routine task, it is a good idea to put the code that reads this into a module that you import. Create a file **`ReadTOML.py`** with the content below.

```
""" ReadTOML """

def ReadTOML(filename="settings.toml"):
    settings = {}
    with open(filename, 'r') as fp:
        line = fp.readline().strip()
        while line:
            if '=' in line:
                words = line.split('=')
                if len(words) == 2:
                    key = words[0].strip()
                    value = words[1].strip()[1:-1]
                    settings[key] = value

            line = fp.readline().strip()

    return settings
```

This code creates a *dictionary* of the settings. A dictionary (also called a *hash*) is a collection where the items are found by a *key* instead of an index number. The line

```
settings = {}
```

creates an empty dictionary.

Next the code opens the **`settings.toml`** file and reads all of the lines with a **`=`** in them. It assumes those lines are the name of the setting and its value (without the quotes.)

Using the `setting.toml` *File*

Copy the **`ReadTOML.py`** to the Pico W so your code can import it. After the last import line, add the line

```
from ReadTOML import ReadTOML
settings = ReadTOML()
```

Replace the code

```
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect("NetworkName", "Password")
```

with the code

```
ssid = settings["WIFI_SSID"]
psk = settings["WIFI_PASSWORD"]

wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, psk)
```

Great! Your code is now shareable without revealing your secrets.

Note: If you are connecting to a different WiFi, create a **`settings_loc.toml`** for that WiFi (replace **`loc`** with a location name,) and use the line

```
settings = ReadTOML("settings_loc.toml")
```

Use this in the rest of the code you write for this chapter.

What Does the Web Page Say?

For each of the URL's five URL, including the first one, describe what the web request returns. Check the results aby navigating to the URL using a browser.

- npr.org -
- wttr.in -
- ipecho.net -
- earthquake.usgs.gov -
- artscene.textfiles.com -

Continue

Keep working through the book. When you save the file connect.py, don't forget to use the **`settings.toml`** file. Later in the section *Hosting a Web Page* you create a file **`connect.py`**. The file connect.py will handle the TOML stuff and connect to the Wifi.

Posting Data to the Cloud

It is nice to have a web page where you can read a temperature or turn an LED on and off, but what we have done so far requires that you be on the same local WiFi as the Pico W. Another restriction is that the temperature data is lost. When you refresh the web page, it shows the current temperare, but that data is not saved.

What you work on next is not in the book. You will post (send) data to a cloud server where it is saved and plotted. With this our data has some permanace and access outside its local WiFi.

MQTT

First we need a protocol for sending data to a server. One of the most widely used in the Internet Of Things (IoT) protocols is *MQTT*. (It stands for Message Queueing Telemetry Transport.) 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 1) Micropython code so publish data, and 2) A cloud service.

Sign Up to io.adafruit.com

- Navigate to **`http:io.adafruit.com`**
- Click on **`Get Started for Free`**.
- Enter the information and make sure you record your user name and password. You should get greeted with:

Welcome! You have signed up successfully.

- Add optional bio
- Click **`[Save Profile].`**
- In the top navigation bar, click **`[IO]`**.
- In the upper right there is a key icon. Click it.
- A window should pop up on the web page titled **`YOUR ADAFRUIT IO KEY`**.
- Copy and paste the lines in the **`CircuitPython`** box. They should look something like this:

```
ADAFRUIT_AIO_USERNAME = "astrohuster"
ADAFRUIT_AIO_KEY      = "aio_TfqB88sgsanYJ69FWBWBbq9DDfYB"
```

- Open the **`settings.toml`** file and paste these lines in at the bottom.
- These lines are how your Pico W will login to your **`io.adafruit.com`** account.

Get the MQTT MicroPython Library

This introduces a Thonny feature that helps with MicriPython – installing modules/packagess.

- Your Pico W has to be plugged in and running in Thonny.
- In Thonny, click **`Tools`** → **`Manage packages…`**
- In the top entry box enter **`umqtt.simple`**, and click **`[Search micropython-lib and PyPI]`**.
- Under Search Results, click **`umqtt.simple`**.
- At the bottom center, click **`[Install]`**. This will take about a minute. When it is done **`umqtt.simple`** will appear in the window on the left.
- You can check by clicking on the folder **`lib`**. In the **`lib`** folder you should see
  - Folder **`umqtt`**
  - Folder **`umqtt.simple-1.4.0.dist-info`** (The numbers may be for a different version.)

- ▶ **mqtt.py**

- It is a good idea to upload the entire lib folder to your laptop. Right click on lib and select **Download to YOUR_WORKING_FOLDER**.
- Check the installation by typing **from umqtt.simple import MQTTClient** at the python prompt. It should import without errors.

Write the MQTT Publishing Code

First import the modules we will use. Note the import of **MQTTClient**. This also uses connect.py to read your **settings.toml** file.

```
from connect import wlan, settings
from time import sleep_ms
from binascii import hexlify
from random import randint
from umqtt.simple import MQTTClient
from machine import Pin, unique_id
```

The **hexlify** function helps to make a unique ID for your Pico W.

Next use **settings** to get your AIO username, key and the name of the cloud service. You also need to give a name for your data feed. Each data stream needs its own feed name. Finally each feed name need to have a *topic* of the structure in the last line below.

```
aio_username = settings['ADAFRUIT_AIO_USERNAME']
aio_key = settings['ADAFRUIT_AIO_KEY']
# This feed name will show up on your io.adafruit.com page Feeds
aio_feed_name = "test-1"

# Default MQTT server to connect to
SERVER = settings['ADAFRUIT_SERVER']
CLIENT_ID = hexlify(machine.unique_id())[-4:]
# The topic of the mqtt message
mqtt_topic = aio_username + "/feeds/" + aio_feed_name
```

Put the code that runs the Pico W in a function called **main**

```
def main(server=SERVER):
    c = MQTTClient(CLIENT_ID, SERVER, port=1883, \
        user=aio_username, password=aio_key, \
        keepalive=300, ssl=False)
    c.connect()
    print(f"Connected to {server}")
    try:
        while True:
            temp = str(randint(200, 300)/10)
            print(f"Publishing {temp}")
            c.publish(mqtt_topic, temp)
            sleep_ms(5000)
    except:
        print("Exception. Disconnecting")
        c.disconnect()

main()
```

The main part of the loop makes a random temperature between 20 and 30 C, and then publishes it to you **io.adafruit** account. The final line calls **main** to run your code.

Once you get the code running, return to your **`io.adafruit`** pages. Click **`Feeds`** in the top navigation bar.

You need to create a *Dashboard* to make a graph of your data.

- Click on **`Dashboards`** in the top nav bar.
- On the left click on **`[(+) New Dashboard]`**.
- Give it a name, then click **`[Create]`**.
- Click on the name of your new dashboard. Click on the settings gear on the right, and select **`Create New Block`**, then click on the graph (called Line Chart.)
- In the **`Connect Feeds`** popup, click on your feed name, then click **`[Next Step]`**.
- On **`Show History`** select **`7 days`**. Scroll down and click **`[Create Block]`**.
- Finally, click on the Settings icon, then **`Dashboard Privacy`** and confirm that you want to make your data public.
- Click on **`Dashboards`** in the top Nav bar, then the name of your new dashboard. You should be greeted with a graph of your data.
- You can also add a new block using stream and it will print out the data.
- You can use the settings icon to select **`Edit Layout`** and move the blacks around and resize them.

E-mail your instructors with the name of your dashboard, so they can share your joy (and data.)

ACTIVITY

Instead of a random number, wire up the TMP36 and post the room temperature to your io.adafruit account. Now, instead of a data file taking up room on your Pico W, the data is sent to the could and stored there. Click on your feed, and in the feed's page there is a button to **`[Download All Data]`**. Then you can analyze with jupyter notebooks.

E-mail your instructor the name of this new feed and dashboard.