Simple Weather App

Sep 2026

MicroPython

Back to Micro Python

Introduction

This project is a bit of an accident. We want to make a new clock, we made one around 10 years ago from a Raspberry Pi Zero and some 7 segment LEDs. That project can be found here - Raspberry Pi LED Clock and NTP Server.

Now, it has worked flawlessly for 10 years. But it is still a small computer with an OS, and you have to do some maintenance on the OS (upgrades etc) periodically. So the thinking was, a new clock using a Raspberry Pi Pico would be cool, newer hardware, no OS worries, better display (LCD rather than 7 segment) and include other things than the time like the weather, or a news feed etc.

The first thing we explored was how to get the weather from the internet, turns out this is pretty simple. We used something called Open Meto, which is:

- Open-Meteo is a free, open-source weather API that delivers localized weather forecasts and historical climate data without requiring an API key.

So basically a free weather service to pull information from using an API.

If you wish to make one of these, then great. Please check out the Project called 'Network Analyzer' and look at Phase 2 (which is the LCD we are using).

You will need:

- The same LCD.
- The Same Driver.
- The Test Code for the Driver

Check out this link: Pico with 2.4" Tenstar SPI LCD. This will ensure your hardware is good, before you start to worry about the software.

A Simple Test

If you have completed the hardware setup, and see some test text on your LCD, then we are good to start. The first test is to get some weather data from Open-Meteo. Now, this means connecting to the internet from your Pico, so please ensure you have done the following:

- Connect Pico to Wifi
- Find Pico MAC Address

This will ensure you have tested your Pico Wifi and you have your MAC address in case your router requires this.

Test Open-Meteo

From Thonny, save this code to your Pico as Meteo-test and run it.

In the code, don't forget to change the following:

Set your Wifi Details.

| download

```
# --- CONFIGURATION ---
WIFI_SSID = "YOUR_WIFI_SSID"
WIFI_PASS = "YOUR_WIFI_PASSWORD"
```

Set your Latitude and Longitude (you can leave it at London for a simple test if you want).

| download

```
# Coordinates (e.g., London: lat=51.5074, lon=-0.1278)
LATITUDE = "51.5074"
LONGITUDE = "-0.1278"
```

Save this code as 'Meteo-test.py' and run it.

| download

```
import network
import urequests
import time

# --- CONFIGURATION ---
WIFI_SSID = "YOUR_WIFI_SSID"
WIFI_PASS = "YOUR_WIFI_PASSWORD"

# Coordinates (e.g., London: lat=51.5074, lon=-0.1278)
LATITUDE = "51.5074"
LONGITUDE = "-0.1278"

# Open-Meteo API URL for temperature, humidity, and weather code
URL =
f"http://api.open-meteo.com/v1/forecast?latitude={LATITUDE}&longitude={LONGITUDE}&current_weather=
true"

def connect_wifi():
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    if not wlan.isconnected():
        print(f"Connecting to Wi-Fi network: {WIFI_SSID}...")
        wlan.connect(WIFI_SSID, WIFI_PASS)

        timeout = 10
```

```
        while not wlan.isconnected() and timeout > 0:
            print("Waiting for connection...")
            time.sleep(1)
            timeout -= 1

    if wlan.isconnected():
        print("Connected successfully!")
        print("Network status/IP:", wlan.ifconfig())
        return True
    else:
        print("Failed to connect to Wi-Fi.")
        return False

def fetch_weather():
    print("Fetching weather data...")
    try:
        response = urequests.get(URL)
        if response.status_code == 200:
            data = response.json()
            current = data.get("current_weather", {})

            temp = current.get("temperature")
            windspeed = current.get("windspeed")
            weathercode = current.get("weathercode")

            print("\n--- WEATHER DATA RECEIVED ---")
            print(f"Temperature: {temp} °C")
            print(f"Wind Speed:  {windspeed} km/h")
            print(f"Weather Code: {weathercode}")
            print("-----------------------------\n")
        else:
            print(f"Error: Server responded with status code {response.status_code}")

        response.close()
    except Exception as e:
        print("Error fetching data:", e)

# --- RUN TEST ---
if connect_wifi():
    fetch_weather()
```

At the bottom of your Thonny, you should see something like the following:

download

```
Connected successfully!
Network status/IP: ('10.194.1.168', '255.255.255.0', '10.194.1.1', '10.194.1.1')
Fetching weather data...

--- WEATHER DATA RECEIVED ---

Temperature: 20.8 °C
Wind Speed:  15.1 km/h
Weather Code: 2
-----------------------------
```

If this has worked, then fantastic, you are ready to use the full weather app code. Some information.

- Network Status is your Internal IP Range
- Temp and Speed is that in London (as denoted by the co-ordinates
- Weather Code 2 is a value for your weather, see below

So the weather code is Meteos 'code' for “Partly Cloudy” here are the Meteo weather codes:

download

```
Code RangeWeather ConditionSuggested Icon0Clear skySun1, 2, 3Mainly clear, partly cloudy,
overcastSun/Cloud or Cloud45, 48Fog and depositing rime fogFog / Low mist51–57Drizzle (light to
```

```
dense, freezing)Light Rain / Drops61–67Rain (slight to heavy, freezing)Rain / Cloud with
rain71–77Snow fall & snow grainsSnowflake80–82Rain showers (slight, moderate, violent)Heavy Rain /
Cloud with shower95, 96, 99Thunderstorm (with/without hail)Lightning / Storm
```

From:
https://wiki.alanwalker.uk/ - **WalkerWiki - wiki.alanwalker.uk**

Permanent link:
https://wiki.alanwalker.uk/doku.php?id=simple_weather_app&rev=1789051844

Last update: **2026/09/10 14:50**