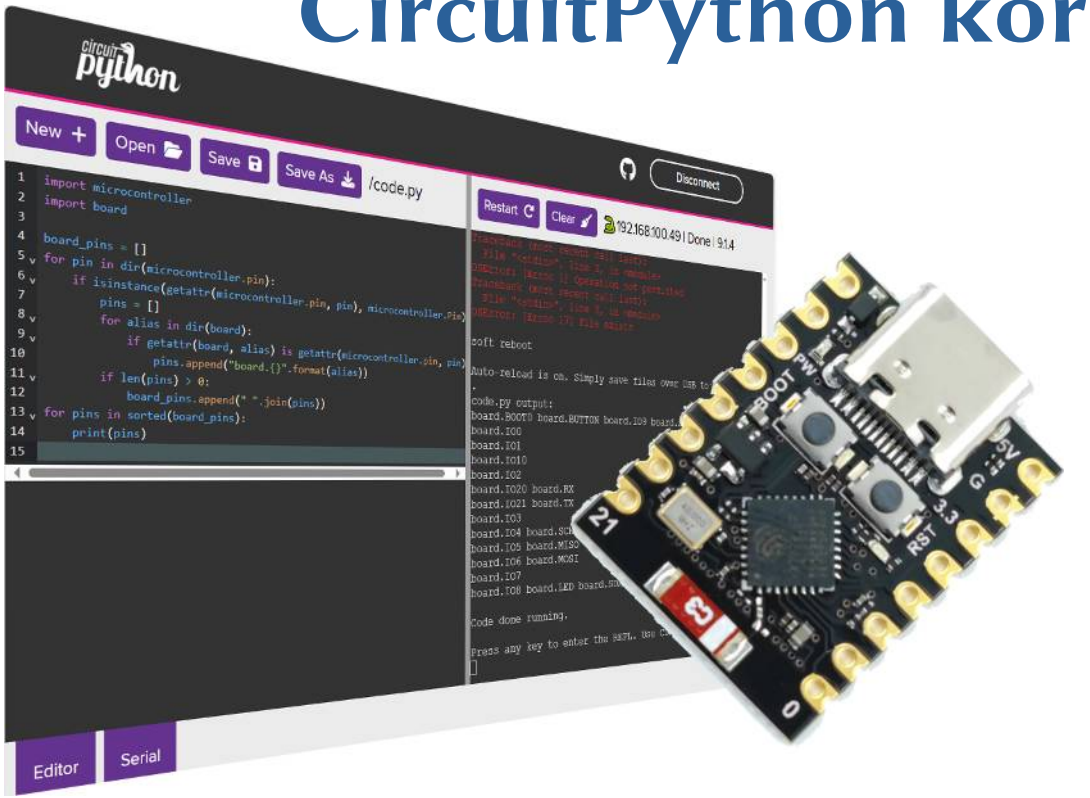
ESP32-C3 mikrovezérlők programozása CircuitPython környezetben



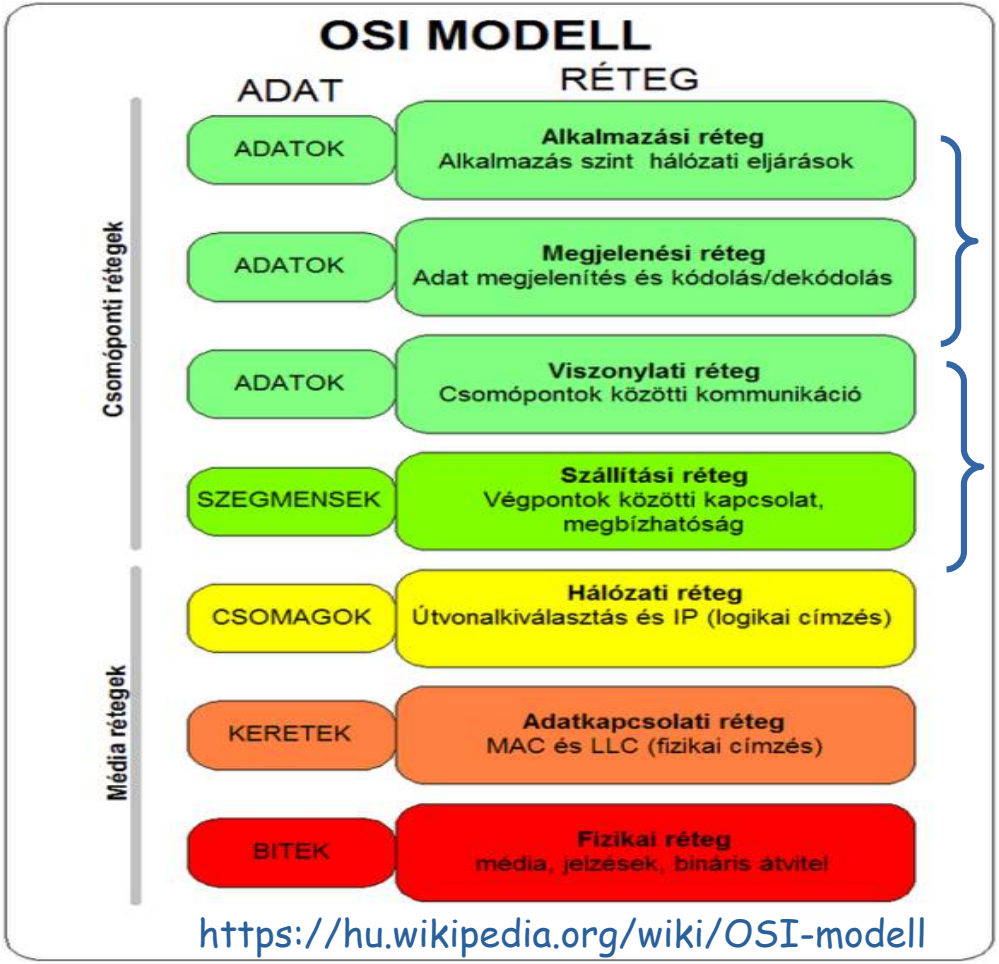
8. Webkliens alkalmazások

Felhasznált és ajánlott irodalom

- **Python:**
  - Mark Pilgrim/Kelemen Gábor: Ugorj fejest a Python 3-ba!
  - P. Wentworth et al. (ford. Biró Piroska, Szeghalmy Szilvia és Varga Imre): Hogyan gondolkozz úgy, mint egy informatikus: Tanulás Python 3 segítségével
- **CircuitPython:**
  - Adafruit: https://circuitpython.org/downloads
  - Learn Adafruit: Welcome to CircuitPython
  - Learn Adafruit: CircuitPython Essentials
  - Adafruit: Adafruit CircuitPython API Reference
  - Adafruit: github.com/adafruit/Adafruit CircuitPython Bundle

- **Online eszközök és támogatás:**
  - Learn Adafruit: CircuitPython on ESP32 Quick Start
  - Adafruit: Adafruit Web Serial ESPTool
  - Adafruit: CircuitPython Code Editor

Emlékeztető: Az OSI-modell (ISO 7498-1)



- Az **Open Systems Interconnection** (nyílt rendszerek összekapcsolása) referenciamodellje hét rétegbe szervezve írja le a hálózati kapcsolatot

HTTP, FTP, SMTP, Telnet, **NTP,** NFS

TCP, UDP

IP (**IPv4** – IPv6), ARP, IPSEC, ...

Ethernet, **WiFi,** PPP, ...

RS-232, V35, DSL, ISDN, 10BASE-T, 100BASE-TX stb.

Emlékeztető: A settings.toml állomány

- A „környezeti változók” – különösen a hálózati kapcsolódásokkal kapcsolatos adatokat a **settings.toml** állományban célszerű tárolni
- Egy tipikus **settings.toml** beállítás látható az alábbi ábrán:

```
CIRCUITPY_WIFI_SSID = "mySSID"
CIRCUITPY_WIFI_PASSWORD = "myPassword"
CIRCUITPY_WEB_API_PASSWORD = "mypass"
OPENWEATHERMAP_APPID = "66cbae54*******"
```

WiFi paraméterek

Web Browser jelszó

OpenWeatherMap API key

- A környezeti változók az **os.getenv("név")** metódussal vehetők elő:

```
import os, wifi

ssid = os.getenv("CIRCUITPY_WIFI_SSID")
password = os.getenv("CIRCUITPY_WIFI_PASSWORD")
wifi.radio.connect(ssid, password)
```

Emlékeztető: Csatlakozás Wi-Fi hálózathoz

- A kapcsolódáshoz szükséges a hálózat neve (SSID) és a jelszó, mintapéldánkban ezeket a **settings.toml** fájlból olvassuk ki
- A sikeres kapcsolódás után lekérdezhetjük az ESP32-C3 által beszerzett IP-címet

```
import wifi
import os
# Wi-Fi hálózat elérése környezeti változókból
ssid = os.getenv("CIRCUITPY_WIFI_SSID")
password = os.getenv("CIRCUITPY_WIFI_PASSWORD")

# Csatlakozás a Wi-Fi hálózathoz
print(f"Csatlakozás a {ssid} hálózathoz...")
wifi.radio.connect(ssid, password)
# Ha sikeres a csatlakozás, kiírja az IP címet
print("Csatlakozás sikeres!")
print("IP cím: ", wifi.radio.ipv4_address)
```

```
Csatlakozás a HUAWEI-2.4G-9bQR hálózathoz...
Csatlakozás sikeres!
IP cím:  192.168.100.33
```

HTTP kérések (HTTP requests)

- **HTTP** (HyperText Transfer Protocol) kliens és szerver közötti adatkommunikáció, ami a weblapok böngészésén kívül API-hívásokhoz és távoli vezérléshez is használható. **HTTPS** változata titkosított átvitelt biztosít
- Egy **HTTP kliens** kéréseket küld a **szervernek,** ezek formátuma:
  - A kérés sora CRLF-fel zárva (pl. `GET /index.html HTTP/1.1`)
  - Opcionális fejléc sorok, CRLF-fel zárva
  - Egy üres sor, amely a fejléc végét jelzi
  - Opcionális üzenet törzs, melyet szintén egy üres sor zár le
- **A kérés sorának formátuma:**

  `Method szóköz Request-URI szóköz HTTP-Version CRLF`

  (magyarázat: URI – Uniform Resource Identifier)
- **Method:**

  `GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE`
- **GET** – információ lekérése az adott szerver adott erőforrásából
- **POST** – adatokat küldünk a szervernek (pl. űrlapok kitöltésekor)

GET vs. POST

- A **GET** kérés is küldhet adatot (ezt a **`Request-URI`**-ban csatolva kell megadni) ami vagy a lekérés pontosításához ad kiegészítő információt, vagy – az eredeti koncepciótól eltérve – a szervernek küld elraktározandó adatot
- A **GET** kérésre egy példa:

```
GET /test/demo_form.php?name1=value1&name2=value2 HTTP/1.1
```

- A **POST** kérés az üzenet törzsében küldi az adatokat és többnyire adatküldésre használjuk (pl. űrlapok kitöltésénél), de a GET-hez hasonló lekérésre is használhatjuk
- A **POST** kérésre egy példa:

```
POST /test/demo_form.php HTTP/1.1
Host: w3schools.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length

name1=value1&name2=value2
```

Az Adafruit_requests programkönyvtár

- A **HTTP** és **HTTPS** kéréseket az **adafruit_requests** könyvtárral kezelhetjük, amely hasonlóan működik, mint a CPythonban elérhető **requests** modul
- **Használata:**

  1. Telepítsük a **/lib** mappába az **adafruit_requests.mpy** és az **adafruit_connection_manager.mpy** programkönyvtárakat
  2. A **wifi** hálózathoz történő kapcsolódás után két választásunk van:
     - A „beépített” **socketpool** programkönyvtár használata:

```
pool = socketpool.SocketPool(wifi.radio)
requests = adafruit_requests.Session(pool)
```

     - Az **adafruit_connection_manager** programkönyvtár használata:

```
pool = adafruit_connection_manager.get_radio_socketpool(wifi.radio)
requests = adafruit_requests.Session(pool)
```

  3. A **HTTP** (HyperText Transfer Protocol) lekérés kiküldése:

```
response = requests.get("http://wifitest.adafruit.com/testwifi/index.html")
print(response.text)
```

Egyszerű HTML lekérés

- Az alábbi program egy egyszerű HTML letöltést végez (itt most a beépített **socketpool** könyvtárat használtuk)

socketpool.py

```
import os
import wifi
import socketpool
import adafruit_requests

# WiFi kapcsolat beállítása
ssid = os.getenv("CIRCUITPY_WIFI_SSID")
password = os.getenv("CIRCUITPY_WIFI_PASSWORD")
wifi.radio.connect(ssid, password)
print("Kapcsolódva:", wifi.radio.ipv4_address)

pool = socketpool.SocketPool(wifi.radio)
requests = adafruit_requests.Session(pool)

# HTTP GET kérés végrehajtása
response = requests.get("http://wifitest.adafruit.com/testwifi/index.html")
print(response.text)
```

Futási eredmény:

```
This is a test of Adafruit WiFi!
If you can read this, its working :)
```

HTTPS lekérések

- A **HTTPS** lekérések **titkosított kapcsolaton** (**SSL** vagy **TLS**) keresztül folynak, ez biztosítja, hogy az adatok ne legyenek lehallgathatók vagy módosíthatók küldés közben
- **Tanúsítványok ellenőrzése:** a kliens eszköznek ellenőriznie kell, hogy a szerver rendelkezik-e érvényes SSL tanúsítvánnyal
- Fentiekhez létre kell hozni és az **adafruit_requests.Session** konstruktorának át kell adni egy **ssl_context** objektumot, amelyet kétféle módon is létrehozhatunk:
  - A „beépített” **socketpool** és **ssl** programkönyvtárak használatával:

```
pool = socketpool.SocketPool(wifi.radio)
ssl_context = ssl.create_default_context()
requests = adafruit_requests.Session(pool, ssl_context)
```

  - Az **adafruit_connection_manager** programkönyvtár használatával:

```
pool = adafruit_connection_manager.get_radio_socketpool(wifi.radio)
ssl_context = adafruit_connection_manager.get_radio_ssl_context(radio)
requests = adafruit_requests.Session(pool, ssl_context)
```

- Fentiek után a **HTTPS** lekérések ugyanúgy végezhetők, mint a „sima” HTTP lekérések

Egyszerű HTTPS lekérés

- Az alábbi példában a **HTTPS** lekérések egy-egy véletlenszerűen kiválasztott idézetet adnak vissza **JSON** formátumban, ami akár **Python dictionary**-nak is tekinthető (kulcs : adat párok)

citations_json.**py**

```
import os
import wifi
import adafruit_connection_manager
import adafruit_requests
ssid = os.getenv("CIRCUITPY_WIFI_SSID")
password = os.getenv("CIRCUITPY_WIFI_PASSWORD")
wifi.radio.connect(ssid, password)

pool = adafruit_connection_manager.get_radio_socketpool(wifi.radio)
ssl_context = adafruit_connection_manager.get_radio_ssl_context(wifi.radio)
requests = adafruit_requests.Session(pool, ssl_context)

QUOTE_API_URL = "https://www.adafruit.com/api/quotes.php"
response = requests.get(QUOTE_API_URL)
quote_data = response.json()[0]
print(f"{quote_data['text']} - {quote_data['author']}")
```

A JSON adatcsere-formátum

- **JSON** (JavaScript Object Notation) szöveg alapú, adatcseréhez való formátum ami C, C++, C#, JavaScript, Python és más nyelveken könnyen kezelhető (https://www.json.org/)
- A **JSON** formátum két struktúrára épül:
  - **Név – érték párok** kollekciója, ami Pythonban szótár, máshol struktúra, vagy asszociatív tömb formájában realizálható
  - **Értékek listája** ami pl. tömbként realizálható
- Példa: `[{'text': 'Adversity is revealing of character', 'author': 'Unknown'}]`
- Az előző oldalon bemutatott programban ezt így formáztuk olvashatóbbá:

```
QUOTE_API_URL = "https://www.adafruit.com/api/quotes.php"
response = requests.get(QUOTE_API_URL)
quote_data = response.json()[0]
print(f"{quote_data['text']} - {quote_data['author']}")
```

- Az eredmény:

```
Adversity is revealing of character - Unknown
```

Idézetek letöltése OLED kijelzőre

- Az előző programmal letöltött idézeteket egy **SSD1306 OLED** kijelzőn is megjeleníthetjük, ám ehhez két problémával kell megküzdenünk:
- A standard **font5x8.bin** használatakor egy karakter befoglaló téglalapja 6x8 képpont, így a 128x64 felbontású kijelzőn 8 sor, soronként 21 karakterrel jeleníthető meg, a 128x32 felbontású kijelzőn pedig csak 4 sor, soronként 21 karakterrel
- **Sortördelés:** a 21 karakter nyilvánvalóan kevés egy idézet megjelenítéséhez, ezért a sorokat tördelni kell. A sortördelés elve: a szöveget szavakra bontjuk, s ha egy sorba a következő szó már nem fér bele, akkor új sort kezdünk
- **A szöveg görgetése:** Ha az idézet hosszú, s nem fér ki a 4 vagy 8 soros kijelzőre, akkor a sorokat bizonyos időközönként eggyel feljebb toljuk, és a képernyő alján a következő sort beszúrjuk. Ezt addig folytatjuk, amíg a szöveg végére nem értünk

word_wrap.py

- Ez kivételesen egy **CPython** program, vagyis számítógépen is futtatható és a sortördelést mutatja be egy példán keresztül

Wisdom consists not so much in knowing what to do in the ultimate as knowing what to do next. - Herbert Hoover

```
def wrap_text(text, max_chars):
    words = text.split()  # Szavak listába bontása
    lines = []
    current_line = ""
    for word in words:
        if len(current_line) + len(word) + 1 <= max_chars:
            current_line += (" " if current_line else "") + word
        else:
            lines.append(current_line)
            current_line = word
    if current_line:
        lines.append(current_line)
    return lines

quote = "Wisdom consists not so much in knowing what to do in the
ultimate as knowing what to do next. - Herbert Hoover"
wrapped_text = wrap_text(quote, 21)
for line in wrapped_text:
    print(line)
```

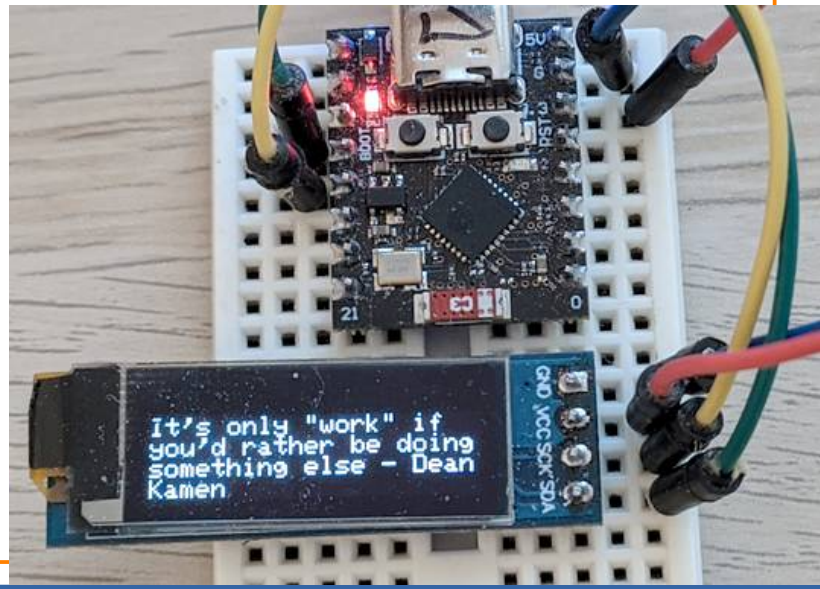
citations_oled.py - 2/1.

```
import board
import adafruit_ssd1306
import adafruit_requests
import adafruit_connection_manager
import wifi
import os
import time
ssid = os.getenv("CIRCUITPY_WIFI_SSID")
password = os.getenv("CIRCUITPY_WIFI_PASSWORD")
wifi.radio.connect(ssid, password)
i2c = board.I2C()  # Alapértelmezett I2C lábak
oled = adafruit_ssd1306.SSD1306_I2C(128, 32, i2c) # Itt a 128x32 kijelzőt használjuk
oled.rotation = 2
pool = adafruit_connection_manager.get_radio_socketpool(wifi.radio)
ssl_context = adafruit_connection_manager.get_radio_ssl_context(wifi.radio)
requests = adafruit_requests.Session(pool, ssl_context)
response = requests.get("https://www.adafruit.com/api/quotes.php")
quote_data = response.json()[0]
quote = f"{quote_data['text']} - {quote_data['author']}"
wrapped_text = wrap_text(quote, 21)
```

Az itt nem részletezett **wrap_text()** függvény ugyanaz, mint ami az előző oldalon látható

citations_oled.py - 2/2.

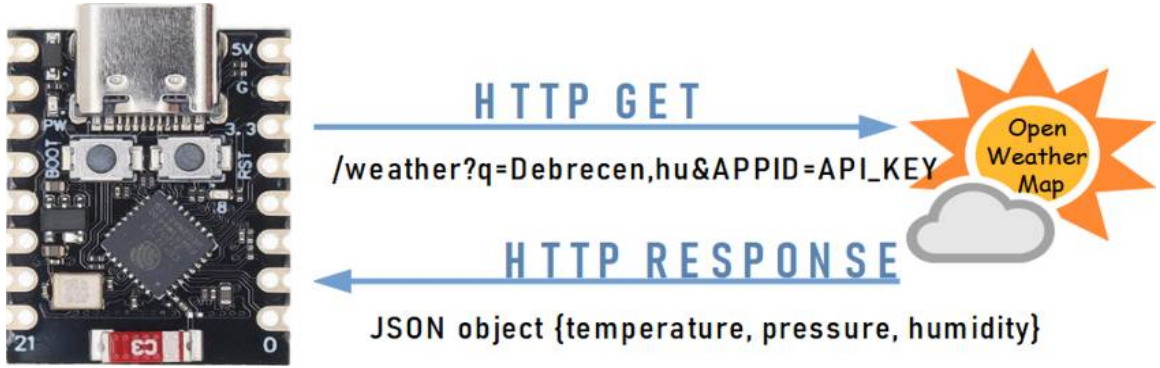
```
# Ha több mint 4 sor van, akkor görgetni kell (a 128x32 kijelzőn 4 sor fér ki)
if len(wrapped_text) > 4:
    start_index = 0
    while True:  # Végtelen ciklus a folyamatos görgetéshez
        oled.fill(0)  # Kijelző törlése
        for i in range(4):  # Mindig 4 sort jelenítünk meg
            if start_index + i < len(wrapped_text):
                oled.text(wrapped_text[start_index + i], 0, i * 8, 1)
        oled.show()  # Frissítés
        time.sleep(5)  # Várunk a következő léptetésig
        start_index += 1  # Lépünk egyet előre
        # Ha elértük az utolsó sort, újrakezdjük
        if start_index + 3 >= len(wrapped_text):
            start_index = 0
else:
    # Ha max. 4 sor van, csak egyszer kiíratjuk
    oled.fill(0)
    for i, line in enumerate(wrapped_text):
        oled.text(line, 0, i * 8, 1)
    oled.show()
```



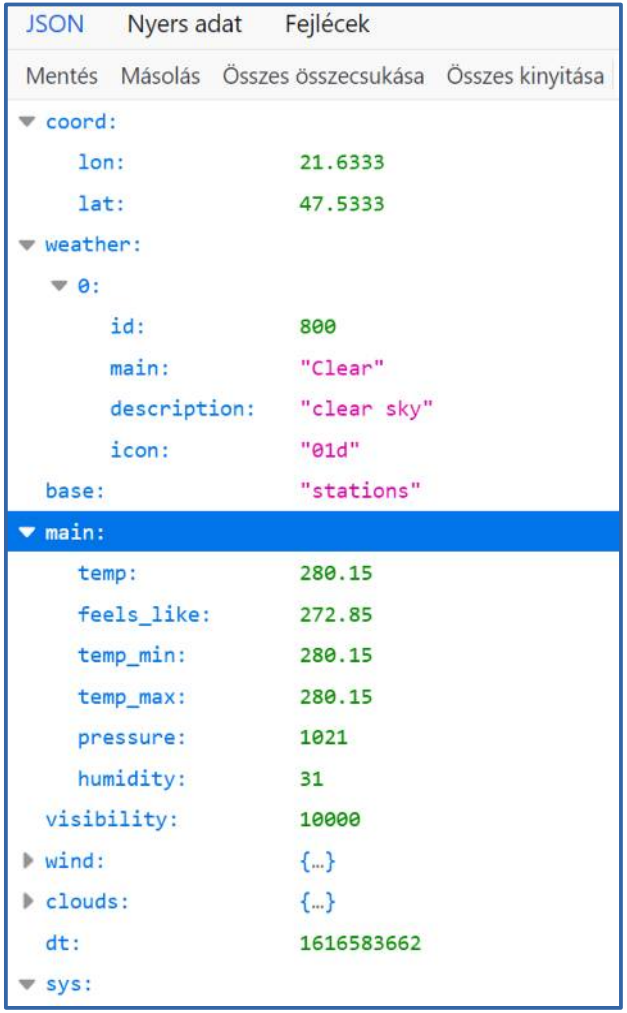
Időjárási adatok lekérése (OpenWeatherMap.org)

- Az aktuális időjárási adatokat és előrejelzéseket az openweathermap.org webszerverről tölthetjük le, HTTP kliensként
- Regisztráció (Sign in/Create Account) után az ingyenes szolgáltatásokat vehetjük igénybe (lásd Pricing, Free oszlop)
- Első bejelentkezéskor **szerezzük meg és jegyezzük fel az API kulcsot!**
- Mi most csak az aktuális adatok lekérdezésével foglalkozunk, például:
  `http://api.openweathermap.org/data/2.5/weather?q=Debrecen&appid=********`
- A választ alapértelmezetten JSON formátumban kapjuk meg, pl.:

```
{"coord":{"lon":21.6333,"lat":47.5333},"weather":[{"id":802,"main":"Clouds","description":"scattered
clouds","icon":"03d"}],"base":"stations","main":{"temp":283.15,"feels_like":279.93,"temp_min":283.15,
"temp_max":283.15,"pressure":1025,"humidity":46} . . .
```



Időjárási adatok kezelése és szűrése



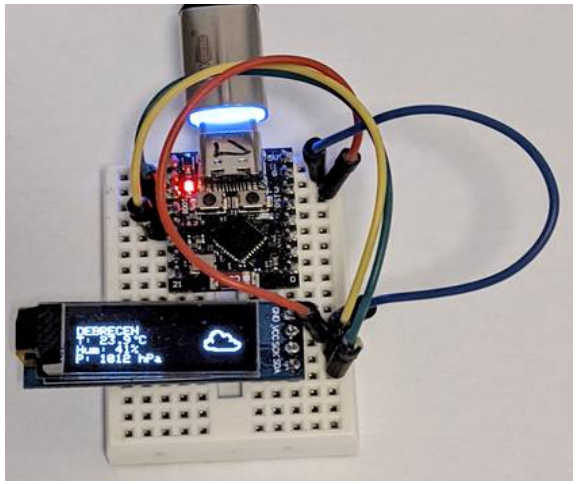
openweathermap.py

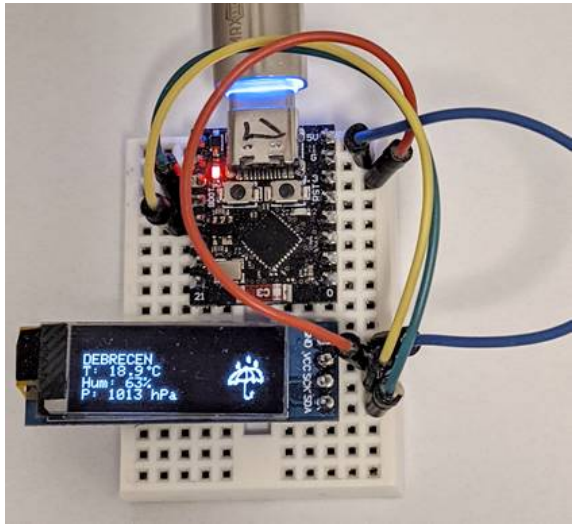
```
import os, time, ssl, wifi, socketpool, adafruit_requests
wifi.radio.connect(
    os.getenv("CIRCUITPY_WIFI_SSID"),
    os.getenv("CIRCUITPY_WIFI_PASSWORD")
)

location = "Debrecen, HU"
url = (
    "https://api.openweathermap.org/data/2.5/weather?q=" + location
    + "&appid=" + os.getenv("OPENWEATHERMAP_APPID")
)

pool = socketpool.SocketPool(wifi.radio)
requests = adafruit_requests.Session(pool, ssl.create_default_context())

while True:
    response = requests.get(url)
    data = response.json()
    print(f"\nWeather in {data['name']}: {data['weather'][0]['main']}")
    temp = f"{data['main']['temp'] - 273.15:.1f}°C"
    humidity = f"{data['main']['humidity']}% humidity"
    pressure = f"{data['main']['pressure']} hPa"
    print(temp, humidity, pressure)
    time.sleep(300)
```

```
Weather in Debrecen: Clear
23.9°C 31% humidity 1020 hPa
```

Készítsünk időjárásjelző alkalmazást!

- Használjuk fel az előző előadásokban bemutatott OLED kijelzőt az időjárási adatok kijelzésére (hőmérséklet, páratartalom, légnyomás)!
- Fokozzuk az élvezetet: a letöltött adatokból a **weather** szekcióból használjuk fel az ***'id'*** elemet és jelenítsünk meg a időjárást jellemző 32x32 felbontású ikonokat
  - Az időjárás kódok táblázata a Weather Conditions oldalon található. Például id = 800 jelentése *'tiszta idő'*
  - A programban felhasznált ikonokat az IconArchive adatbázisból töltöttük le, és a Marlin bitmap konverter segítségével konvertáltuk
- Szorgalmi feladat: Ha 128x64 méretű kijelzőnk van, akkor annak alsó felére az előző előadásban szereplő NTP órát is odavarázsolhatjuk





weather_oled_icon.py 3/1.

```
import time
import os
import wifi
import socketpool
import board
import adafruit_ssd1306
import adafruit_requests

# WiFi kapcsolódás
wifi.radio.connect(os.getenv('CIRCUITPY_WIFI_SSID'), os.getenv('CIRCUITPY_WIFI_PASSWORD'))

# OpenWeatherMap API hívás
location = "Debrecen, HU"
url = f"http://api.openweathermap.org/data/2.5/weather?q={location}
&appid={os.getenv('OPENWEATHERMAP_APPID')}"

pool = socketpool.SocketPool(wifi.radio)
requests = adafruit_requests.Session(pool)

# OLED kijelző inicializálása
i2c = board.I2C()  # Alapértelmezett I2C lábak
oled = adafruit_ssd1306.SSD1306_I2C(128, 32, i2c)
oled.rotation = 2
```

weather_oled_icon.py 3/2.

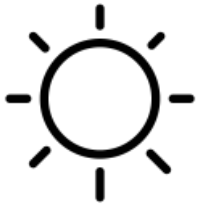
```
# Függvény az ikon kiválasztására
def select_icon(weather_id):
    if weather_id == 800:
        return bitmap_sun
    elif 200 <= weather_id < 300:
        return bitmap_lightning
    elif 800 < weather_id < 900:
        return bitmap_cloud
    else:
        return bitmap_umbrella

def draw_bitmap(x, y, bitmap):
    width = 32
    height = 32
    byte_width = width // 8  # Minden sorban 4 bájt van

    for j in range(height):
        for i in range(width):
            byte_index = j * byte_width + (i // 8)
            bit = (bitmap[byte_index] >> (7 - (i % 8))) & 0x01
            if bit:
                oled.pixel(x + i, y + j, 1)
    oled.show()
```

id	Ikon	Időjárás
800	bitmap_sun	napos idő
200 <= id < 300	bitmap_lightning	zivatar
800 < id < 900	bitmup_cloud	felhős idő
egyéb	bitmap_umbrella	eső

weather_oled_icon.py 3/3.

```
while True:
    response = requests.get(url).json()
    place = response["name"]                                # Időjárási adatok beolvasása
    weather_id = response["weather"][0]["id"]
    temperature = response["main"]["temp"] - 273.15         # Kelvin -> Celsius
    pressure = response["main"]["pressure"]
    humidity = response["main"]["humidity"]
    temp_text = f"T: {temperature:.1f} C"                   # Hőmérséklet tárolása szöveges változóban
    x_pos = len(temp_text) * 6 - 8                          # Fokjel koordináták kiszámítása
    oled.fill(0)                                            # display buffer törlése
    oled.text(place.upper(), 0, 0, 1)                       # Sor 1 (0px)  location (DEBRECEN)
    oled.text(temp_text, 0, 8, 1)                           # Sor 2 (8px)  temperature data
    oled.pixel(x_pos - 2, 9, 1)
    oled.pixel(x_pos - 1, 8, 1)                             # Fokjel kirajzolása
    oled.pixel(x_pos - 1,10, 1)
    oled.pixel(x_pos, 9, 1)
    oled.text(f"Hum: {humidity}%", 0, 16, 1)                # Sor 3 (16px) humidity data
    oled.text(f"P: {pressure} hPa", 0, 24, 1)               # Sor 4 (24px) pressure data
    selected_icon = select_icon(weather_id)                 # Az időjárási ID alapján ikon kiválasztása
    draw_bitmap(96, 0, selected_icon)                       # Ikon kirajzolása a jobb szélen
    oled.show()                                             # Refresh display screen
    time.sleep(300)                                         # Wait for 5 minutes
```

```
bitmap_sun = [
    0x00,0x00,0x00,0x00, 0x00,0x01,0x80,0x00,
    0x00,0x01,0x80,0x00, 0x00,0x01,0x80,0x00,
    0x00,0x01,0x80,0x00, 0x06,0x00,0x00,0xC0,
    0x07,0x0F,0xF1,0xC0, 0x03,0x1F,0xF9,0x80,
    0x00,0x70,0x0E,0x00, 0x00,0xE0,0x07,0x00,
    0x00,0xC0,0x03,0x00, 0x01,0x80,0x01,0x80,
    0x01,0x80,0x01,0x80, 0x01,0x00,0x00,0x80,
    0x3B,0x00,0x00,0xDC, 0x7B,0x00,0x00,0xDE,
    0x03,0x00,0x00,0xC0, 0x01,0x80,0x01,0x80,
    0x01,0x80,0x01,0x80, 0x01,0x80,0x01,0x80,
    0x00,0xC0,0x03,0x00, 0x00,0x60,0x06,0x00,
    0x00,0x38,0x1C,0x00, 0x03,0x9F,0xF9,0x80,
    0x07,0x07,0xE1,0xC0, 0x06,0x00,0x00,0xE0,
    0x00,0x01,0x80,0x60, 0x00,0x01,0x80,0x00,
    0x00,0x01,0x80,0x00, 0x00,0x01,0x80,0x00,
    0x00,0x01,0x80,0x00, 0x00,0x00,0x00,0x00
]
bitmap_cloud = [
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x3C,0x00,0x00, 0x00,0xFF,0x00,0x00,
    0x01,0xE3,0x00,0x00, 0x01,0xC1,0xF8,0x00,
    0x03,0x81,0xFC,0x00, 0x03,0x00,0xC6,0x00,
    0x03,0x00,0x06,0x00, 0x07,0x00,0x03,0x00,
    0x1F,0x80,0x03,0xC0, 0x3D,0x80,0x06,0xE0,
    0x38,0x80,0x00,0x78, 0x30,0x00,0x00,0x7C,
    0x30,0x00,0x00,0x04, 0x30,0x00,0x00,0x0C,
    0x38,0x00,0x00,0x0C, 0x1F,0xE0,0x00,0x38,
    0x0F,0xFF,0xFF,0xF0, 0x00,0x7F,0xFF,0xC0,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00
]
```



```
bitmap_umbrella = [
    0x00,0x00,0x00,0x00, 0x00,0x00,0x40,0x00,
    0x00,0x00,0xE0,0x00, 0x00,0x00,0xE0,0x00,
    0x00,0x00,0xE2,0x00, 0x00,0x20,0x67,0x00,
    0x00,0x30,0x07,0x00, 0x00,0x70,0x07,0x00,
    0x00,0x71,0x83,0x00, 0x00,0x73,0xFE,0x00,
    0x00,0x1F,0xFF,0x80, 0x00,0x3F,0xB1,0xC0,
    0x00,0xFC,0x98,0xE0, 0x01,0xD8,0x88,0x60,
    0x03,0xB0,0x8C,0x60, 0x07,0x20,0x8C,0x30,
    0x06,0x60,0xBD,0x30, 0x0C,0x7F,0xFF,0xF0,
    0x0F,0xFF,0xCE,0xF0, 0x0F,0xE1,0x84,0x20,
    0x0E,0xC1,0x00,0x00, 0x00,0x01,0x00,0x00,
    0x00,0x01,0x00,0x00, 0x00,0x01,0x00,0x00,
    0x00,0x01,0x00,0x00, 0x00,0x01,0x08,0x00,
    0x00,0x01,0x1C,0x00, 0x00,0x01,0x18,0x00,
    0x00,0x01,0x98,0x00, 0x00,0x01,0xF8,0x00,
    0x00,0x00,0xE0,0x00, 0x00,0x00,0x00,0x00
]
bitmap_lightning = [
    0x00,0x00,0x00,0x00, 0x00,0x00,0xE0,0x00,
    0x00,0x01,0xFF,0x00, 0x00,0x03,0xFF,0x00,
    0x00,0x03,0xC6,0x00, 0x00,0x07,0xEE,0x00,
    0x00,0x07,0x9C,0x00, 0x00,0x0F,0xB8,0x00,
    0x00,0x0E,0x70,0x00, 0x00,0x1F,0xF0,0x00,
    0x00,0x38,0xFF,0x00, 0x00,0x30,0x3F,0x80,
    0x00,0x70,0x07,0x00, 0x00,0x7F,0x8F,0x00,
    0x00,0x7F,0x8E,0x00, 0x00,0x73,0x9C,0x00,
    0x00,0x03,0x38,0x00, 0x00,0x06,0x30,0x00,
    0x00,0x06,0x70,0x00, 0x00,0x0C,0xE0,0x00,
    0x00,0x1C,0xC0,0x00, 0x00,0xD9,0xC0,0x00,
    0x01,0xF3,0x80,0x00, 0x01,0xF3,0xE0,0x00,
    0x00,0xC1,0xE0,0x00, 0x00,0xC3,0xC0,0x00,
    0x00,0xE7,0x80,0x00, 0x00,0xEE,0x00,0x00,
    0x00,0xFC,0x00,0x00, 0x00,0x70,0x00,0x00,
    0x00,0x60,0x00,0x00, 0x00,0x00,0x00,0x00
]
```

Az ESP32 C3 Super Mini kártya kivezetései