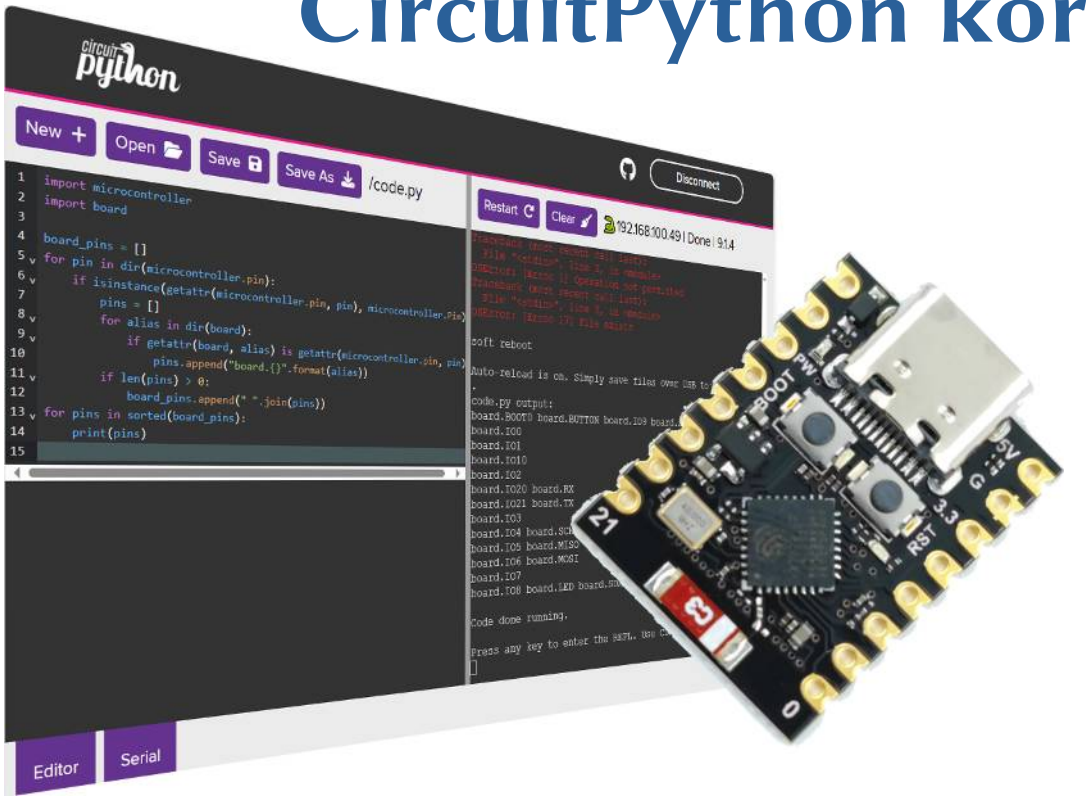
# ESP32-C3 mikrovezérlők programozása CircuitPython környezetben




## 6. E-papír kijelzők használata – 2. rész

# 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

# Felhasznált és ajánlott irodalom

- **Tananyagok, útmutatók:**
  - Carter Nelson: CircuitPython Display Support Using displayio
  - Tim C.: CircuitPython Display_Text Library
  - Ruiz Brothers: Custom Fonts for CircuitPython Displays
  - Anna Barela: Creating Slideshows in CircuitPython

- **CircuitPython kiegészítő programkönyvtárak:**
  - Adafruit_IL0373 – displayio driver for IL0373 2.9” E-paper displays
  - Adafruit_CircuitPython_Display_Text – text labels
  - Adafruit_CircuitPython_Bitmap_Font – custom font support
  - Adafruit_CircuitPython_ImageLoad – load image files
  - Adafruit_CircuitPython_Display_Shapes – lines, circles, triangles etc.

- **Adatlapok és dokumentáció:**
  - 2.9inch e-Paper Module (B) Manual

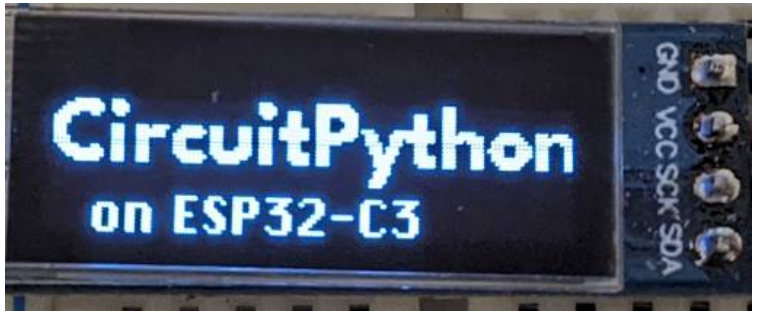
# Bitmap fontok használata

- A **displayio** kompatibilis kijelzőkön *(OLED, TFT, E-paper)* tetszés szerinti fontokat is használhatunk. Ezt a lehetőséget az alábbi példában mutatunk be, ahol a kirajzolandó szöveg egy **Text Label** elemként kezelhető
- Követelmények:
  - telepíteni kell az **adafruit_display_text** és az **adafruit_bitmap_font** könyvtárakat
  - a mikrovezérlő /**fonts** mappájába másoljuk be a **Chicago-12.bdf**, valamint a **LeagueSpartan-Bold-16.bdf** fontokat
- Természetesen telepíteni kell a használni kívánt kijelző **displayio** kompatibilis meghajtóját is:
  - SH1106 ⟶ adafruit_displayio_sh1106.mpy
  - SSD1306 ⟶ adafruit_displayio_ssd1306.mpy
  - ST7735R TFT ⟶ adafruit_st7735r.mpy
  - E-paper 2.9” tricolor ⟶ adafruit_il0373.mpy

# ssd1306_bitmap_font.py

```
import board, displayio, terminalio
from i2cdisplaybus import I2CDisplayBus
from adafruit_display_text.bitmap_label import Label
from adafruit_displayio_ssd1306 import SSD1306
from adafruit_bitmap_font import bitmap_font

displayio.release_displays()
i2c = board.I2C()
display_bus = I2CDisplayBus(i2c, device_address=0x3C)
display = SSD1306(display_bus, width=128, height=32, rotation=180)

big_font = bitmap_font.load_font('fonts/LeagueSpartan-Bold-16.bdf')
small_font = bitmap_font.load_font('fonts/Chicago-12.bdf')
text1 = Label(big_font,text="CircuitPython",color=0xFFFFFF, X = 0, y = 8)
text2 = Label(small_font,text="on ESP32-C3",color=0xFFFFFF, X = 10, y = 28)
splash = displayio.Group()
display.root_group = splash
splash.append(text1)
splash.append(text2)

while True:
    pass
```

Ebben a programban kétféle betűtípust töltünk be, majd ezekkel két szöveg-címkét készítünk és jelenítünk meg.
A kijelző itt **SSD1306** vezérlőjű OLED, 128 x 32 pixeles felbontással



# Hogyan helyezkedik el egy szöveg?

- A szöveg elhelyezkedése a font metrikából adódik
  - **Baseline:** a betűk alapvonala — ez a referencia.
  - **Ascent:** legmagasabb karakter teteje és a baseline távolsága
  - **Descent:** lefelé lógó részek (pl. „g”, „p”) és a baseline távolsága
  - **Bounding box** = a szöveg szélessége és magassága (ascent + descent)


```
big_font = bitmap_font.load_font('fonts/LeagueSpartan-Bold-16.bdf')
small_font = bitmap_font.load_font('fonts/Chicago-12.bdf')
print("Big font ascent:", big_font.ascent)
print("Big font descent:", big_font.descent)
print("Small font ascent:", small_font.ascent)
print("Small font descent:", small_font.descent)
```

```
Big font ascent: 15
Big font descent: 4
Small font ascent: 9
Small font descent: 3
```

# Szöveg pozicionálása

- Amikor a szöveg helyét a Label x és y tulajdonságaival beállítjuk, valójában a szöveg referenciapontját adjuk meg
- Ez a referenciapont a nagybetűk magasságának a felénél (az ascent felénél) helyezkedik el, vagyis körülbelül ascent/2-vel van lejjebb, mint a bounding box bal felső sarka

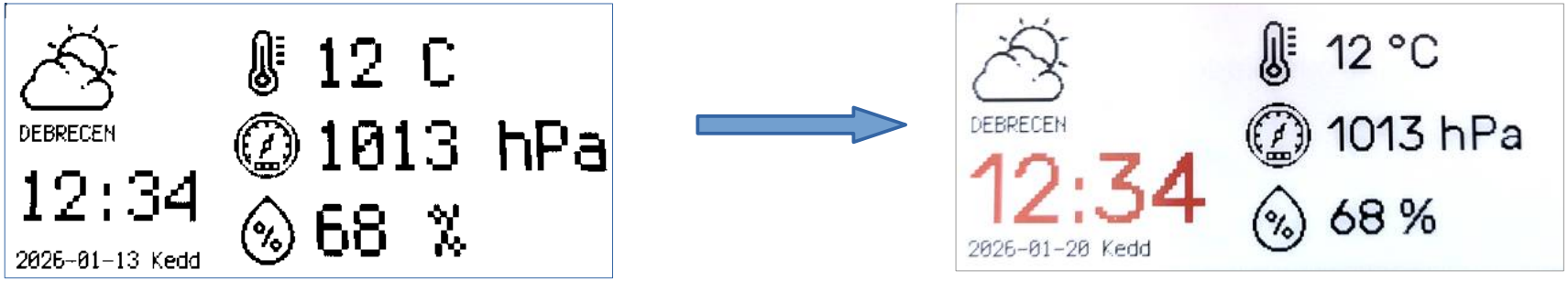

- Példa: **LeagueSpartan-Bold-16.bdf** font esetén **ascent** értéke 15, így `Label.y = 8` beállítással érhetjük el, hogy a szövegsor felül ne „lógjon ki” a képernyőről

# „Megfontolt” dashboard demo

- Az előző előadásban elkészített időjárás-áttekintő kijelző demót továbbfejlesztjük: a felnagyított **terminalio.FONT** helyett simább rajzolatú bitmap fontot használva
- A kivitelezéshez a Google Rubik fontcsaládjából választottuk a **regular** típust

2025/26 Hobbielektronika, Megtestesülés Plébánia, Debrecen

- A **FontForge** program segítségével készítettük el a TrueType formátumú fontból a 24 pt méretű bitmap fontot **Latin2** kódtáblával és exportáltuk **.bdf** formátumba
- A **rubik_r24.bdf** fontot az idő (scale=2), a hőmérséklet, a légnyomás és a páratartalom kijelzésére használtuk. Bónusz: a fok jel is kiíratható…



# dashboard_demo_fontos.py – 3/1.

```
import time, board, displayio, terminalio, fourwire
from adafruit_display_text.bitmap_label import Label
import adafruit_il0373
from adafruit_bitmap_font import bitmap_font
rubik24 = bitmap_font.load_font("/fonts/rubik_r24.bdf")
displayio.release_displays()
spi = board.SPI()
epd_cs = board.IO3
epd_dc = board.IO2
epd_reset = board.IO1
epd_busy = board.IO0
display_bus=fourwire.FourWire(spi,command=epd_dc,chip_select=epd_cs,reset=epd_reset,baudrate=1000000)
display = adafruit_il0373.IL0373(display_bus, width=296, height=128, rotation=270,
    busy_pin=epd_busy, highlight_color=0xFF0000)

root = displayio.Group()
palette = displayio.Palette(3)
palette[0] = 0xFFFFFF   # fehér
palette[1] = 0x000000   # fekete
palette[2] = 0xFF0000   # piros
bg_bitmap = displayio.Bitmap(296, 128, 1)  # Fehér háttér
bg_bitmap.fill(0)  # fehér
bg = displayio.TileGrid(bg_bitmap, pixel_shader=palette)
root.append(bg)
```

# dashboard_demo_fontos.py – 3/2.

- A baloldali elemek: egy ikon és három szövegcímke (a **time_label** 2x nagyítással)
- Csak a time_label kiírása változott meg

```
# Időjárás ikon (48 x 48 px)
weather_bmp = displayio.OnDiskBitmap("/icon1.bmp")
icon = displayio.TileGrid(weather_bmp, pixel_shader=weather_bmp.pixel_shader, x=4, y=4)
root.append(icon)

# Helynév
city = Label(terminalio.FONT,text="DEBRECEN",
    color=0x000000, scale=1, x=4, y=60)
root.append(city)

# Idő (nagy)
time_label = Label(rubik24, text="12:34", color=0xFF0000, scale=2, x=4, y=90)
root.append(time_label)

# Dátum
date_label = Label(terminalio.FONT, text="2026-01-13 Kedd",
    color=0x000000, scale=1, x=4, y=120)
root.append(date_label)
```

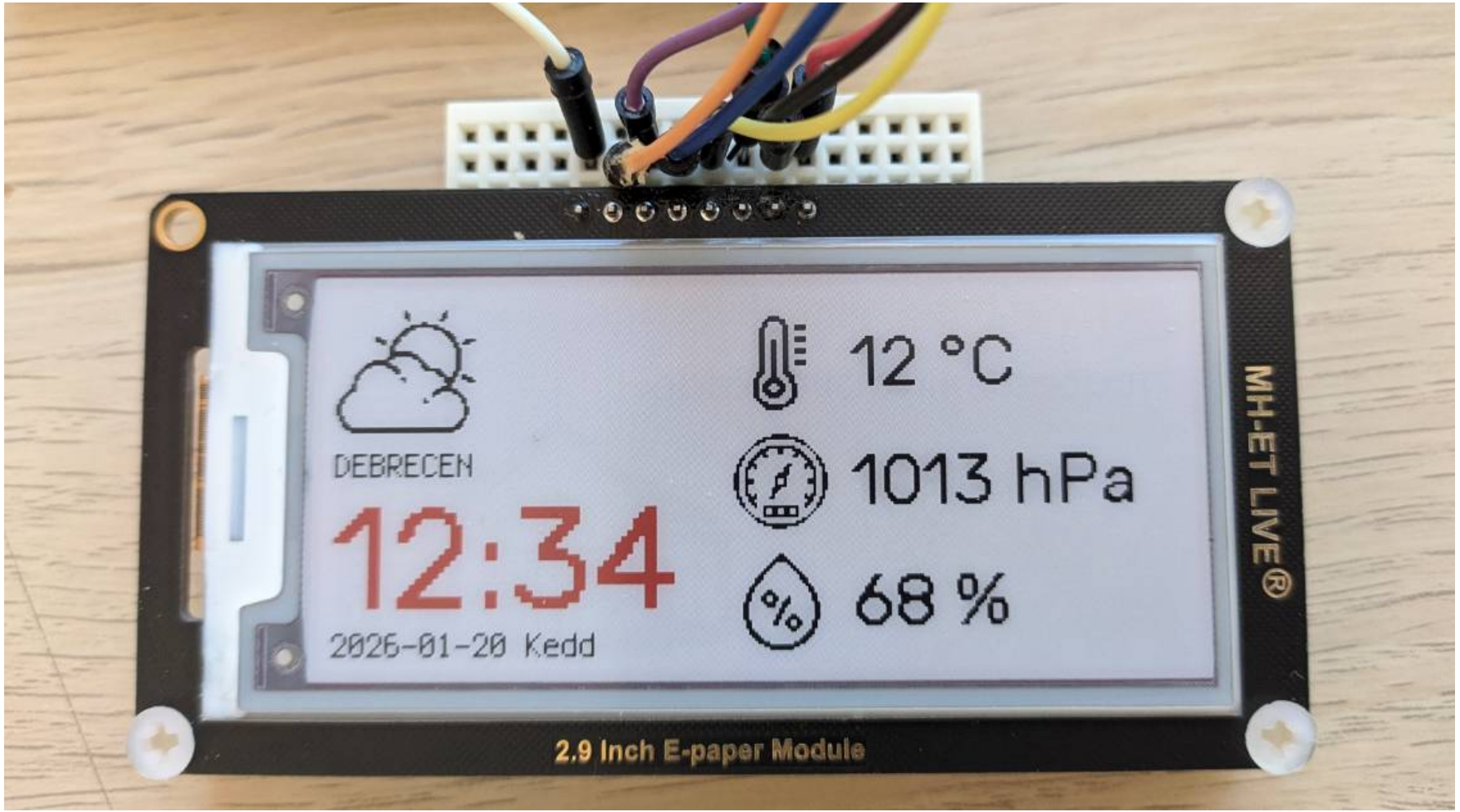
# dashboard_demo.py – 3/3.

- A jobboldali elemek (egy-egy ikon és szövegcímke) egy paraméterezhető függvénnyel hatékonyan létrehozhatók (a változások pirossal vannak jelölve)

```
def add_row(y, bmp_filename, text):
    """Egy sor: ikon + nagy szöveg."""
    bmp = displayio.OnDiskBitmap(bmp_filename)
    ic = displayio.TileGrid(bmp, pixel_shader=bmp.pixel_shader,x=140,y=y)
    root.append(ic)
    lbl = Label(rubik24, text=text,
        color=0x000000, scale=1, x=180, y=y+16)
    root.append(lbl)
    return lbl

temp_label  = add_row(10, "/temp.bmp", "12 °C")
press_label = add_row(50, "/press.bmp", "1013 hPa")
hum_label   = add_row(90, "/humidity.bmp", "68 %")

# --- MEGJELENÍTÉS ---
display.root_group = root
while display.busy:
    pass
display.refresh()
```

```
KÉPELEMEK HIERARCHIÁJA (displayio struktúra):

  display.root_group
  └── root (Group)
      ├── bg (TileGrid: fehér háttér)
      ├── icon (TileGrid: bal oldali időjárás ikon)
      ├── city (Label: városnév)
      ├── time_label (Label: idő, piros)
      ├── date_label (Label: dátum)
      ├── [add_row(…)] hőmérséklet ikon + Label
      ├── [add_row(…)] légnyomás ikon + Label
      └── [add_row(…)] páratartalom ikon + Label
```

# dashboard_demo_fontos.py eredménye



# weather_dashboard.py – 6/1.

```
import os, wifi, socketpool, adafruit_requests, time, board, displayio, terminalio, fourwire
from adafruit_display_text.bitmap_label import Label
import adafruit_il0373
from adafruit_bitmap_font import bitmap_font
rubik24 = bitmap_font.load_font("/fonts/rubik_r24.bdf")
displayio.release_displays()
spi = board.SPI()
epd_cs = board.IO3
epd_dc = board.IO2
epd_reset = board.IO1
epd_busy = board.IO0
display_bus=fourwire.FourWire(spi,command=epd_dc,chip_select=epd_cs,reset=epd_reset,baudrate=1000000)
display = adafruit_il0373.IL0373(display_bus, width=296, height=128, rotation=270,
    busy_pin=epd_busy, highlight_color=0xFF0000)

root = displayio.Group()
palette = displayio.Palette(3)
palette[0] = 0xFFFFFF   # fehér
palette[1] = 0x000000   # fekete
palette[2] = 0xFF0000   # piros
bg_bitmap = displayio.Bitmap(296, 128, 1)  # Fehér háttér
bg_bitmap.fill(0)  # fehér
bg = displayio.TileGrid(bg_bitmap, pixel_shader=palette)
root.append(bg)
```

# weather_dashboard.py – 6/2.

```
# Időjárás ikon (48 x 48 px)
weather_bmp = displayio.OnDiskBitmap("/icon1.bmp")
icon = displayio.TileGrid(weather_bmp, pixel_shader=weather_bmp.pixel_shader, x=4, y=4)
root.append(icon)

# Helynév
city = Label(terminalio.FONT,text="DEBRECEN",
    color=0x000000, scale=1, x=4, y=60)
root.append(city)

# Idő (nagy)
time_label = Label(rubik24, text="12:34", color=0xFF0000, scale=2, x=4, y=90)
root.append(time_label)

# Dátum
date_label = Label(terminalio.FONT, text="2026-01-13 Kedd",
    color=0x000000, scale=1, x=4, y=120)
root.append(date_label)
```

# weather_dashboard.py – 6/3.

```
def add_row(y, bmp_filename, text):
    """Egy sor: ikon + nagy szöveg."""
    bmp = displayio.OnDiskBitmap(bmp_filename)
    ic = displayio.TileGrid(bmp, pixel_shader=bmp.pixel_shader,x=140,y=y)
    root.append(ic)
    lbl = Label(rubik24, text=text,
        color=0x000000, scale=1, x=180, y=y+16)
    root.append(lbl)
    return lbl

temp_label  = add_row(10, "/temp.bmp", "12 °C")
press_label = add_row(50, "/press.bmp", "1013 hPa")
hum_label   = add_row(90, "/humidity.bmp", "68 %")

# --- MEGJELENÍTÉS ---
display.root_group = root
while display.busy:
    pass
```

# weather_dashboard.py – 6/4.

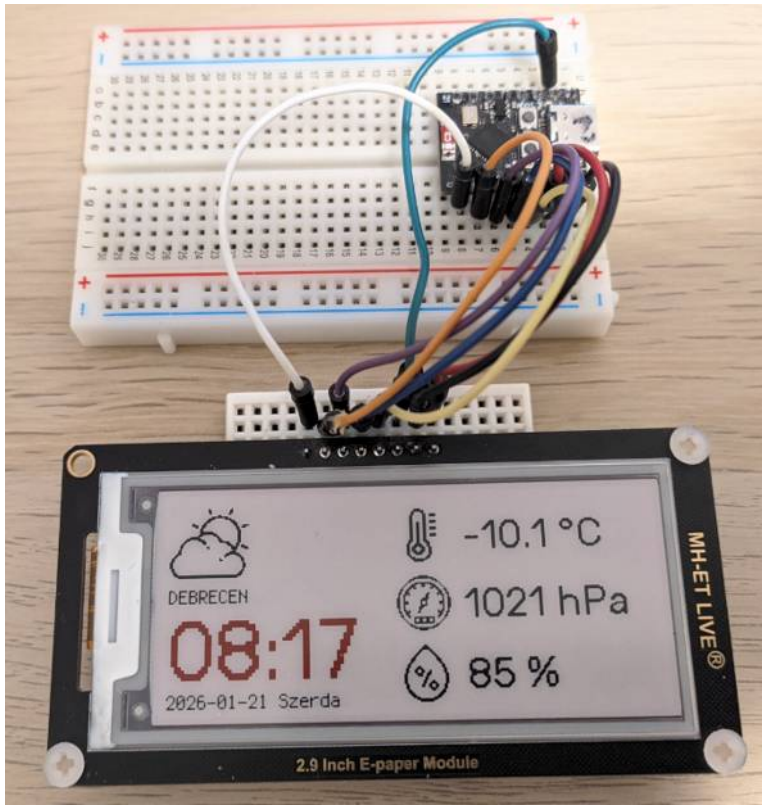
```
# ---------------------------------------------------------
# 6. WIFI + HTTP
# ---------------------------------------------------------

wifi.radio.connect(
    os.getenv("CIRCUITPY_WIFI_SSID"),
    os.getenv("CIRCUITPY_WIFI_PASSWORD")
)

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

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

# ---------------------------------------------------------
# 7. FŐ CIKLUS
# ---------------------------------------------------------
napok = ["Hétfő", "Kedd", "Szerda", "Csütörtök", "Péntek", "Szombat", "Vasárnap"]
last_dt = 0
```

# weather_dashboard.py – 6/5.

```
while True:
    try:
        data = requests.get(url).json()
        utc_ts = data["dt"]
        if utc_ts != last_dt:
            last_dt = utc_ts
            offset = data["timezone"]
            now = time.localtime(utc_ts + offset)
            time_label.text = "{:02}:{:02}".format(now.tm_hour, now.tm_min)
            date_str = "{:04}-{:02}-{:02} {}".format(now.tm_year, now.tm_mon, now.tm_mday,
               napok[now.tm_wday])
            date_label.text = date_str

            temp_c = data['main']['temp'] - 273.15
            temp_label.text = f"{temp_c:.1f} °C"
            humidity = data['main']['humidity']
            hum_label.text = f"{humidity} %"
            pressure = data['main']['pressure']
            press_label.text = f"{pressure} hPa"

            while display.busy:
                pass
            display.refresh()
```

# weather_dashboard.py – 6/6.

```
            print("Idő: {:02}:{:02}".format(now.tm_hour, now.tm_min))
            print("Hőmérséklet: {:.1f} °C".format(temp_c))
            print("Páratartalom:", humidity, "%")
            print("Légnyomás:", pressure, "hPa")
            print("---")

            time.sleep(180)

    except Exception as e:
        print("Hiba:", e)

    time.sleep(10)
```



# Az ESP32 C3 Super Mini kártya kivezetései