

Velocidad del Viento al paso de las horas en Grecia

Lo que yo cambié en el código fueron los parámetros para obtener la longitud y latitud de Grecia, además modifiqué los siguientes datos:

- (modifiqué latitude y longitude en la URL).
- parámetros: hourly=relativehumidity_2m, windspeed_10m.
- Ajusté marcadores (ej. marker="s"), grosor de línea (linewidth), transparencia (alpha).
- Agregué la rejilla: ax.grid(True, linestyle="--", alpha=.5).

El código quedó de la siguiente manera:

```
import tkinter as tk

from tkinter import ttk, messagebox

import requests

import matplotlib.pyplot as plt

from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg


def fetch_data():

    """

    Conecta con la API de Open-Meteo y obtiene temperaturas horarias

    de Grecia (últimas 24 horas).

    Devuelve dos listas: horas y temperaturas.

    """

    try:

        url = (

            "https://api.open-meteo.com/v1/forecast"

            "?latitude=39.22&longitude=22.01"
```

```

        "&hourly=windspeed_10m&past_days=1"

        "&timezone=auto"
    )

    response = requests.get(url, timeout=15)

    response.raise_for_status()

    data = response.json()

    horas = data["hourly"]["time"]

    temperaturas = data["hourly"]["windspeed_10m"]

    return horas, temperaturas

except Exception as e:

    messagebox.showerror("Error", f"No se pudieron obtener los datos:\n{e}")

    return [], []

def create_line_chart(horas, temps):

    """Gráfica de línea."""

    fig, ax = plt.subplots(figsize=(6, 3))

    ax.plot(horas, temps, linestyle="-", marker="o", markersize=3)

    ax.set_title("Velocidad del viento en Grecia (línea)")

    ax.set_xlabel("Hora")

    ax.set_ylabel("°C")

    ax.tick_params(axis="x", rotation=45)

    fig.tight_layout()

    return fig

def create_bar_chart(horas, temps):

    """Gráfica de barras."""

    fig, ax = plt.subplots(figsize=(6, 3))

    ax.bar(horas, temps)

    ax.set_title("Velocidad del viento en Grecia (barras)")

    ax.set_xlabel("Hora")

    ax.set_ylabel("°C")

    ax.tick_params(axis="x", rotation=45)

```

```
fig.tight_layout()
```

```
return fig
```

```
def mostrar_graficas(frm, horas, temps):
```

```
    """Inserta las tres gráficas en el frame de la ventana tkinter."""
```

```
    # Línea
```

```
    fig1 = create_line_chart(horas, temps)
```

```
    canvas1 = FigureCanvasTkAgg(fig1, master=frm)
```

```
    canvas1.draw()
```

```
    canvas1.get_tk_widget().pack(pady=10, fill="x")
```

```
    # Barras
```

```
    fig2 = create_bar_chart(horas, temps)
```

```
    canvas2 = FigureCanvasTkAgg(fig2, master=frm)
```

```
    canvas2.draw()
```

```
    canvas2.get_tk_widget().pack(pady=10, fill="x")
```

```
def open_win_canvas(parent: tk.Tk):
```

```
    """
```

```
    Crea la ventana secundaria con gráficas de la API.
```

```
    """
```

```
    win = tk.Toplevel(parent)
```

```
    win.title("Canvas con API (Open-Meteo) y gráficas")
```

```
    win.geometry("960x1000")
```

```
    frm = ttk.Frame(win, padding=12)
```

```
    frm.pack(fill="both", expand=True)
```

```
    # Botón para cargar datos y graficar
```

```
    def cargar():
```

```
        horas, temps = fetch_data()
```

```
        if horas and temps:
```

```
            mostrar_graficas(frm, horas, temps)
```

```
ttk.Button(frm, text="Cargar y mostrar gráficas", command=cargar).pack(pady=10)
```

```
# Para pruebas independientes (opcional)
```

```
if __name__ == "__main__":
```

```
    root = tk.Tk()
```

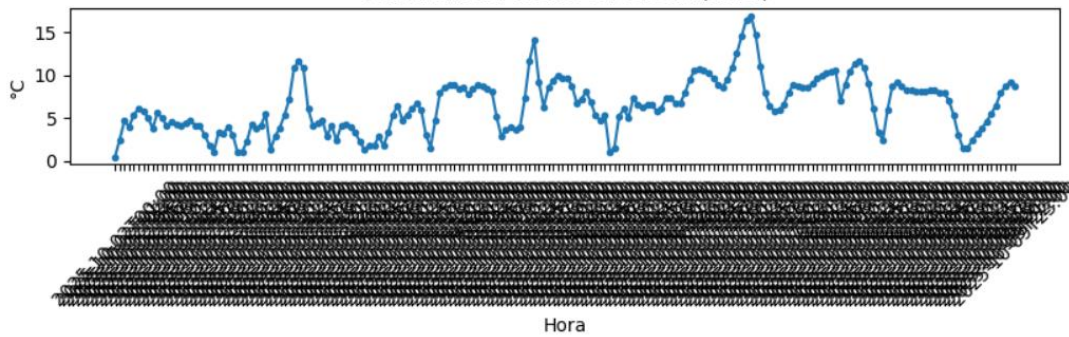
```
    root.title("Prueba win_canvas")
```

```
    ttk.Button(root, text="Abrir ventana Canvas", command=lambda: open_win_canvas(root)).pack(pady=20)
```

```
    root.mainloop()
```

Cargar y mostrar gráficas

Velocidad del viento en Grecia (línea)



Velocidad del viento en Grecia (barras)

