PROJECT DAY 6: Weather Forecast Project with Python

Nhập các thư viện

```
from tkinter import *
import requests
import json
import datetime
from PIL import ImageTk, Image
```

- **from tkinter import ***: Nhập tất cả các hàm và lớp từ thư viện **tkinter**. Thư viện này giúp tạo giao diện người dùng đồ họa (GUI).
- **import requests**: Nhập thư viện **requests**, dùng để gửi các yêu cầu HTTP đến các API để lấy dữ liệu.
- **import json**: Nhập thư viện **json**, dùng để phân tích dữ liệu JSON, định dạng thường được sử dụng để trả về dữ liệu từ API.
- **import datetime**: Nhập module **datetime**, dùng để làm việc với ngày và giờ.
- **from PIL import ImageTk, Image**: Nhập các lớp **ImageTk** và **Image** từ thư viện **PIL** (Python Imaging Library), dùng để xử lý và hiển thị hình ảnh trong ứng dụng.

Tạo cửa sổ ứng dụng chính

```
# necessary details
root = Tk()
root.title("Weather App")
root.geometry("450x700")
root['background'] = "white"
```

- **root = Tk()**: Tạo một đối tượng **Tk** mới, đây là cửa sổ chính của ứng dụng.
- **root.title("Weather App")**: Đặt tiêu đề cho cửa sổ là "Weather App".
- **root.geometry("450x700")**: Đặt kích thước của cửa sổ là 450 pixel chiều rộng và 700 pixel chiều cao.
- **root['background'] = "white"**: Đặt màu nền của cửa sổ là màu trắng.

Thêm hình ảnh

```
# Image
new = ImageTk.PhotoImage(Image.open('logo-300x117.png'))
panel = Label(root, image=new)
panel.place(x=0, y=520)
```

- **new = ImageTk.PhotoImage(Image.open('logo-300x117.png'))**: Mở tệp hình ảnh **logo-300x117.png** và chuyển đổi hình ảnh này thành đối tượng **PhotoImage** của **tkinter**.
- **panel = Label(root, image=new)**: Tạo một nhãn (**Label**) trong **root** để hiển thị hình ảnh **new**.
- **panel.place(x=0, y=520)**: Đặt vị trí của nhãn **panel** tại tọa độ (0, 520) trong cửa sổ.
- **ImageTk.PhotoImage()**: Chuyển đổi hình ảnh thành định dạng mà **tkinter** có thể sử dụng.
- **Label()**: Tạo một nhãn để hiển thị hình ảnh.
- **place()**: Đặt nhãn ở vị trí (0, 520) trong cửa sổ.

Hiển thị ngày và giờ

```
# Dates
dt = datetime.datetime.now()
date = Label(root, text=dt.strftime('%A--'), bg='white', font=("bold", 15))
date.place(x=5, y=130)
month = Label(root, text=dt.strftime('%m %B'), bg='white', font=("bold", 15))
month.place(x=100, y=130)

# Time
hour = Label(root, text=dt.strftime('%I : %M %p'),
             bg='white', font=("bold", 15))
hour.place(x=10, y=160)
```

- **dt = datetime.datetime.now()**: Lấy ngày và giờ hiện tại và lưu vào biến **dt**.
- **date = Label(root, text=dt.strftime('%A--'), bg='white', font=("bold", 15))**: Tạo một nhãn để hiển thị ngày trong tuần (ví dụ: Monday--). **strftime('%A--')** định dạng ngày thành chuỗi.
- **date.place(x=5, y=130)**: Đặt vị trí của nhãn **date** tại tọa độ (5, 130) trong cửa sổ.
- **month = Label(root, text=dt.strftime('%m %B'), bg='white', font=("bold", 15))**: Tạo một nhãn để hiển thị tháng (ví dụ: 05 May). **strftime('%m %B')** định dạng tháng thành chuỗi.
- **month.place(x=100, y=130)**: Đặt vị trí của nhãn **month** tại tọa độ (100, 130) trong cửa sổ.

- `hour = Label(root, text=dt.strftime('%I : %M %p'), bg='white', font=("bold", 15))`: Tạo một nhãn để hiển thị giờ (ví dụ: 10 : 30 PM). `strftime('%I : %M %p')` định dạng giờ thành chuỗi.
- `hour.place(x=10, y=160)`: Đặt vị trí của nhãn `hour` tại tọa độ (10, 160) trong cửa sổ.

Hiển thị hình ảnh dựa trên thời gian

```
# Theme for the respective time the application is used
if int((dt.strftime('%I'))) >= 8 & int((dt.strftime('%I'))) <= 5:
    img = ImageTk.PhotoImage(Image.open('moon-200x149.png'))
    panel = Label(root, image=img)
    panel.place(x=210, y=200)
else:
    img = ImageTk.PhotoImage(Image.open('sun-200x200.png'))
    panel = Label(root, image=img)
    panel.place(x=210, y=200)
```

- `if int((dt.strftime('%I'))) >= 8 & int((dt.strftime('%I'))) <= 5:`: Kiểm tra nếu giờ hiện tại nằm trong khoảng từ 8 giờ sáng đến 5 giờ chiều.
- `img = ImageTk.PhotoImage(Image.open('moon-200x149.png'))`: Nếu đúng, mở hình ảnh mặt trăng và chuyển đổi thành đối tượng `PhotoImage`.
- `panel = Label(root, image=img)`: Tạo một nhãn để hiển thị hình ảnh mặt trăng.
- `panel.place(x=210, y=200)`: Đặt vị trí của nhãn tại tọa độ (210, 200) trong cửa sổ.
- `else:`: Nếu không nằm trong khoảng từ 8 giờ sáng đến 5 giờ chiều.
- `img = ImageTk.PhotoImage(Image.open('sun-200x200.png'))`: Mở hình ảnh mặt trời và chuyển đổi thành đối tượng `PhotoImage`.
- `panel = Label(root, image=img)`: Tạo một nhãn để hiển thị hình ảnh mặt trời.
- `panel.place(x=210, y=200)`: Đặt vị trí của nhãn tại tọa độ (210, 200) trong cửa sổ.

Tạo ô nhập liệu tên thành phố

```
# City Search
city_name = StringVar()
city_entry = Entry(root, textvariable=city_name, width=45)
city_entry.grid(row=1, column=0, ipady=10, stick=W+E+N+S)
```

- `city_name = StringVar()`: Tạo một biến chuỗi trong `tkinter` để lưu tên thành phố.
- `city_entry = Entry(root, textvariable=city_name, width=45)`: Tạo một ô nhập liệu cho phép người dùng nhập tên thành phố, liên kết với biến `city_name`, và có độ rộng là 45 ký tự.
- `city_entry.grid(row=1, column=0, ipady=10, stick=W+E+N+S)`: Đặt ô nhập liệu vào lưới, hàng 1, cột 0, với khoảng đệm trong là 10 pixel và dán vào các hướng (West, East, North, South).

Hàm lấy dữ liệu thời tiết

```
def city_name():

    # API Call
    api_key = 'fe03f66944492599d78bbf22010988ee'
    api_request = requests.get("https://api.openweathermap.org/data/2.5/weather?q="
                               + city_entry.get() + "&units=metric&appid="+ api_key)

    api = json.loads(api_request.content)

    # Temperatures
    y = api['main']
    current_temprature = y['temp']
    humidity = y['humidity']
    tempmin = y['temp_min']
    tempmax = y['temp_max']

    # Coordinates
    x = api['coord']
    longtitude = x['lon']
    latitude = x['lat']
```

```
    # Country
    z = api['sys']
    country = z['country']
    citi = api['name']

    # Adding the received info into the screen
    lable_temp.configure(text=current_temprature)
    lable_humidity.configure(text=humidity)
    max_temp.configure(text=tempmax)
    min_temp.configure(text=tempmin)
    lable_lon.configure(text=longtitude)
    lable_lat.configure(text=latitude)
    lable_country.configure(text=country)
    lable_citi.configure(text=citi)
```

- `def city_name():`: Định nghĩa một hàm có tên `city_name`.
- `api_key = 'fe03f66944492599d78bbf22010988ee'`: Đặt khóa API cho OpenWeatherMap.
- `api_request = requests.get("https://api.openweathermap.org/data/2.5/weather?q=" + city_entry.get() + "&units=metric&appid=" + api_key)`: Gửi yêu cầu GET đến API của

OpenWeatherMap với tên thành phố lấy từ ô nhập liệu, sử dụng đơn vị đo là độ C và khóa API.

- `api = json.loads(api_request.content)`: Phân tích cú pháp nội dung JSON trả về từ API thành một từ điển Python.
- `y = api['main']`: Lấy phần 'main' từ phản hồi API, chứa thông tin nhiệt độ và độ ẩm.
- `current_temprature = y['temp']`: Lấy nhiệt độ hiện tại.
- `humidity = y['humidity']`: Lấy độ ẩm.
- `tempmin = y['temp_min']`: Lấy nhiệt độ thấp nhất.
- `tempmax = y['temp_max']`: Lấy nhiệt độ cao nhất.
- `x = api['coord']`: Lấy phần 'coord' từ phản hồi API, chứa thông tin tọa độ.
- `longtitude = x['lon']`: Lấy kinh độ.
- `latitude = x['lat']`: Lấy vĩ độ.
- `z = api['sys']`: Lấy phần 'sys' từ phản hồi API, chứa thông tin hệ thống.
- `country = z['country']`: Lấy tên quốc gia.
- `citi = api['name']`: Lấy tên thành phố.
- `lable_temp.configure(text=current_temprature)`: Cập nhật nhãn nhiệt độ hiện tại với giá trị lấy từ API.
- `lable_humidity.configure(text=humidity)`: Cập nhật nhãn độ ẩm với giá trị lấy từ API.
- `max_temp.configure(text=tempmax)`: Cập nhật nhãn nhiệt độ cao nhất với giá trị lấy từ API.
- `min_temp.configure(text=tempmin)`: Cập nhật nhãn nhiệt độ thấp nhất với giá trị lấy từ API.
- `lable_lon.configure(text=longtitude)`: Cập nhật nhãn kinh độ với giá trị lấy từ API.
- `lable_lat.configure(text=latitude)`: Cập nhật nhãn vĩ độ với giá trị lấy từ API.
- `lable_country.configure(text=country)`: Cập nhật nhãn quốc gia với giá trị lấy từ API.
- `lable_citi.configure(text=citi)`: Cập nhật nhãn tên thành phố với giá trị lấy từ API.
- Các dòng tiếp theo trích xuất dữ liệu cụ thể từ phản hồi và cập nhật các nhãn tương ứng trên GUI.

Nút tìm kiếm

```
# Search Bar and Button
city_nameButton = Button(root, text="Search", command=city_name)
city_nameButton.grid(row=1, column=1, padx=5, stick=W+E+N+S)
```

- `city_nameButton = Button(root, text="Search", command=city_name)`: Tạo một nút với nhãn "Search", khi nhấn vào nút này, hàm `city_name` sẽ được gọi.
- `city_nameButton.grid(row=1, column=1, padx=5, stick=W+E+N+S)`: Đặt nút vào lưới, hàng 1, cột 1, với khoảng đệm ngoài là 5 pixel và dán vào các hướng (West, East, North, South).

Các nhãn để hiển thị dữ liệu

```
# Current Temperature

lable_temp = Label(root, text="...", width=0, bg='white',
                   font=("Helvetica", 110), fg='black')
lable_temp.place(x=18, y=220)
```

```
# Other temperature details

humi = Label(root, text="Humidity: ", width=0,
             bg='white', font=("bold", 15))
humi.place(x=3, y=400)

lable_humidity = Label(root, text="...", width=0,
                       bg='white', font=("bold", 15))
lable_humidity.place(x=107, y=400)


maxi = Label(root, text="Max. Temp.: ", width=0,
             bg='white', font=("bold", 15))
maxi.place(x=3, y
```

(variable) root: Tk

```
max_temp = Label(root, text="...", width=0,
                 bg='white', font=("bold", 15))
max_temp.place(x=128, y=430)


mini = Label(root, text="Min. Temp.: ", width=0,
             bg='white', font=("bold", 15))
mini.place(x=3, y=460)

min_temp = Label(root, text="...", width=0,
                 bg='white', font=("bold", 15))
min_temp.place(x=128, y=460)
```

- `Label(root, text="...", width=0, bg='white', font=("bold", 15))`: Tạo các nhãn với văn bản ban đầu là "...", không có chiều rộng cụ thể (`width=0`), nền màu trắng, và phông chữ đậm cỡ 15.

- **place(x=10, y=63)**: Đặt nhãn tại tọa độ (10, 63) trong cửa sổ.
- **lable_citi**, **lable_country**, **lable_lon**, **lable_lat**, **lable_temp**, **lable_humidity**, **max_temp**, **min_temp**: Các nhãn này được tạo để hiển thị thông tin tương ứng như tên thành phố, quốc gia, kinh độ, vĩ độ, nhiệt độ, độ ẩm, nhiệt độ cao nhất, và nhiệt độ thấp nhất.

Ghi chú

```
# Note
note = Label(root, text="All temperatures in degree celsius",
             bg='white', font=("italic", 10))
note.place(x=95, y=495)
```

- **note = Label(root, text="All temperatures in degree celsius", bg='white', font=("italic", 10))**: Tạo một nhãn để hiển thị ghi chú rằng tất cả các nhiệt độ đều được đo bằng độ C, nền màu trắng, và phông chữ nghiêng cỡ 10.
- **place(x=95, y=495)**: Đặt nhãn tại tọa độ (95, 495) trong cửa sổ.

Vòng lặp chính

```
root.mainloop()
```

- **root.mainloop()**: Bắt đầu vòng lặp sự kiện chính của **tkinter**, chờ đợi các sự kiện (như nhấn nút) và cập nhật GUI tương ứng. Vòng lặp này giữ cho cửa sổ ứng dụng mở cho đến khi người dùng đóng nó.