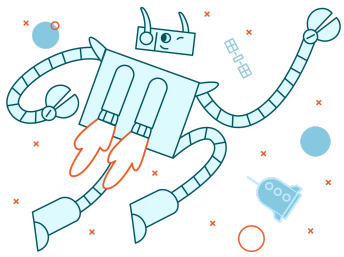
Rainy with a Chance of Packets



Goal

Develop a Python server and client that work together to provide weather information to users.

Tasks

1. Write a server (`weather_server.py`) that does the following:
    - Stores weather data for multiple cities in a dictionary format, where the keys are city names and the values are dictionaries containing weather information (e.g., temperature, humidity, description)
    - Example weather information dictionary:

```
weather_data = {
  "London": {"temperature": 18, "humidity": 60, "description": "Partly cloudy"},
  "Paris": {"temperature": 22, "humidity": 55, "description": "Sunny"},
    "New York": {"temperature": 20, "humidity": 70, "description": "Cloudy"},
  "Tokyo": {"temperature": 25, "humidity": 80, "description": "Rainy"}
}
```

    - Listens for incoming connections on port 5000
    - When a client connects, the server receives the name of the city from the client

- The server looks up the weather information for the requested city in its stored data
- If the city is found, the server formats the weather information into a readable string and sends it back to the client
- If the city is not found, the server sends an appropriate error message to the client
- The server closes the connection with the client after sending the response
- The server continues to listen for additional weather information requests

2. Write a client (`weather_client.py`) that does the following:
    - Prompts the user to enter the name of a city
    - Connects to the server on port 5000
    - Sends the name of the city to the server
    - Receives the weather information response from the server
    - Displays the received weather information to the user (if available) or displays an error message (if the city is not found)
    - Closes the connection with the server

Example of Running the Client

Let's assume you have implemented the server (`weather_server.py`) and client (`weather_client.py`) scripts according to the above instructions. Here's an example of how the client interaction would look like:

1. Run the server script using the command: `python weather_server.py`.

2. In another terminal, run the client script using the command: `python weather_client.py`.

3. The client will prompt you to enter the name of a city. Let's say you enter "Paris":

```
Enter the name of a city: Paris
```

4. The client sends the city name "Paris" to the server and waits for the response.

5. The server finds the weather information for Paris in its stored `weather_data` dictionary and sends the data back to the client. The client will display the received weather information:

```
Weather information for Paris:
Temperature: 22°C
Humidity: 55%
Description: Sunny
```

6. If you run the client again and enter a city that is not in the `weather_data` dictionary, such as "Berlin", the server will send an error message. The client will display the error:

```
Enter the name of a city: Berlin
City not found: Berlin
```

7. After displaying the weather information or error message, the client closes the connection with the server.

Note: Make sure the server is running and listening for connections before running the client script.

Submission

1. The `weather_server.py` file containing the complete code for the server, including the stored weather data.
2. The `weather_client.py` file containing the complete code for the client.

Bonus

- Implement a feature to allow the client to request weather information for multiple cities in a single session.
- Add error handling to gracefully handle situations like connection errors or invalid user inputs.