6.2.2 Get Started with OpenWeatherMap API

Awesome. You get it—mostly. We'll use the API setup to go out and get information when our clients ask us for it. So, now it's time to download the Python Requests Library and register for an API key.

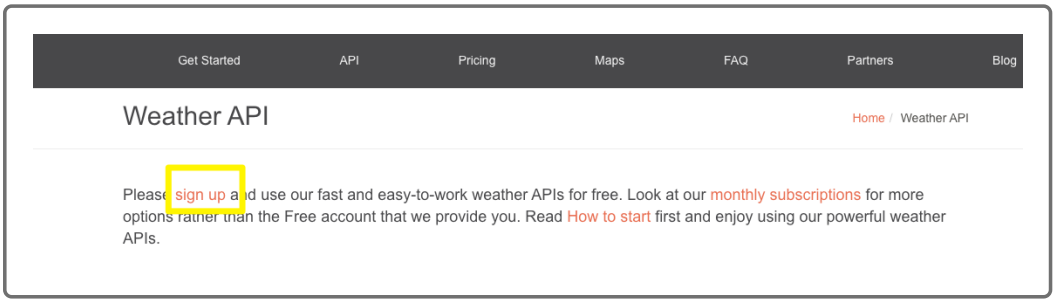
Register for an API key, then review a short example of how to retrieve weather data from an API call.

Register for an API Key

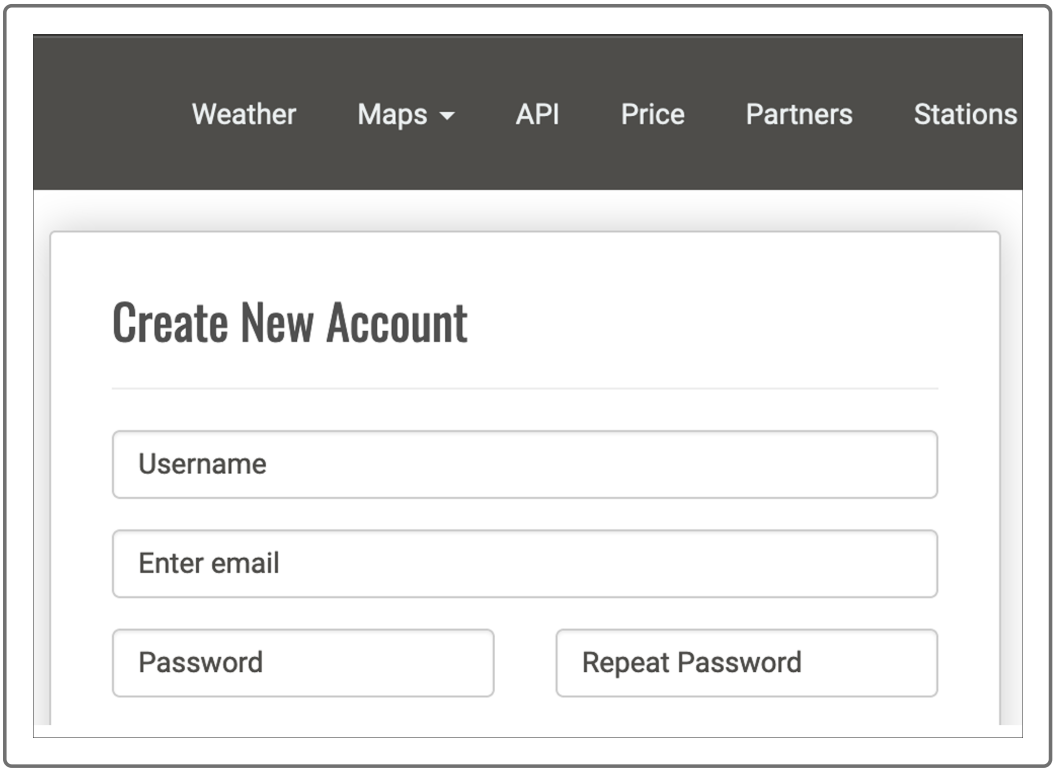
Follow these steps to register for an OpenweatherMap API key:

1. Navigate to the **OpenWeatherMap website (https://openweathermap.org/api)** .

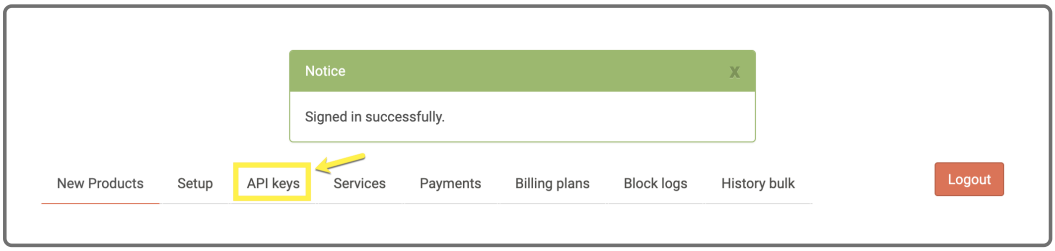
2. Click "sign up."



3. Complete the form Create New Account.

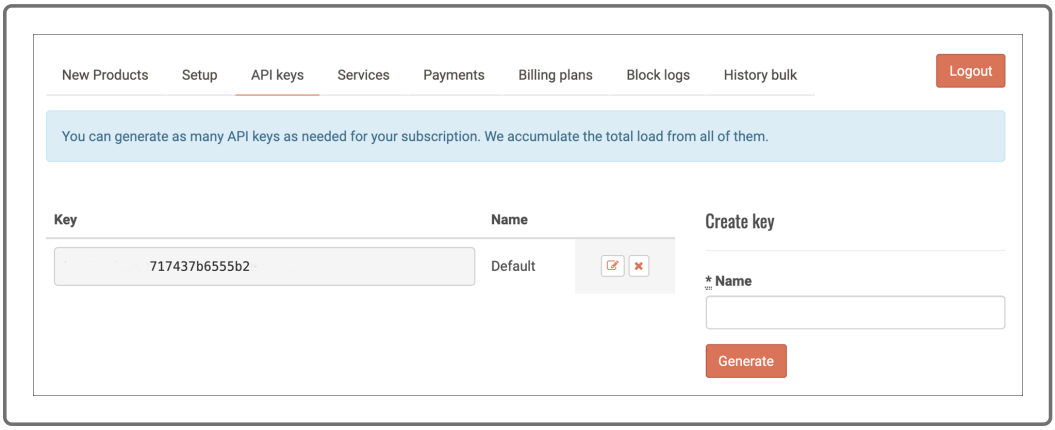


4. Once you have a new account, sign in and click on "API keys."



5. The site will likely generate an API key automatically. If not, under "Create key," type a name in the available cell and click "Generate" to

create an API key.



6. Save your API key to a Python file, which we'll add as a dependency to your `WeatherPy.ipynb` file.

    - Navigate to your World_Weather_Analysis folder and launch Jupyter Notebook.
    - Click the New button and select Text File.
    - Rename the text file `config.py`.
    - On the first line, type `weather_api_key=""` and add your API key between the double quotation marks.
    - Save and close the `config.py` file.

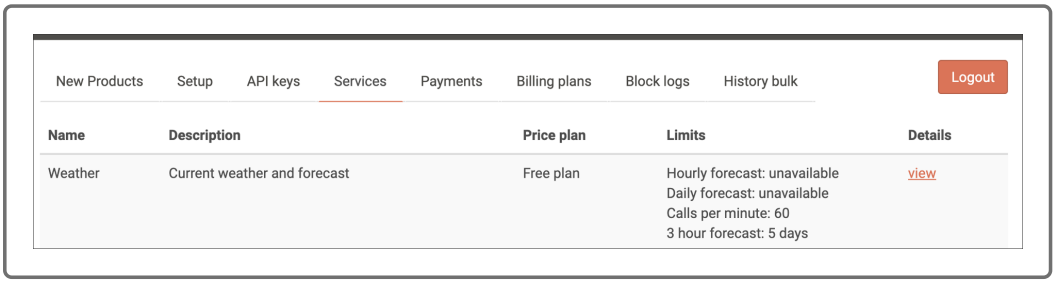
NOTE

> You can also create the `config.py` file using VS Code.

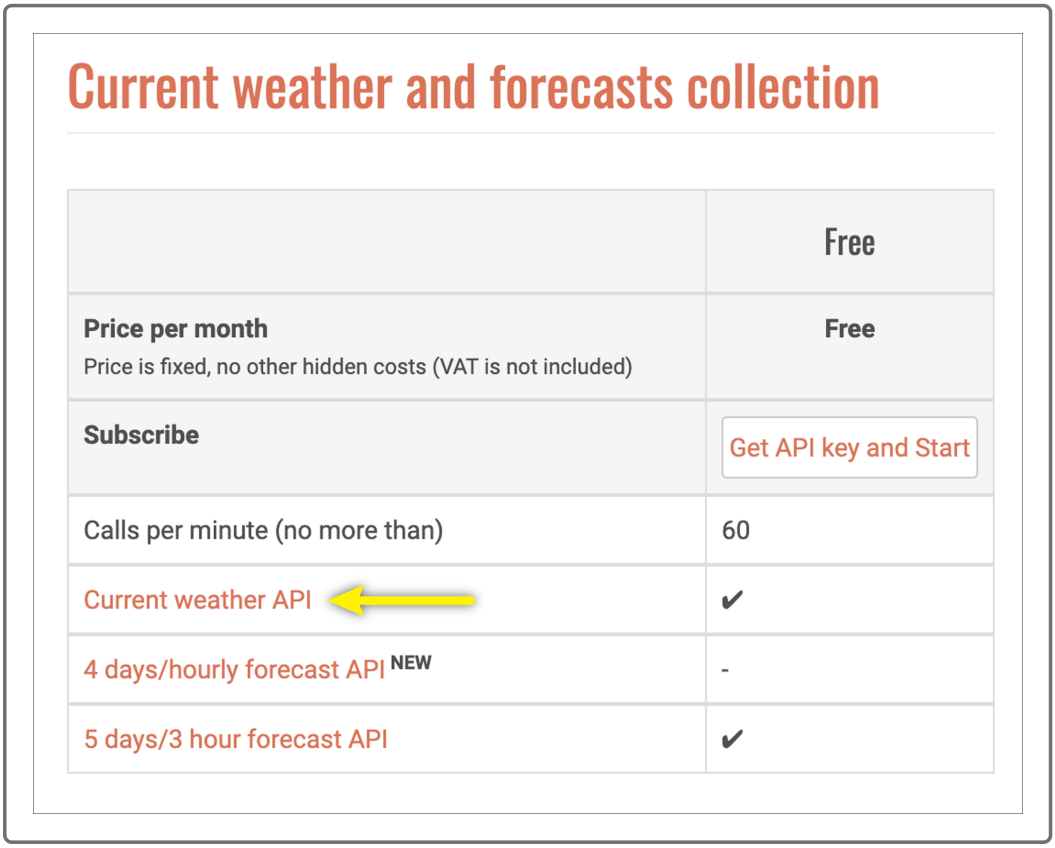
IMPORTANT

> Don't share your API key with anyone, and do not add the `config.py` file to your GitHub repository–someone might copy and use it, and you could incur charges on your credit card.

7. Click on "Services" for details on your free subscription and its limitations.



8. Click on "View" to see more options on your plan, and then click on "Current weather API" to see how to get the current weather from a city.



We'll refer to this documentation when we make an API call to the server.

The JavaScript Object Notation Format for API Data

The API has reached the website or server, its endpoint, and now we can retrieve data from the website. When we retrieve data from a website, we have to make a "request," which returns data in a text format, not in a tab- or comma-separated file.

One format we can use to parse data is **JavaScript Object Notation (JSON)**. The JSON format is also referred to as an "object" or "JSON object." The data inside a JSON object opens and closes with curly braces, much like a Python dictionary. Inside the JSON object is a collection of dictionaries and arrays.

Below is an example of what weather data looks like in the JSON format when we request it from the OpenWeatherMap website. There are curly braces that wrap the data, and inside the curly braces are dictionaries and arrays.

```
{
  - coord: {
        lon: -0.13,
        lat: 51.51
    },
  - weather: [
      - {
            id: 300,
            main: "Drizzle",
            description: "light intensity drizzle",
            icon: "09d"
        }
    ],
    base: "stations",
  - main: {
        temp: 280.32,
        pressure: 1012,
        humidity: 81,
        temp_min: 279.15,
        temp_max: 281.15
    },
    visibility: 10000,
  - wind: {
        speed: 4.1,
        deg: 80
    },
  - clouds: {
        all: 90
    },
    dt: 1485789600,
  - sys: {
        type: 1,
        id: 5091,
        message: 0.0103,
        country: "GB",
        sunrise: 1485762037,
        sunset: 1485794875
    },
    id: 2643743,
    name: "London",
    cod: 200
}
```

The Python Requests Library

To request JSON data over the internet, we use the Requests Library in Python. The Anaconda installation comes with version 2.22 of the Requests Library.

Confirm you have the latest version of the Requests Library using the command line, or in the Jupyter Notebook environment. Follow the instructions for your operating system.

Check out the macOS instructions below, or jump to the **Windows instructions**.

macOS

1. Launch the command line and activate your PythonData environment.
2. After the prompt, type `$ python` to launch Python.
3. After the Python prompt, `>>>`, type `import requests` and press Enter.
4. On the next line, type `requests.__version__` and press Enter.
5. The output should be version `2.22.0` or later.

Alternatively, you can check the version of request in Jupyter Notebook.

In Jupyter Notebook, create a new file if one hasn't been created. Add the following code to the new cell and run it.

```
import requests
requests.__version__
```

The output should be `2.22.0` or later.

If you have an older version, please upgrade it in your PythonData environment by typing `conda install -c conda-forge requests` at the command prompt and press Enter.

Windows

1. Launch the Anaconda Prompt for your PythonData environment.
2. After the Python prompt, `>`, type `python` to launch Python.
3. At the Python prompt, `>>>`, type `import requests` and press Enter.
4. On the next line type `requests.__version__` and press Enter.
5. The output should be version `2.22.0` or later.

Alternatively, you can check the version of request in Jupyter Notebook.

In Jupyter Notebook, create a new file if one hasn't been created. Add the following code to the new cell and run it.

```
import requests
requests.__version__
```

The output should be `2.22.0` or later.

If you have an older version, please upgrade it in your PythonData environment by typing `conda install -c conda-forge requests`, at the command prompt and press Enter.

NOTE

> For more information about the Requests Library, see the following documentation:

> **Requests: HTTP for Humans**
> **(https://requests.kennethreitz.org/en/master/)**
>
> **Quickstart**
> **(https://requests.kennethreitz.org/en/master/user/quickstart/#make-a-request)**