6.2.3 Make an API Call

You're about to start just trying random functions to see what works when Jack texts you and tells you your company has a free in-house API tutorial that was designed to teach new team members how to do this very thing. And it gets better–he sent you the link to the tutorial, too. Excited, you click through to access the API tutorial.

In our `API_practice` file, add a new cell after the code we wrote to get the unique cities from with the citipy module. In the new cell, we will import the Requests Library and your API key from the `config.py` file.

NOTE

> Your `config.py` file should be in the same folder as your `API_practice.ipynb` file or any Jupyter Notebook file that is accessing the `config.py` file.

Add the following code to the new cell and run the cell.

```
# Import the requests library.
import requests

# Import the API key.
from config import weather_api_key
```

NOTE

> If you get a `ModuleNotFoundError` message, then either your `config.py` file is not in the same folder as the Jupyter Notebook file, or the name of your `config.py` file is not the same in the import statement.
>
> If you get a `ImportError` message, then the variable for your API key in the code statement is not the same as the variable in your `config.py` file.

Make an API Call

Before we make an API call for the OpenWeatherMap, we need to use the URL provided on the OpenWeatherMap website.

Let's look at the documentation on the OpenWeatherMap website.

1. Navigate to the **OpenWeatherMap API documentation for current weather (https://openweathermap.org/current)**.

2. This page provides instructions on how to make the API call by city name. The structure of our URL should look like the following:

   `api.openweathermap.org/data/2.5/weather?q=city&appid=b6907d289e10d714a6e88b30761fae22`

3. Add your API key and the city from the cities array for each call.

Next let's practice making an API call and look at the data returned from the API call. Add the following code to a new cell and run the cell.

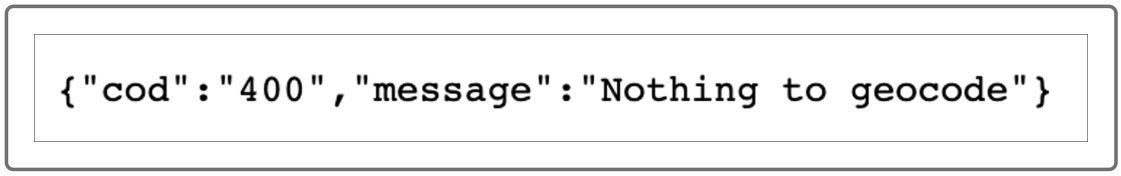
```
# Starting URL for Weather Map API Call.
url = "http://api.openweathermap.org/data/2.5/weather?units=Imperial&APPID="
print(url)
```

NOTE

> Once you print out your URL, be sure to delete your printed URL from your notebook. This prevents your private API key from being stored in plain text.

You may have noticed that we added another feature to the URL: `units=Imperial`. There are three unit options: standard, metric, and imperial. Navigating to **the units section of the current weather data page (https://openweathermap.org/current#data)** will show you the options for the unit format: standard metric, and imperial.

When we run the cell, the output will be a URL. Click the URL, and a new window will open in your default web browser. The URL will return a 400 message because we haven't added a city to our URL.



Now let's add a city to the URL to get the current weather data. Add the following code to a new cell and run the cell.

```
# Create an endpoint URL for a city.
city_url = url + "&q=" + "Boston"
print(city_url)
```

In the code, we are creating a string to get the weather data for Boston by using the `city_url`. To create the `city_url` we add the parameter, `&q=` and "Boston" to the `url`.

The output of this cell will also be a URL. Click the URL and a new window will open in your default web browser that shows the current weather data for Boston.


```
'{"coord":{"lon":-71.06,"lat":42.36},"weather":[{"id":804,"main":"Clouds","description":"over
cast clouds","icon":"04d"}],"base":"stations","main":{"temp":80.01,"pressure":1011,"humidit
y":65,"temp_min":77,"temp_max":82.99},"visibility":16093,"wind":{"speed":11.41,"deg":200},"cl
ouds":{"all":90},"dt":1565193597,"sys":{"type":1,"id":3486,"message":0.0097,"country":"US","s
unrise":1565170963,"sunset":1565222243},"timezone":-14400,"id":4930956,"name":"Boston","cod":
200}'
```


To make the JSON format readable, **add the Chrome JSONView extension**

(https://chrome.google.com/webstore/detail/jsonview/chklaanhfefbnpoihckbnefhakgolnmc) to your Chrome web browser. The Boston weather data will appear as follows.

```
{
  - coord: {
        lon: -71.06,
        lat: 42.36
    },
  - weather: [
    - {
            id: 501,
            main: "Rain",
            description: "moderate rain",
            icon: "10d"
        },
    - {
            id: 211,
            main: "Thunderstorm",
            description: "thunderstorm",
            icon: "11d"
        },
    - {
            id: 721,
            main: "Haze",
            description: "haze",
            icon: "50d"
        },
    - {
            id: 701,
            main: "Mist",
            description: "mist",
            icon: "50d"
        }
    ],
    base: "stations",
  - main: {
        temp: 79.68,
        pressure: 1009,
        humidity: 74,
        temp_min: 73.4,
        temp_max: 82.99
    },
```

```
    visibility: 16093,
  - wind: {
        speed: 5.82,
        deg: 210
    },
  - rain: {
        1h: 2.54
    },
  - clouds: {
        all: 75
    },
    dt: 1565207600,
  - sys: {
        type: 1,
        id: 3486,
        message: 0.0126,
        country: "US",
        sunrise: 1565170963,
        sunset: 1565222243
    },
    timezone: -14400,
    id: 4930956,
    name: "Boston",
    cod: 200
}
```


NOTE

> You will need to use an extension on Safari to view JSON shown in the above image. Firefox does not need a JSON extension to view JSON files.
>
> Since we're retrieving current weather data, the JSON data shown in the above image will differ from that in your endpoint printout. Instead, your weather data will reflect the specific time you ran the code.