PY0101EN-5.2_API_2.v2

November 18, 2023

1 Hands-on Lab: API Examples

1.1 Random User and Fruityvice API Examples

Estimated time needed: **30** minutes

1.2 Objectives

After completing this lab you will be able to:

- Load and use RandomUser API, using `RandomUser()` Python library
- Load and use Fruityvice API, using `requests` Python library
- Load and use Open-Joke-API, using `requests` Python library

The purpose of this notebook is to provide more examples on how to use simple APIs. As you have already learned from previous videos and notebooks, API stands for Application Programming Interface and is a software intermediary that allows two applications to talk to each other.

The advantages of using APIs: * **Automation**. Less human effort is required and workflows can be easily updated to become faster and more
productive. * **Efficiency**. It allows to use the capabilities of one of the already developed APIs than to try to independently implement some functionality from scratch.

The disadvantage of using APIs: * **Security**. If the API is poorly integrated, it means it will be vulnerable to attacks, resulting in data breeches or losses having financial or reputation implications.

One of the applications we will use in this notebook is Random User Generator. RandomUser is an open-source, free API providing developers with randomly generated users to be used as placeholders for testing purposes. This makes the tool similar to Lorem Ipsum, but is a placeholder for people instead of text. The API can return multiple results, as well as specify generated user details such as gender, email, image, username, address, title, first and last name, and more. More information on RandomUser can be found here.

Another example of simple API we will use in this notebook is Fruityvice application. The Fruityvice API web service which provides data for all kinds of fruit! You can use Fruityvice to find out interesting information about fruit and educate yourself. The web service is completely free to use and contribute to.

1.3 Example 1: RandomUser API

Bellow are Get Methods parameters that we can generate. For more information on the parameters, please visit this documentation page.

1.4 Get Methods

- get_cell()
- get_city()
- get_dob()
- get_email()
- get_first_name()
- get_full_name()
- get_gender()
- get_id()
- get_id_number()
- get_id_type()
- get_info()
- get_last_name()
- get_login_md5()
- get_login_salt()
- get_login_sha1()
- get_login_sha256()
- get_nat()
- get_password()
- get_phone()
- get_picture()
- get_postcode()
- get_registered()
- get_state()
- get_street()
- get_username()
- get_zipcode()

To start using the API you can install the `randomuser` library running the `pip install` command.

[ ]:
```
!pip install randomuser
```

Then, we will load the necessary libraries.

[ ]:
```
from randomuser import RandomUser
import pandas as pd
```

First, we will create a random user object, r.

[ ]:
```
r = RandomUser()
```

Then, using `generate_users()` function, we get a list of random 10 users.

[ ]:
```
some_list = r.generate_users(10)
```

[ ]:
```
some_list
```

The **"Get Methods"** functions mentioned at the beginning of this notebook, can generate the required parameters to construct a dataset. For example, to get full name, we call `get_full_name()`

function.

```
[ ]: name = r.get_full_name()
```

Let's say we only need 10 users with full names and their email addresses. We can write a "for-loop" to print these 10 users.

```
[ ]: for user in some_list:
         print (user.get_full_name()," ",user.get_email())
```

1.5 Exercise 1

In this Exercise, generate photos of the random 10 users.

```
[ ]: ## Write your code here
```

Click here for the solution

```
for user in some_list:
    print (user.get_picture())
```

To generate a table with information about the users, we can write a function containing all desirable parameters. For example, name, gender, city, etc. The parameters will depend on the requirements of the test to be performed. We call the Get Methods, listed at the beginning of this notebook. Then, we return pandas dataframe with the users.

```
[ ]: def get_users():
         users =[]

         for user in RandomUser.generate_users(10):
             users.append({"Name":user.get_full_name(),"Gender":user.
     ↪get_gender(),"City":user.get_city(),"State":user.get_state(),"Email":user.
     ↪get_email(), "DOB":user.get_dob(),"Picture":user.get_picture()})

         return pd.DataFrame(users)
```

```
[ ]: get_users()
```

```
[ ]: df1 = pd.DataFrame(get_users())
```

Now we have a *pandas* dataframe that can be used for any testing purposes that the tester might have.

1.6 Example 2: Fruityvice API

Another, more common way to use APIs, is through `requests` library. The next lab, Requests and HTTP, will contain more information about requests.

We will start by importing all required libraries.

```
[ ]: import requests
     import json
```

We will obtain the fruityvice API data using `requests.get("url")` function. The data is in a json format.

```
[ ]: data = requests.get("https://fruityvice.com/api/fruit/all")
```

We will retrieve results using `json.loads()` function.

```
[ ]: results = json.loads(data.text)
```

We will convert our json data into *pandas* data frame.

```
[ ]: pd.DataFrame(results)
```

The result is in a nested json format. The 'nutrition' column contains multiple subcolumns, so the data needs to be 'flattened' or normalized.

```
[ ]: df2 = pd.json_normalize(results)
```

```
[ ]: df2
```

Let's see if we can extract some information from this dataframe. Perhaps, we need to know the family and genus of a cherry.

```
[ ]: cherry = df2.loc[df2["name"] == 'Cherry']
     (cherry.iloc[0]['family']) , (cherry.iloc[0]['genus'])
```

1.7 Exercise 2

In this Exercise, find out how many calories are contained in a banana.

```
[ ]: # Write your code here
```

Click here for the solution

```
cal_banana = df2.loc[df2["name"] == 'Banana']
cal_banana.iloc[0]['nutritions.calories']
```

1.8 Exercise 3

This page contains a list of free public APIs for you to practice. Let us deal with the following example.

Official Joke API This API returns random jokes from a database. The following URL can be used to retrieve 10 random jokes.

https://official-joke-api.appspot.com/jokes/ten

1. Using `requests.get("url")` function, load the data from the URL.

```
[ ]: # Write your code here
```

Click here for the solution

```
data2 = requests.get("https://official-joke-api.appspot.com/jokes/ten")
```

2. Retrieve results using `json.loads()` function.

```
[ ]: # Write your code here
```

Click here for the solution

```
results2 = json.loads(data2.text)
```

3. Convert json data into *pandas* data frame. Drop the type and id columns.

```
[ ]: # Write your code here
```

Click here for the solution

```
df3 = pd.DataFrame(results2)
df3.drop(columns=["type","id"],inplace=True)
df3
```

2 Congratulations! - You have completed the lab

2.1 Author

Svitlana Kramar

Svitlana is a master's degree Data Science and Analytics student at University of Calgary, who enjoys travelling, learning new languages and cultures and loves spreading her passion for Data Science.

2.2 Additional Contributor

Abhishek Gagneja