**Simple APIs**

**Random User and Fruitvice API Examples**

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:

* **Secirity**. 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](https://randomuser.me/documentation?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0101ENSkillsNetwork1005-2022-01-01" \l "intro" \t "_blank) can be found here.

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

## Example 1: RandomUser API

Bellow are Get Methods parameters that we can generate. For more information on the parameters, please visit this [documentation](https://randomuser.me/documentation?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0101ENSkillsNetwork1005-2022-01-01) page.

## ****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

[<randomuser.RandomUser at 0x7f11f89f51d0>,

<randomuser.RandomUser at 0x7f11f89f5210>,

<randomuser.RandomUser at 0x7f11f89f5250>,

<randomuser.RandomUser at 0x7f11f89f5290>,

<randomuser.RandomUser at 0x7f11f89f52d0>,

<randomuser.RandomUser at 0x7f11f89f5350>,

<randomuser.RandomUser at 0x7f11f89f5390>,

<randomuser.RandomUser at 0x7f11f89f53d0>,

<randomuser.RandomUser at 0x7f11f89f5410>,

<randomuser.RandomUser at 0x7f11f89f5310>]

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())

Siegfried Adam siegfried.adam@example.com

Susanne Moulin susanne.moulin@example.com

Aurora Cabrera aurora.cabrera@example.com

Harper Taylor harper.taylor@example.com

Eline Boyer eline.boyer@example.com

اميرمحمد حسینی myrmhmd.hsyny@example.com

Murat Limoncuoğlu murat.limoncuoglu@example.com

Grit Wilkens grit.wilkens@example.com

Barış Avan baris.avan@example.com

Jetty Lindeboom [jetty.lindeboom@example.com](mailto:jetty.lindeboom@example.com)

## Exercise 1

In this Exercise, generate photos of the random 5 users.

*# Write your code here*

for user in some\_list:

print (user.get\_picture())

<https://randomuser.me/api/portraits/men/89.jpg>

<https://randomuser.me/api/portraits/women/79.jpg>

<https://randomuser.me/api/portraits/women/37.jpg>

<https://randomuser.me/api/portraits/women/72.jpg>

<https://randomuser.me/api/portraits/women/55.jpg>

<https://randomuser.me/api/portraits/men/26.jpg>

<https://randomuser.me/api/portraits/men/61.jpg>

<https://randomuser.me/api/portraits/women/78.jpg>

<https://randomuser.me/api/portraits/men/88.jpg>

<https://randomuser.me/api/portraits/women/29.jpg>

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.

## Example 2: Fruitvice 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 [fruitvice](https://www.fruityvice.com/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0101ENSkillsNetwork1005-2022-01-01" \t "_blank) API data using requests.get("url") function. The data is in a json format.

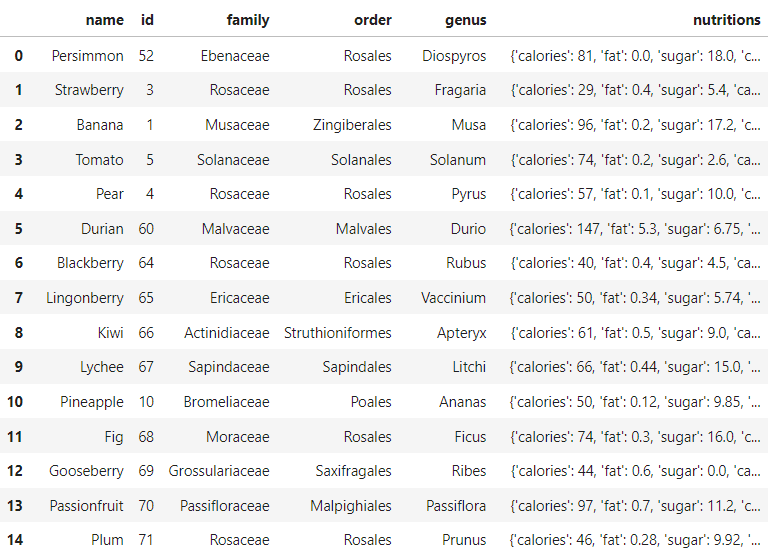
data = requests.get("https://fruityvice.com/api/fruit/all")

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'])

('Rosaceae', 'Prunus')