Python One Cool Cat (Meow Facts) API Tutorial

Contents

Python One Cool Cat (Meow Facts) API Tutorial 1
What is an API? 2
Why Use a Web API? 3
Status Codes 3
- 200 - 4
- 301 - 4
- 400 - 4
- 401 - 4
- 403 - 5
- 404 - 5
Public API's 5
Meow Facts API 5
What is JSON? 5
What is a Dictionary? 6
Dictionary Basics 7
API Response Dictionaries and Lists 7
Tutorial: Meow Facts 7
Install the Python Requests Library 7
Create the Program 8
Assignment Submission 10

Time required: 60 minutes

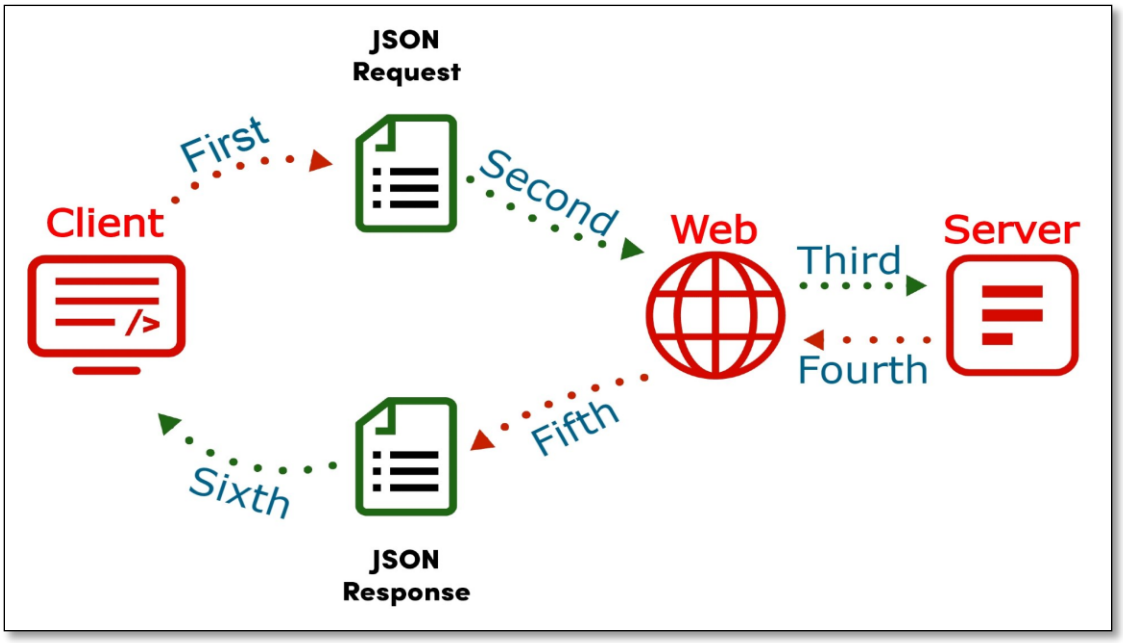
- Comment each line of code as show in the tutorials and other code examples.
- Follow all directions carefully and accurately.
- Think of the directions as minimum requirements.

What is an API?

What is an API? This is what Wikipedia says.

"An **application programming interface** ('API') is a computing interface that defines interactions between multiple software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees. An API can be entirely custom, specific to a component, or designed based on an industry-standard to ensure interoperability. Through information hiding, APIs enable modular programming, allowing users to use the interface independently of the implementation."

An API is a software intermediary that allows two applications to talk to each other. In other words, an API is the messenger that delivers your request to the provider that you're requesting it from and then delivers the response back to you.



API's are hosted on webservers. Instead of a web browser asking for a webpage (like how most interactions with the internet is done today) your program asks for data. This data is usually returned to your program in a JSON format. JSON stands for JavaScript Object Notation and is a lightweight data-interchange format. to get data to our program we make a request to a web server. The webserver then replies with our desired data and a status code. Sometimes an API key is needed to be sent along with the request to successfully return data.

There are many different ways to call information from an API. One of the most common are | **get** | requests. You use a get request each time you browse to a web site.

For example, to request data to a Python script using the extra functionality gained from the request library the statement below would be used.

```
response = requests.get('http://api_example.com/example.json')
```

This data object can be named anything, response was used as the identifier for this example. This API doesn't exist. This is the format they tend to take.

Data from the web can tell us the exact weather situation of any place on the globe, the location of faces on an image we provide or the number of astronauts currently floating far, far out in space. Any information freely accessible on the World Wide Web is literally within our grasp from within our Python programs.

Why Use a Web API?

You could use static data which you have downloaded or created yourself. Data used this way with Python would work perfectly fine. Here are some examples of when APIs are more useful than static data sets.

- When the data is changing quickly, and you need the data to be accurate and up to date. Planes are currently in the air or the location of the international space station, the data is constantly changing.
- When only a small piece of a much larger data set is desired.
- When you are taking advantage of already completed, repeated computations. Sources such Facebook, Spotify and Google already have a ton of data, which have undergone serious calculations, which you can access easily with an API request.

In cases like the ones above, an API is the right solution. Using an API will save time and effort over doing all the computation ourselves or constantly refreshing our desired data.

Status Codes

Status codes are shorthand explanations of what happened when you requested for data and are very useful when it comes to problem solving. After requesting an API there are a number of situations that can occur. Each situation has a three-digit numerical status code associated with it. The web server will return status codes to your computing device every single time they receive an API request. This allows for robust error handling.

To see the status code of your request, type this line into your code where the placeholder text | response | is the user-defined variable name decided by you in your script line when data from the API is requested.

```
print(response.status_code)
```

All the potential responses that could be printed to the Python IDLE Shell can be seen below. An example of asking for the status code in script can be seen in the JokeAPI Script.

- 200 -

This means that everything worked, the resource was requested, and the desired data was returned successfully. It will most likely be returned to python as a JSON file. JSON file (longhand is JavaScript Object Notation format) is a lightweight method to store simple data structures and objects. It is primarily used for data-interchange between a web application and a server. This is because when exchanging data between a browser and a server, the data can only be text.

This is the code that is most often used to verify the API request. If any other code is returned, the request is considered in error.

- 301 -

This means, after the resource was requested, the server redirected your request to a different endpoint. This can happen when an organization switches domain names or an endpoint name is changed. The data may or may not get returned successfully.

- 400 -

This means, after the resource was requested, the server stated you made a bad request. When requesting something you must send information to gain access to the particular data you desire. This status code appears when you do not send along the right data and you would not have the data you desire.

- 401 -

This means, after the resource was requested, the server stated that you are not authenticated. When requesting something you must send authentication information to gain access to the particular data you desire. This status code appears when you do not send the right credentials to access an API. In this case you would not have the data you desire.

- 403 -

This means, after the resource was requested, the server stated you do not have the right permissions to see it. Some data online is locked meaning access to them is forbidden. In this case you would not have the data you desire.

- 404 -

This means, after the resource was requested, the server stated that the resource you attempted to access was not found on the server. Perhaps the data has been moved, deleted or was not there to begin with. In this case you would not have the data you desire.

Public API's

There are hundreds of public API's available. This website contains a list of some of them.

https://github.com/public-apis/public-apis

Meow Facts API

Meow Facts is a simple API with only item returned, a random Meow Fact.

We are going to create a Python program to get a Meow Fact and display it.

Before we do that, we have a few new concepts to go through.

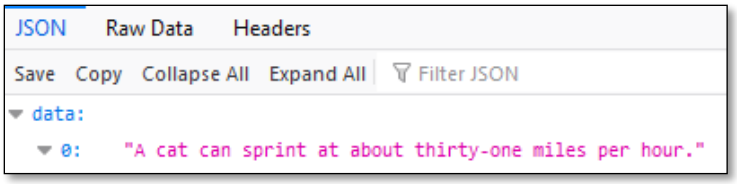
What is JSON?

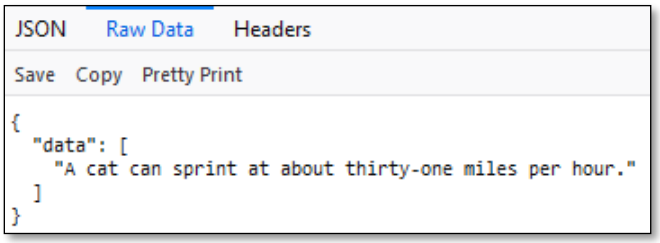
The Meow Facts is a free https-based joke API which provides a JSON response.

Raw API data is typically in a text file in JSON or XML format. For our API projects, we will use JSON.

JavaScript Object Notation (JSON) is an open standard file and data interchange format. It uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types.

Go to https://meowfacts.herokuapp.com/ to see some raw JSON data in two different formats.





What is a Dictionary?

A dictionary is a list of items. Dictionaries are mutable, they can be changed.

Here is a Python list that contains the number of days in the months of the year.

```
days = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
```

If we want the number of days in January, use `days[0]`. December is `days[11]` or `days[-1]`.

Here is a dictionary of the days in the months of the year:

```
days = {'January':31, 'February':28, 'March':31, 'April':30, 'May':31,
        'June':30, 'July':31, 'August':31, 'September':30, 'October':31,
        'November':30, 'December':31}
```

To get the number of days in January, we use `days['January']`.

One benefit of using dictionaries is the code is more readable. We don't have to figure out which index in the list a given month is at.

Unlike other sequences, items in dictionaries are unordered. The first item in a list named spam would be `spam[0]`. There is no "first" item in a dictionary. The dictionary does remember the order of entry. While the order of items matters for determining whether two lists or tuples are the same, it does not matter in what order the key-value pairs are typed in a dictionary.

Dictionary Basics

Parameter	Details
`key`	The desired key to lookup
`value`	The value to set or return

API Response Dictionaries and Lists

Many API responses are combinations of dictionaries and lists. MeowFacts is an example of a dictionary with a key and a list for a value.

```
# Dictionary with the key "data"
# The value is a list [0]
meow_facts = {
    # Inside the key value is a single item list, [0]
    "data": [
        "A cat can sprint at about thirty-one miles per hour."
    ]
}
# Access the API dictionary response
# with list inside the dictionary
meow_fact = meow_facts.get("data")[0]

print(meow_fact)
```

Tutorial: Meow Facts

Install the Python Requests Library

To retrieve information from API's, we expand the functionality of default Python. Requests is a Python Library that lets you send HTTP/1.1 requests, add headers, form data, multipart files, and parameters with Python dictionaries. It lets you access the response data in the same way.

Requests are synchronous. Only one HTTP call can be made at a time.

1. In Windows → Click the Start button → type **cmd** → Click **Command Prompt**
2. Type the following command.

```
# Windows
pip install requests
# Linux
pip3 install requests
# Mac
python3 -m ensurepip
```

https://www.groovypost.com/howto/install-pip-on-a-mac/

The command should either install requests or confirm that it is already installed.

Create the Program

Create the following program named **meow_facts.py** which demonstrates how to access a Web API data with Python.

Right Click the URL below, Click Copy Link location. Paste this into your program to avoid typo's.

https://meowfacts.herokuapp.com/

```python
"""
    Name: meowfacts_cli.py
    Author: William A Loring
    Created: 04/18/21
    Purpose: Get random cat facts from https://meowfacts.herokuapp.com
"""

# Install requests module
# pip install requests
# Import requests module
import requests

# URL for mewfacts API
URL = "https://meowfacts.herokuapp.com/"

# Get requests object from URL
response = requests.get(URL)
# Print response for troubleshooting
# Comment out this line when the program is working
print(response)

# Convert requests json requests object to Python dictionary
meowfacts = response.json()

# Comment out this line when the program is working
print()
# Print dictionary for troubleshooting
# Comment out this line when the program is working
print(meowfacts)

# Print just the data using dictionary created from API Json
print()
print(meowfacts.get("data")[0])
```

Example run (the cat ASCII art is optional):

```
  ^   ^
 " o o "
 ===X===
  ' " '       _
 /''''  \___/  __/
|          /
("|")__\   |
"" ""(_____/
<Response [200]>

{'data': ['Adult cats only meow to communicate with humans.']}

Adult cats only meow to communicate with humans.

>^.^<
```

This is a very simple example of an API call. There isn't any error handling or debugging. We will add that to other API programs.

Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.