**What Is An API?**

**Done**

**API** stands for “application programming interface.” APIs allow one application to “talk-to” another to allow for the transfer of stored data or instructions. API’s are everywhere in your daily life, from the embedded YouTube videos on social media applications to the smart speaker on your kitchen counter. They make it possible for software developed at different times by different people to interact and, thereby, increase functionality and productivity.

**How Does An API Work?**

For the following example, we will be working in JavaScript (jQuery enabled) with [The One API](https://www.artists.bandsintown.com/support/api-installation) (a fun Lord of the Rings Themed API). Consult the API documentation for specific instructions on usage. Many APIs require an authenticating id token to access. The instructions for acquiring that token are also found within the API’s documentation.

*In order to call an API, the “caller” has to send unique identifying information as an identifier to the API service provider. This identifier is called an API key.*

Consider the following code:

*The URL string below is the path of your API and can be found in API documentation.*

const lotrQuote = "https://the-one-api.dev/v2/quote/";

$.ajax({

url: lotrQuote,

method: "GET"

}).then(function(response){

console.log(response);

})

In this example, the variable lotrQuote is defined as a string that contains an API route. When called by the jQuery .ajax method in line 2, the API is queried and returns either JSON or XML data. That data is then passed into our .then callback function and rendered to the console in its raw form.

**Examples Of APIs**

There are tons of APIs available to the public that are free to use. Check out sites like [public-apis.xyz](https://public-apis.xyz/) to explore and get creative with the wide variety of options. Here are a few examples of common public APIs that you may already be familiar with in your day-to-day life:

-[Google’s public API suite](https://developers.google.com/apis-explorer) - Google has an overwhelming collection of public APIs that developers can try for free. They do have pretty strict security on their id keys and free usage is capped by a daily limit so test sparingly. This suite facilitates all the embedded “Google Maps” you see on other developers’ apps and websites. And that’s just the tip of the iceberg, so dive in.

-[Mailchimp’s Marketing API](https://mailchimp.com/developer/api/marketing/) - Mailchimp is an email marketing platform that powers the email marketing needs of many common websites. These sites and apps are able to make use of Mailchimp’s platform by utilizing the power of their marketing API.

-[OpenWeather API - OpenWeather](https://openweathermap.org/api)is a free, open weather data API that can be used to power anything from small widgets to whole applications. According to the reporting site [Built With](https://trends.builtwith.com/websitelist/OpenWeatherMap) , many news organizations use OpenWeather on their sites.

### What Is An API

API : application programming interface is an interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services.

Web API is an API over the web which can be **accessed using HTTP protocol**

APIs are hosted on web servers. When you type www.google.com in your browser’s address bar, your computer is actually asking the www.google.com server for a webpage, which it then returns to your browser.

APIs work much the same way, except instead of your web browser asking for a webpage, your program asks for data. This data is usually returned in **JSON** format.

### The Request Library

In order to get the data, we make a request to a webserver. The server then replies with our data. In Python, we’ll use the requests library to do this.

The [requests](http://docs.python-requests.org/en/master/) library is used to make HTTP requests in Python.

First you have to install the module:

pip install requests

Then you have to import it in your file

import requests

### Working With An API

We are going to use the [Open Notify Api](http://open-notify.org/) : “**Open Notify** is an open source project to provide a simple programming interface for some of NASA’s awesome data”

#### The GET Request

The GET method indicates that you’re trying to get or retrieve data from a specified resource. To make a GET request, invoke requests.get().

requests.get('api\_link')

response = requests.get("http://api.open-notify.org/iss-now.json")

#### The Response

A Response is a powerful object for inspecting the results of the request.

**Status Code**

response = requests.get("http://api.open-notify.org/iss-now.json")

# Print the status code of the response.

print(response.status\_code)

# 200

**Content**

The response of a GET request often has some valuable information.

response = requests.get("http://api.open-notify.org/iss-now.json")

# Print the content of the response.

print(response.text)

# {"message": "success", "iss\_position": {"longitude": "-32.3037", "latitude": "-12.7147"}, "timestamp": 1588684375}

You see that the response is serialized JSON content. To convert the data into a Python dictionary, you could use [json.loads()](https://realpython.com/python-json/" \l "deserializing-json" \t "_blank). However, a simpler way to accomplish this task is to use .json():

response = requests.get("http://api.open-notify.org/iss-now.json")

# Print the content of the response in Python

print(response.json())

# {'message': 'success', 'iss\_position': {'longitude': '-29.3427', 'latitude': '-16.5135'}, 'timestamp': 1588684451}

**Headers**

The response headers can give you useful information, such as the content type.

You can use response.header: returns a dictionary-like object, allowing you to access header values by key

response = requests.get("http://api.open-notify.org/iss-now.json")

# Print the header of the response

print(response.header)

# {'Server': 'nginx/1.10.3', 'Date': 'Tue, 05 May 2020 13:16:04 GMT', 'Content-Type': 'application/json', 'Content-Length': '114', 'Connection': 'keep-alive', 'access-control-allow-origin': '\*'}

#### Query String Parameters

By doing a GET request you can pass values through query string parameters in the URL. To do this using get(), you pass data to the optional keyword argument params.

In the **Open Notify** , you can add the latitude and the longitude of a country to a GET request : ” Given a location on Earth (latitude, longitude, and altitude) this API will compute the next n number of times that the ISS will be overhead.”

Let’s use the latitude and the longitude of Israel :

It’s the same as doing : http://api.open-notify.org/iss-pass.json?lat=31.771959&lon=35.217018

parameters = {"lat": 31.771959, "lon": 35.217018}

response = requests.get("http://api.open-notify.org/iss-pass.json", params=parameters)

print(response.text)

# The JSON response

{

"message": "success",

"request": {

"altitude": 100,

"datetime": 1588685144,

"latitude": 31.771959,

"longitude": 35.217018,

"passes": 5

},

"response": [

{

"duration": 417,

"risetime": 1588709443

},

{

"duration": 653,

"risetime": 1588715100

},

{

"duration": 528,

"risetime": 1588720995

},

{

"duration": 220,

"risetime": 1588727046

},

{

"duration": 262,

"risetime": 1588732953

}

]

}

You can use this data :

parameters = {"lat": 31.771959, "lon": 35.217018}

# Make a get request with the parameters.

response = requests.get("http://api.open-notify.org/iss-pass.json", params=parameters)

# Print the content of the response (the data the server returned)

data = response.json()

print(type(data))

# >> <class 'dict'>

print(data["response"][0])

# >> {'duration': 417, 'risetime': 1588709443}

You can also find the number of people currently in space : Just follow the [Documentation](http://open-notify.org/Open-Notify-API/People-In-Space/)

response = requests.get("http://api.open-notify.org/astros.json")

# Print the content of the response (the data the server returned)

data = response.json()

print(data)

# {"number": 3, "message": "success", "people": [{"craft": "ISS", "name": "Chris Cassidy"}, {"craft": "ISS", "name": "Anatoly Ivanishin"}, {"craft": "ISS", "name": "Ivan Vagner"}]}

#### Inspecting Your Request

You can use the request key

response = requests.get("http://api.open-notify.org/astros.json")

data = response.text

print(response.request.url)

print(response.request.body)

# http://api.open-notify.org/astros.json

# None

#### Exercise

* Use the Chuck Norris API https://api.chucknorris.io/ to retrieve some jokes in a specific category
* Use every notion described in the lesson