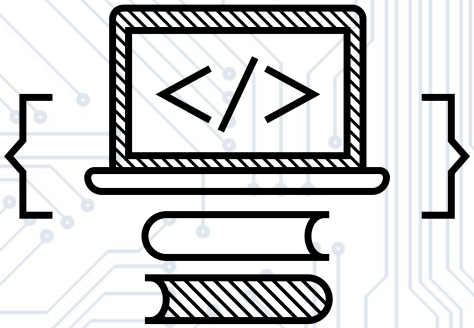
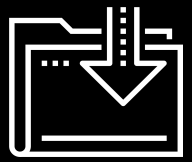


Unit 6 Day 1 - Python APIs

Web Development

Lesson 6.1



Class Objectives

By the end of today's class you will be able to:

Make GET requests with Python requests library

Manipulate JSON response to retrieve necessary values

Store JSON responses in Python lists and dictionaries

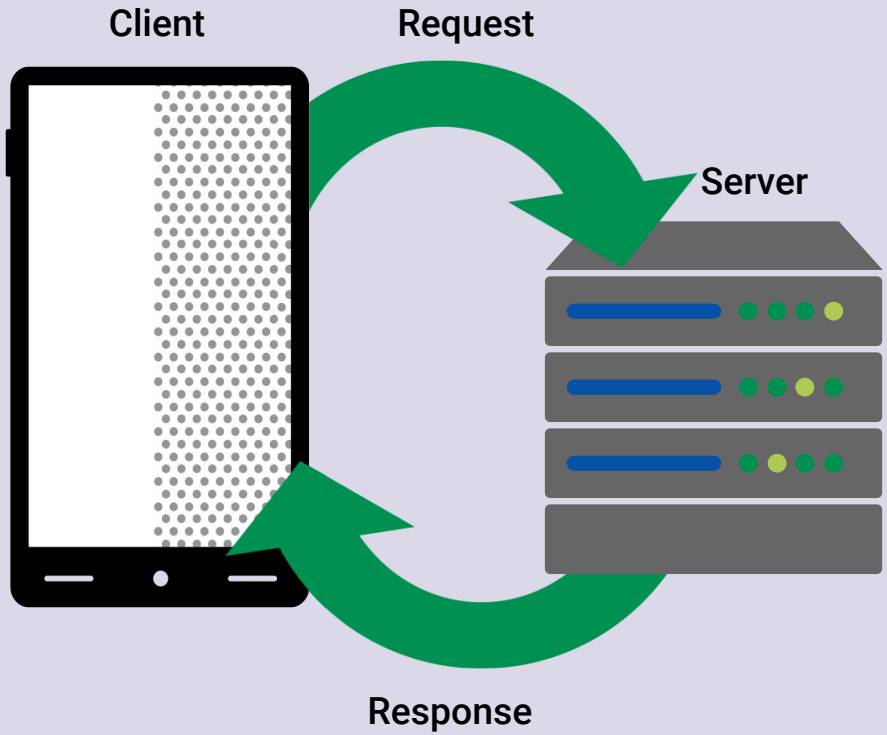
Identify and generate the type of request needed to request movies by leveraging the OMDB API documentation

What is a client versus a server?

- A client is an application/device that asks for information.
- A server is an application/device that supplies information to the client
- Analogies: Netflix and Roku

What is an API?

Application Programing Interface

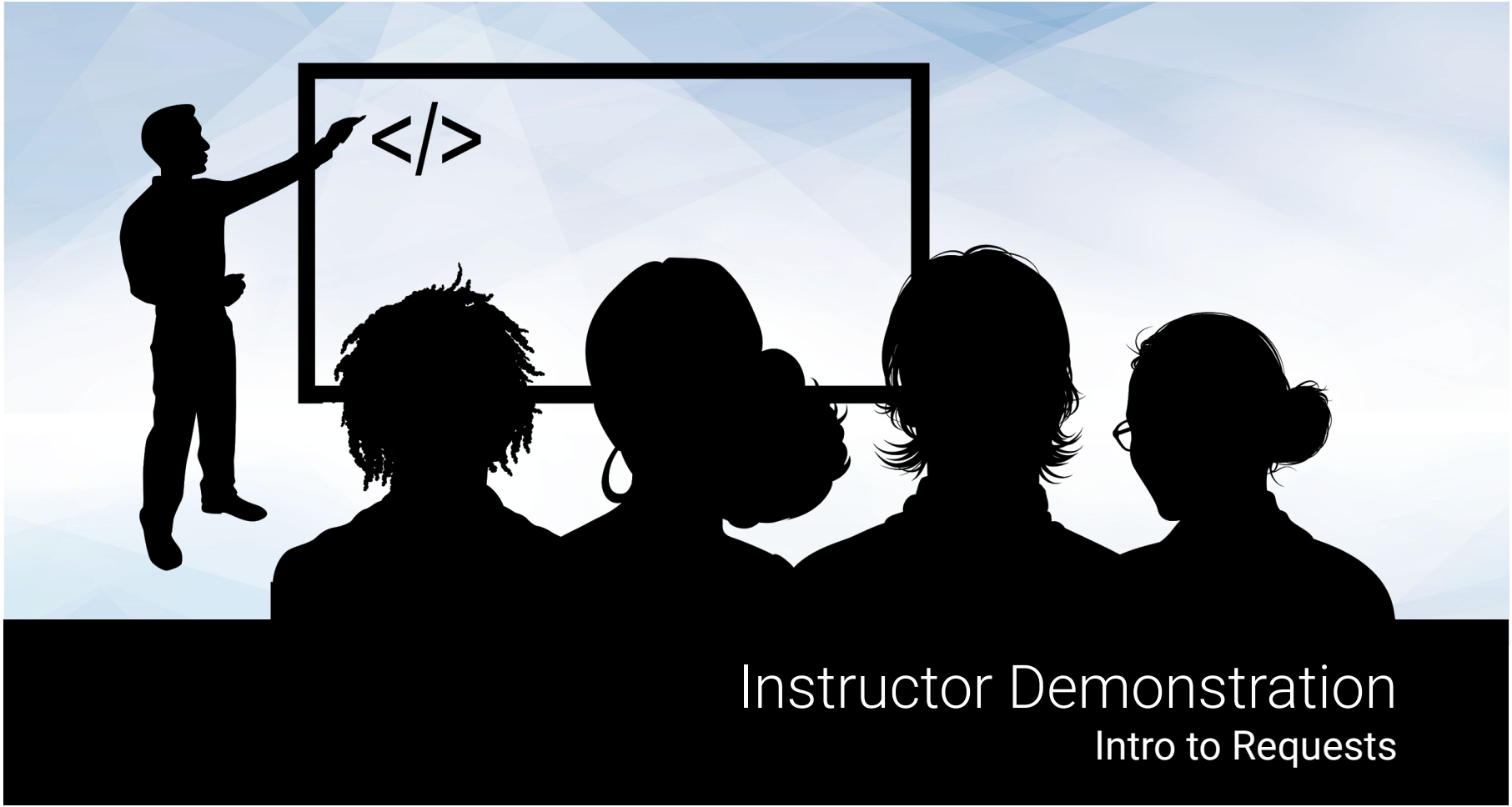


- A request is a communication to the API to retrieve data.
- API calls are similar to visiting a website manually. They point to a URL and collect some data from the page.

JSON - JavaScript Object Notation

- Simplified, human-readable data interchange format
- A web page response to an API call may be formatted as JSON
- URLs used to communicate with APIs are called endpoints.
- Note that the syntax is similar to Python
    - Lists in [ ]
    - Dicts in { }
    - Numbers
    - Strings

```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio
reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et
cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem
eveniet architecto"
  },
```



Instructor Demonstration

Intro to Requests

Install the *requests* library

01 Activate the virtual environment

```
conda activate PythonData
```

02 Install requests

```
conda install requests
```

You may need to restart your Jupyter notebook...

There are two components to our API request

01 requests.get(url)

- Sends a GET request to the URL passed as a parameter.

```
# Dependencies
import requests
import json
```

```
# URL for GET requests to retrieve vehicle data
url = "https://api.spacexdata.com/v2/launchpads"
```

```
# Print the response object to the console
print(requests.get(url))
```

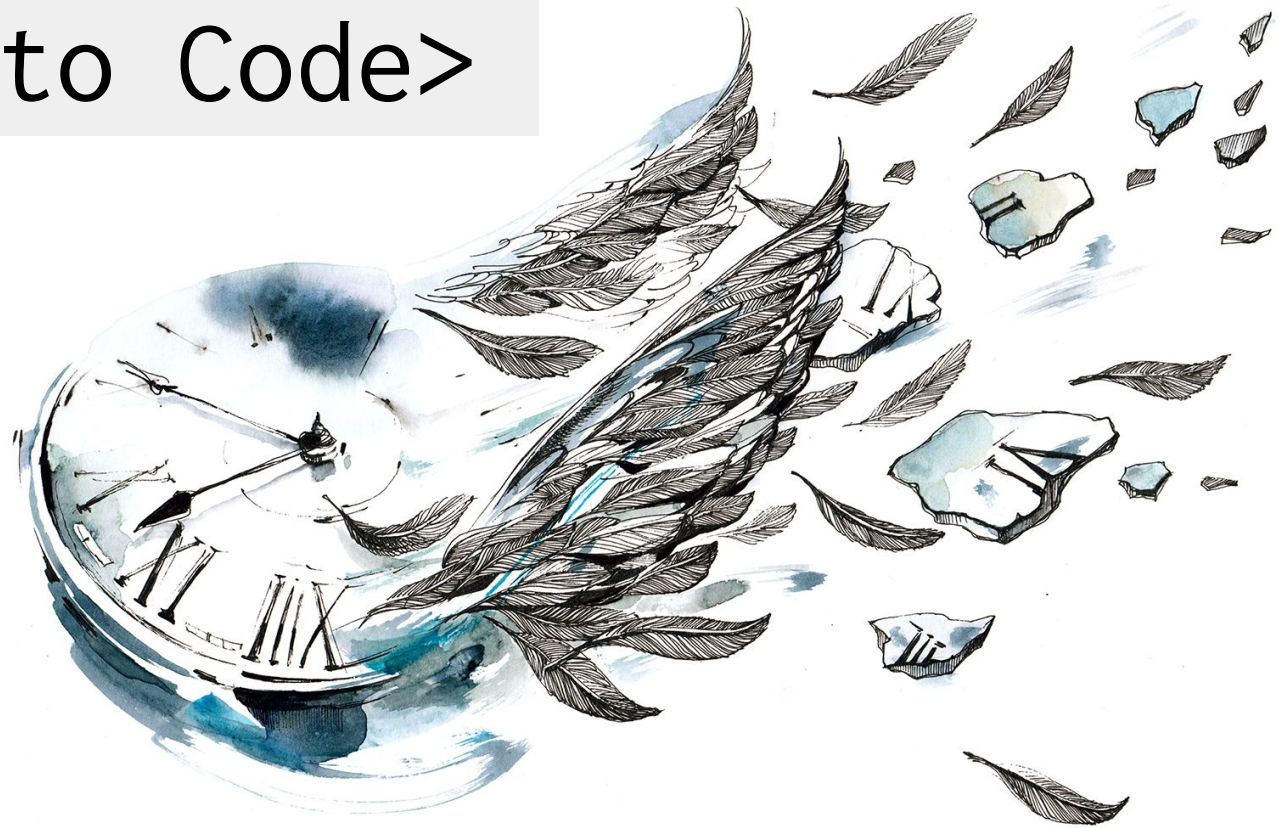
02 .json()

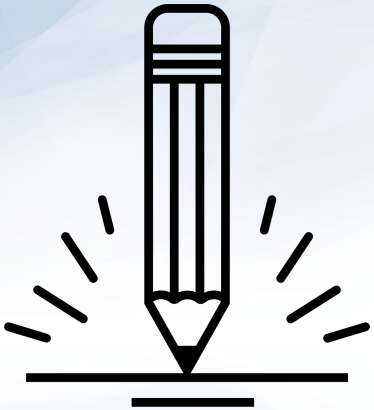
- A call to convert the response object from JSON format into Python objects
- json.dumps() is a method used to convert back to JSON and “pretty print”.

```
# Pretty Print the output of the JSON
response = requests.get(url).json()
print(json.dumps(response, indent=4, sort_keys=True))
```


```
[
    {
        "details": "SpaceX primary Falcon 9 launch pad, where all east coast Falcon 9s launch
ed prior to the AMOS-6 anomaly. Initially used to launch Titan rockets for Lockheed Martin. H
eavily damaged by the AMOS-6 anomaly with repairs expected to be complete by late summer 201
7.",
        "full_name": "Cape Canaveral Air Force Station Space Launch Complex 40",
        "id": "ccafs_slc_40",
        "location": {
            "latitude": 28.5618571,
            "longitude": -80.577366,
            "name": "Cape Canaveral",
            "region": "Florida"
        },
        "status": "under construction",
        "vehicles_launched": "falcon 9"
    },
    {
        "details": "SpaceX new launch site currently under construction to help keep up with
the Falcon 9 and Heavy manifests. Expected to be completed in late 2018. Initially will be li
```

<Time to Code>





Activity: Requesting SpaceX

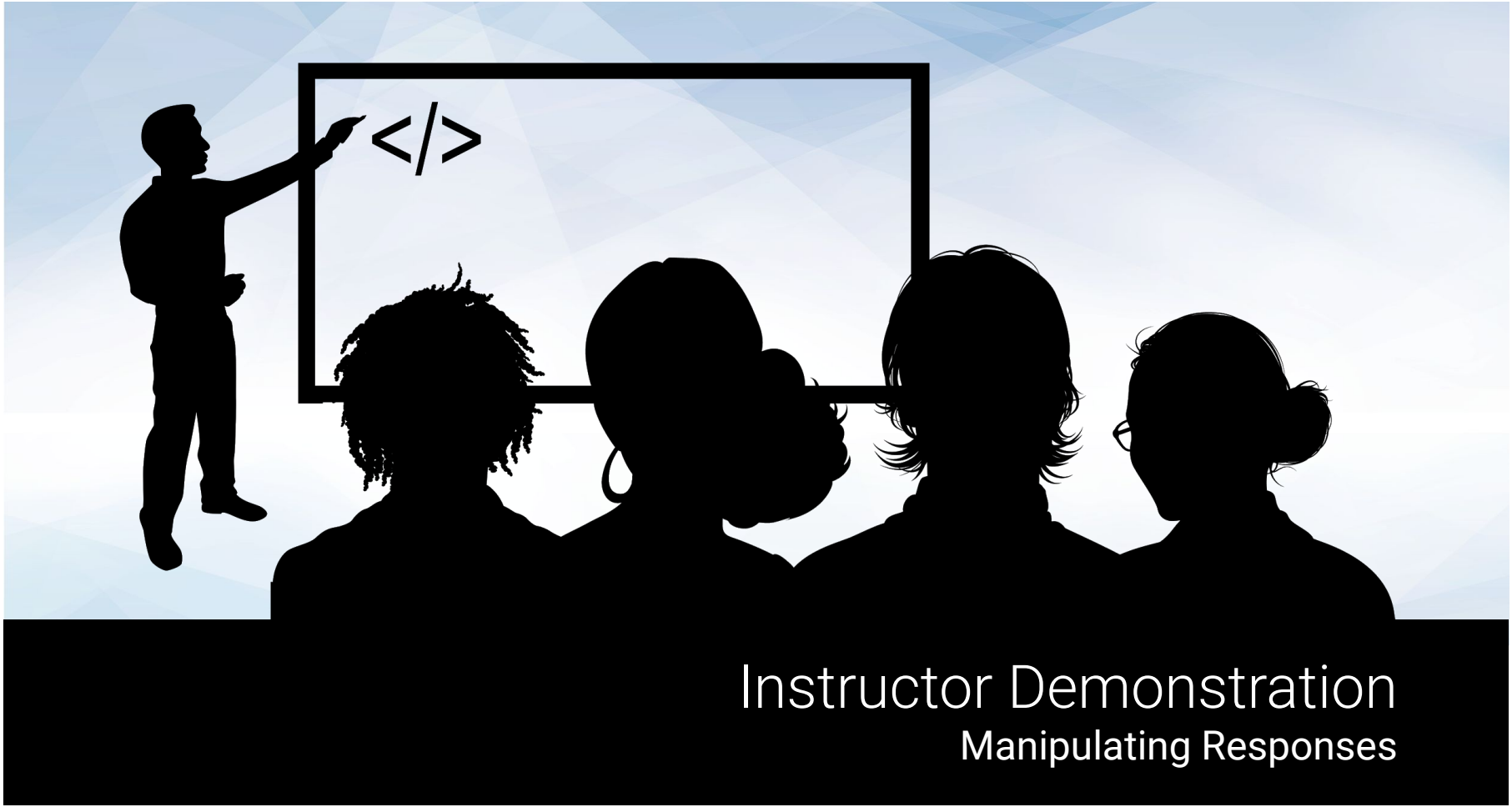
In this activity you will dig into a rather simple and well-documented API - The SpaceX API - and make calls to the API using the Requests library.

Suggested Time:
10 Minutes

Requesting SpaceX Instructions

- **Make sure you have *requests* installed!!**
- Take a few minutes to explore the SpaceX V3 API:
  - [GitHub](https://github.com/r-spacex/SpaceX-API)
  - [API Documentation](http://bit.ly/SpaceXAPI)
- Once you understand the structure of the API and its endpoint, choose one of the endpoints and do the following:
  - Retrieve and print the JSON for *all* of the records from your chosen endpoint.
  - Retrieve and print the JSON for the a *specific* record from your chosen endpoint.

Time’s Up! Let’s Review.



Instructor Demonstration

Manipulating Responses

Working with JSON Responses

01 Simple Method

Use `requests.get()`, store the output, and print the JSON response

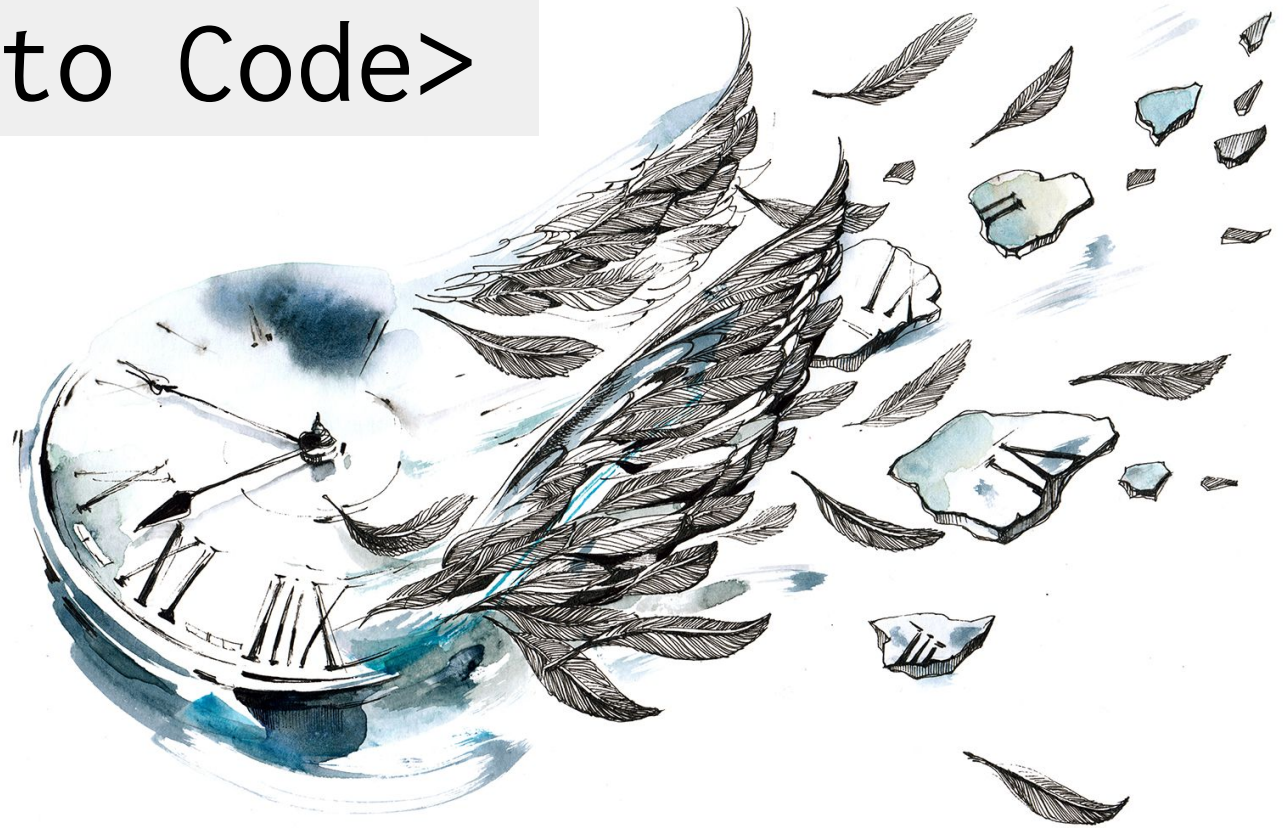
- Must interpret the full JSON object each time
- More difficult to import into pandas
- Less scalable

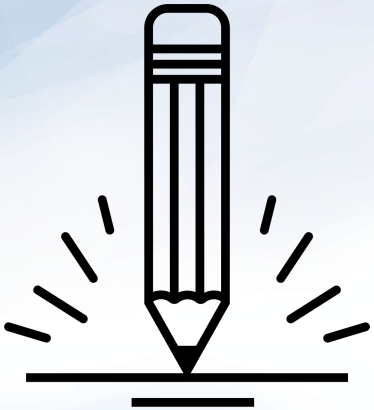
02 Advanced Method

Store the `requests.get()` object, store the `response.json()` and access the JSON object as Python

- Navigate the JSON object like a dictionary
- Easy to import into pandas
- More Scalable

<Time to Code>





Activity: Requesting a Galaxy Far Far Away

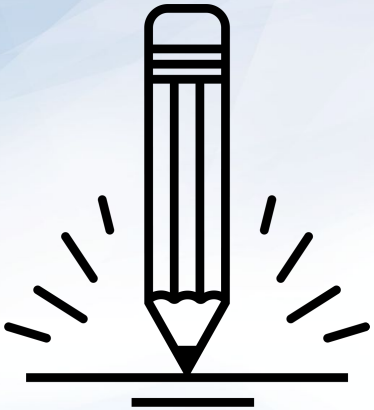
In this activity you will create an application that accesses data from the Star Wars API and prints out values from within it.

Suggested Time:
15 Minutes

Requesting a Galaxy Far Far Away **Instructions**

- Using the starter file provided, collect the following pieces of information from the Star Wars API.
  - The name of the character
  - The number of films they were in
  - The name of their first starship
- Once the data has been collected, print it out to the console.
- **Hints:**
  - It would be in your best interest to print out the JSON from the initial request before anything else. This will let you know what keys you should reference.
  - The "starship" values are links to another API call. This means that you will need to create a request based off of the values of a previous request.
- **Bonus:** Collect and print out all of the films a character appeared in.

Time’s Up! Let’s Review.



Activity: Number Facts

In this activity you and a partner will join forces in creating an interactive application that uses the "numbers" API. The application will take in a number and then return a random fact about that number.

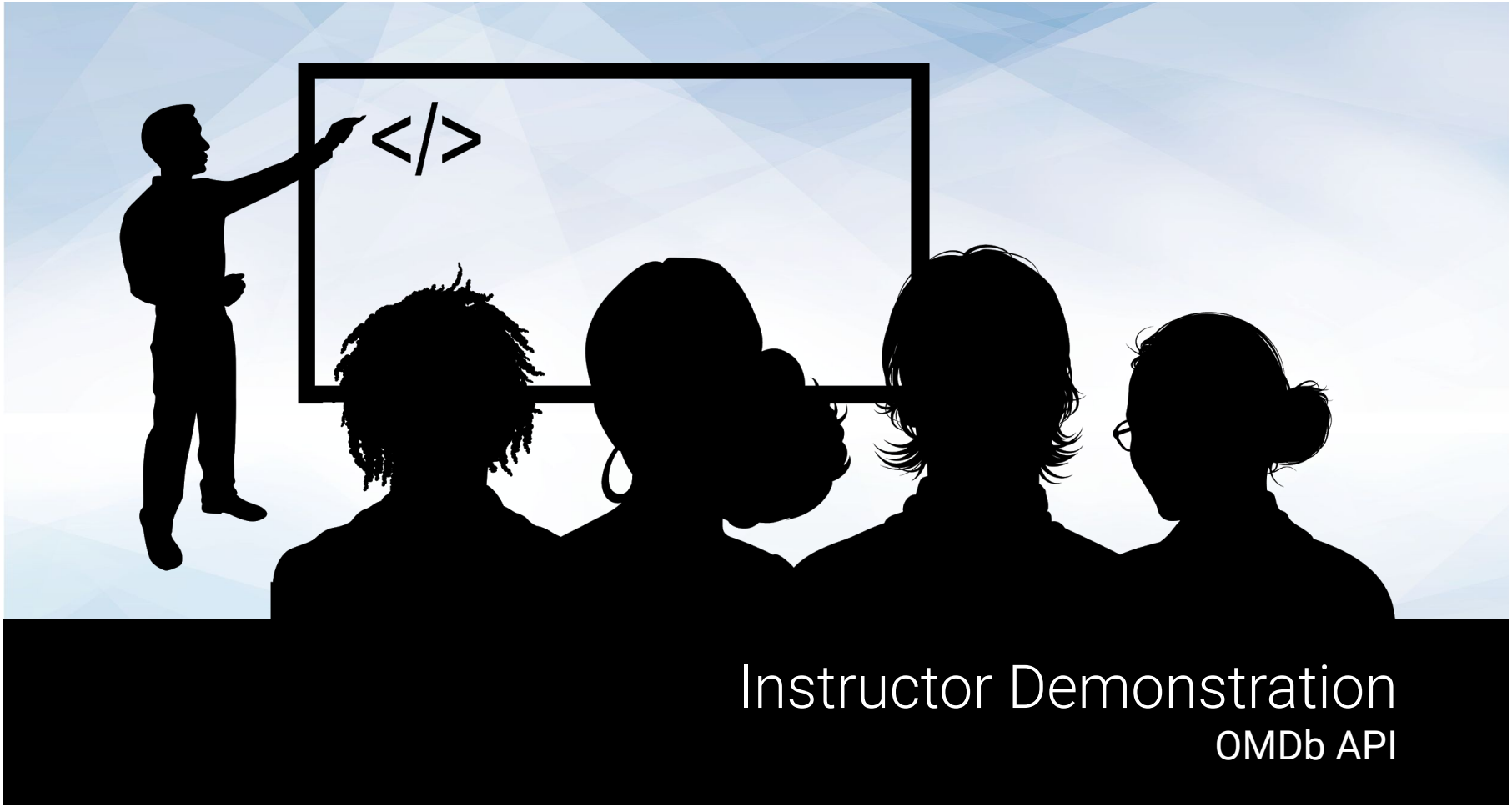
Suggested Time:
20 Minutes

Number Facts **Instructions**

- Using the [Numbers API](http://numbersapi.com), create an application that takes in a user's inputs and returns a number fact based upon it.
- **Hints:**
  - The URL to make your request to must have `?json` at its end so that the data format returned is JSON. The default response is pure text.
  - Make sure to read through the documentation when creating your application. Some endpoints require more or less data than others.

Time’s Up! Let’s Review.

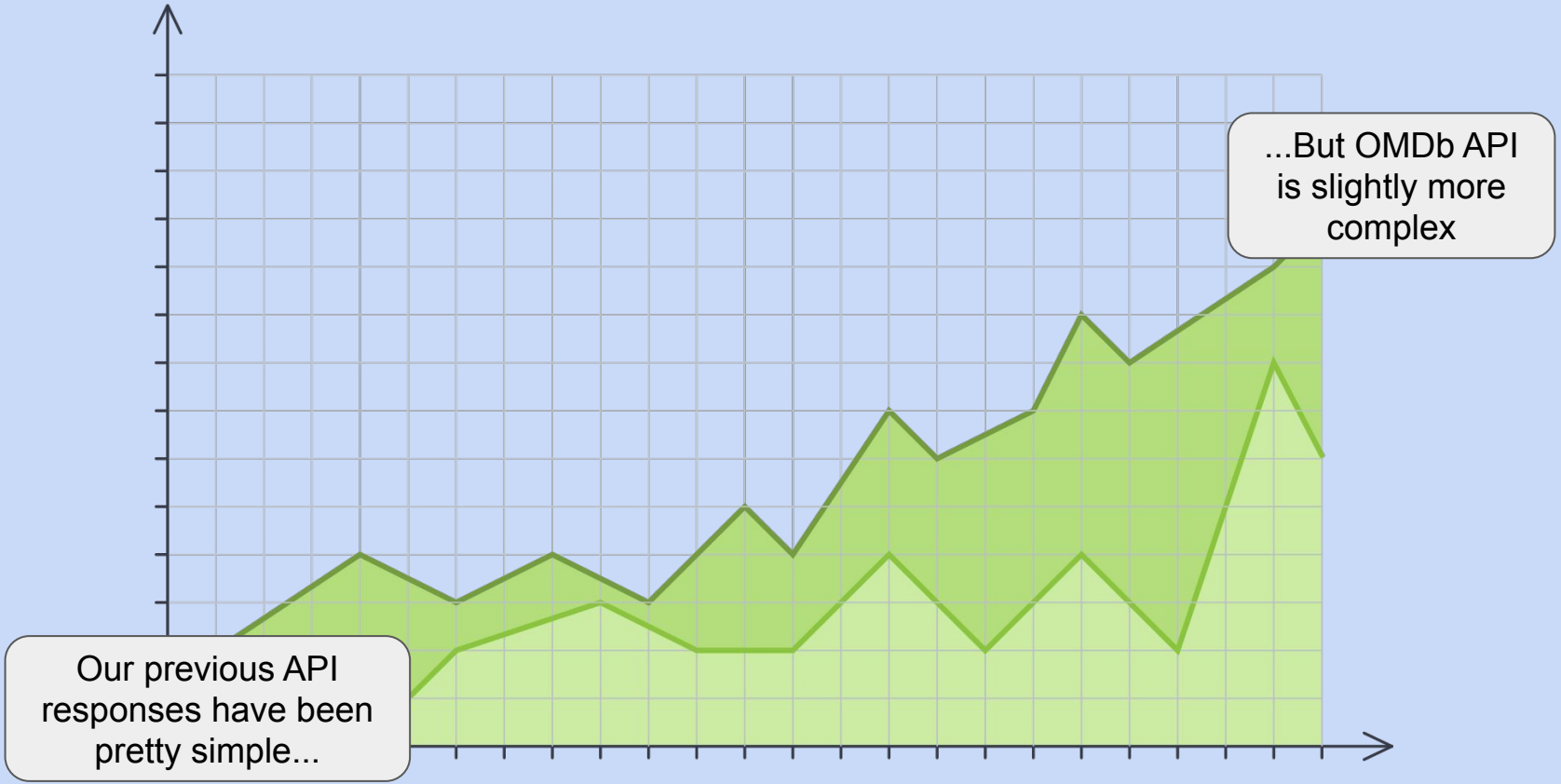
Take a Break!



Instructor Demonstration

OMDb API

JSON responses so far



URL Parameters

01 `?t=`

- Stands for title.
- Example:

  http://www.omdbapi.com/?t=Aliens

  - This is asking the API to return all information on movies with the title “Aliens.”

02 `api_key`

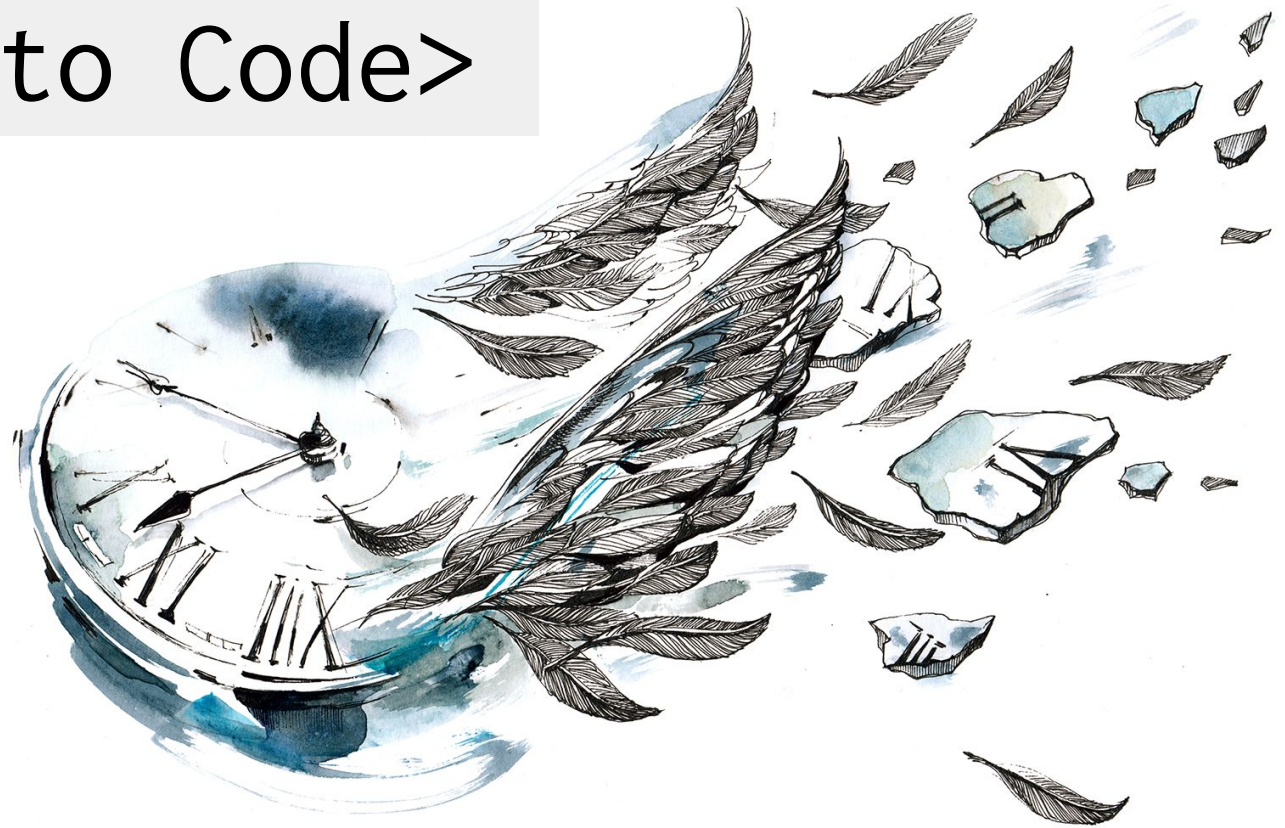
- API Keys restrict API access to specific users
- Without an API key no data would be returned.

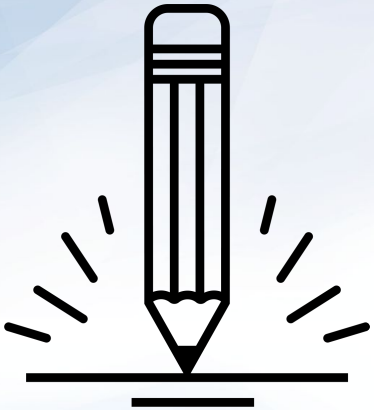
```
# Note that the ?t= is a query param for the t-itle of the
# movie we want to search for.
url = "http://www.omdbapi.com/?t="
api_key = "&apikey=trilogy"
```

```
# Performing a GET request similar to the one we executed
# earlier
response = requests.get(url + "Aliens" + api_key)
print(response.url)
```

```
http://www.omdbapi.com/?t=Aliens&apikey=trilogy
```

<Time to Code>





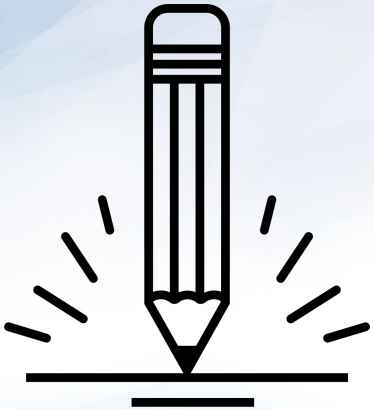
Activity: Study the OMDb API

In this activity, you will take some time reviewing the documentation for OMDb API and testing it out!

Suggested Time:
5 Minutes

Study the OMDb API Instructions

- Read the OMDb documentation, and make a few API calls to get some information about your favorite movie:
  - http://www.omdbapi.com/
- You'll need to register for a free API key
  - Free keys are limited to 1,000 requests/day
  - Free keys can't download poster images



Activity: Movie Questions

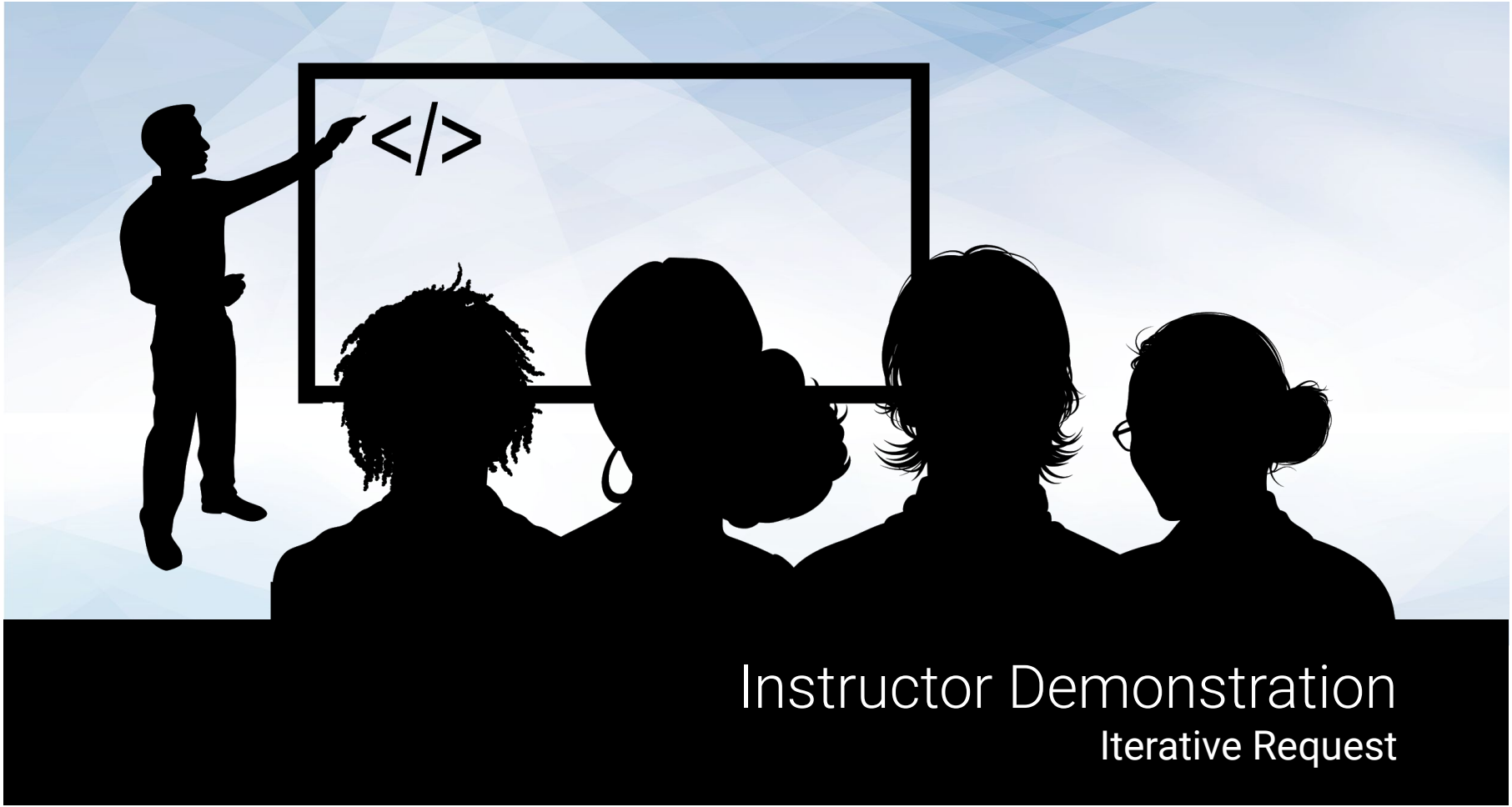
In this activity, you will now test your skills with the OMDB API and collect some data from the API in order to answer a series of questions.

Suggested Time:
20 Minutes

Movie Questions **Instructions**

- Use the OMDb API to retrieve and print the following information.
  - Who was the director of the movie **Aliens**?
  - What was the movie **Gladiator** rated?
  - What year was **50 First Dates** released?
  - Who wrote **Moana**?
  - What was the plot of the movie **Sing**?

Time’s Up! Let’s Review.



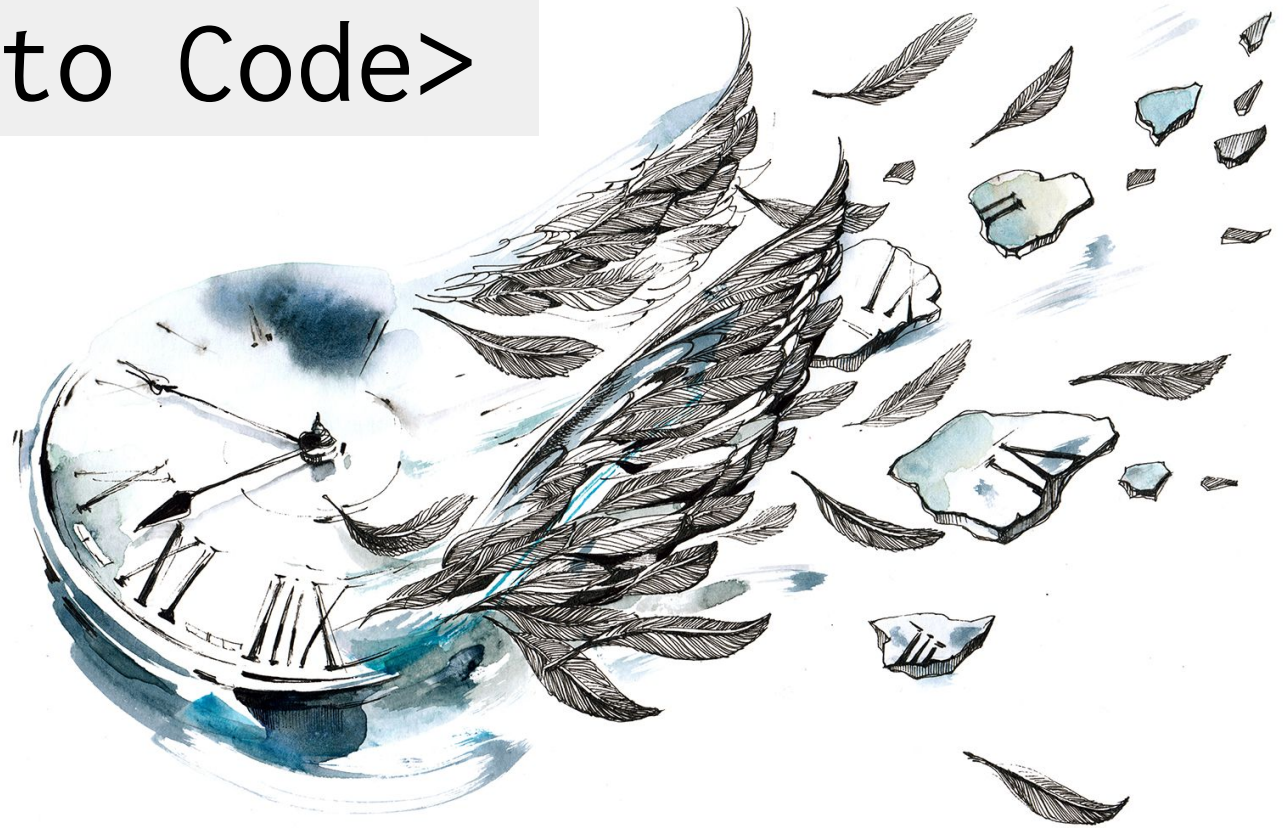
Instructor Demonstration

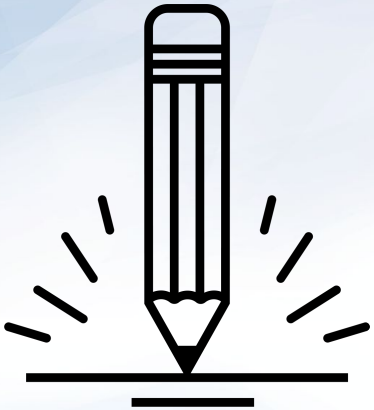
Iterative Request

Our Requests so far

- All information being requested so far has worked from a single request.
- Sometimes, APIs will only respond with some of the information.
  - Some APIs limit the amount of data in response to a single API call.
  - For example, the New York Times API will return only 10 articles at a time. A programmer would have to make 3 requests to retrieve 30 articles.

<Time to Code>





Activity: Iterative Requests

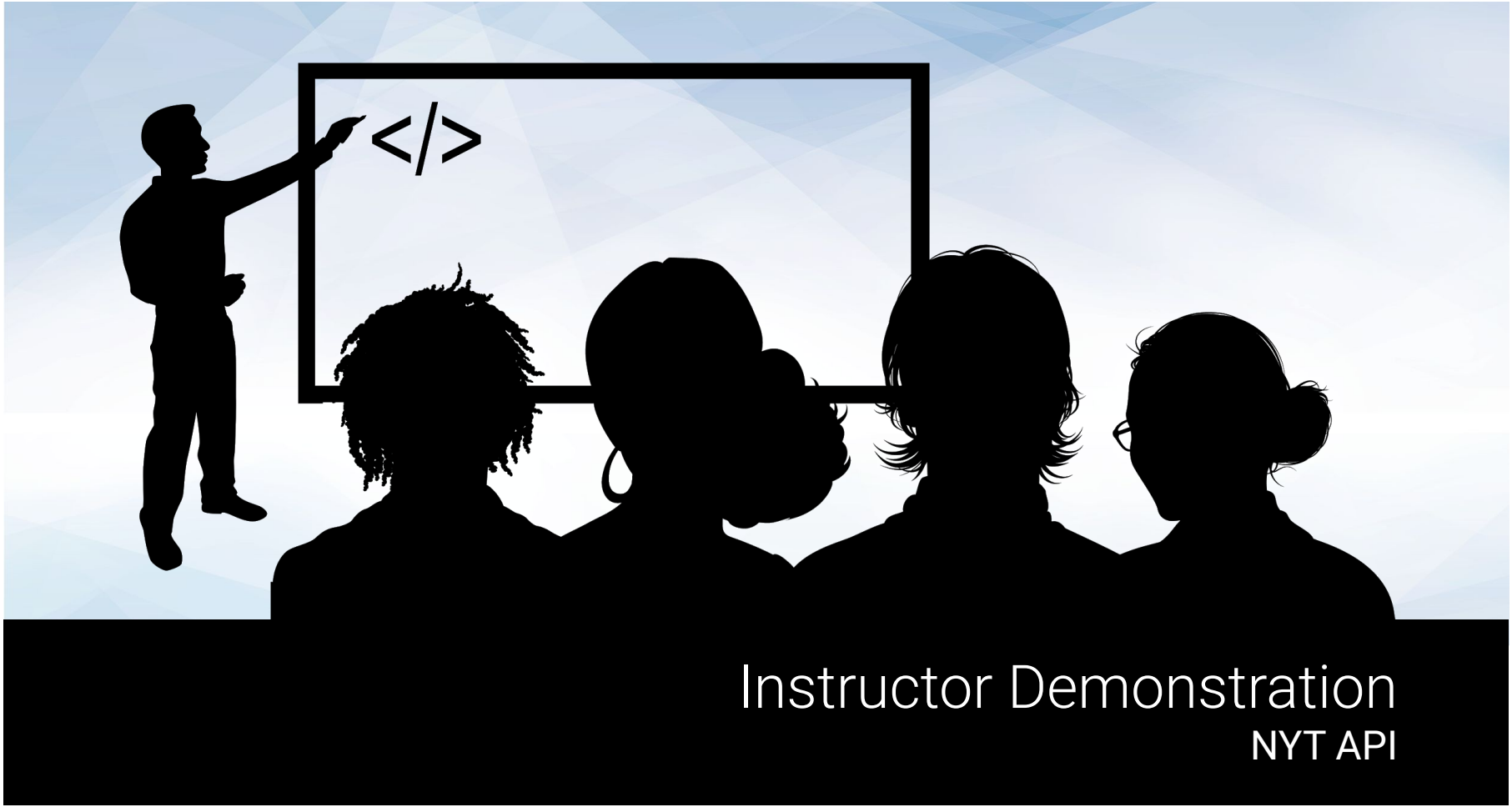
In this activity, you will test your knowledge of iterative requests by looping through a list of movies and collecting data from the OMDB API on each movie.

Suggested Time:
10 Minutes

Movie Questions **Instructions**

- Consider the following list of movie titles:
  - movies = ["Aliens", "Sing", "Moana"]
- Make a request to the OMDb API for each movie in the list. Then:
  - Print the director of each movie
  - Save the responses in another list

Time’s Up! Let’s Review.



Instructor Demonstration

NYT API

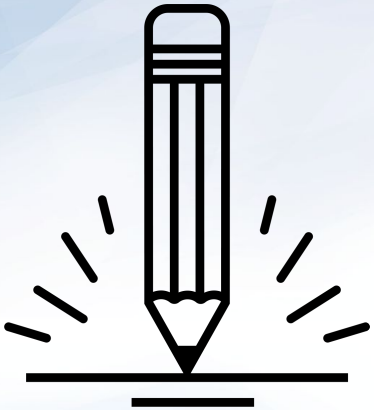
Group activity: NYT API Signup

- This API requires a sign up.
- Open https://developer.nytimes.com in a browser
- Follow the directions…
  - Create a new login (this will be different from your NYT login)
  - Wait for the verification email
  - Log in again
  - Create a new app
  - Enable the “Article Search API”
  - Hit “Create”
  - Make a copy of the App ID, the key, and the secret

NYT API Signup

- When using API keys be sure to store in a config.py file.
- Add the file to your .gitignore file so keys don't get added to a public repo.

<Time to Code>



Activity: Retrieving Articles

In this activity, you will create an application that grabs articles from the NYT API, stores them within a list, and prints snippets of the articles to the screen.

Suggested Time:
20 Minutes

Retrieving Articles **Instructions**

- Save the following to variables in your script:
  - The NYT API endpoint - make sure you include the right query parameter to retrieve JSON data!
  - Your selected search term
- Build your query URL, and save it to a variable.
- Retrieve your list of articles with a GET request.
- Store each article in the response inside of a list.
- Print a `snippet` from each article.

Time’s Up! Let’s Review.

One more thing!

Please sign up for API keys at

https://home.openweathermap.org/users/sign_up

https://api.census.gov/data/key_signup.html