MAKE SCHOOL

APIs

WEB 1.1

Agenda

- Learning Outcomes
- Hook: Hip-Hop Artists API
- Postman & APIs
- **BREAK**
- Using the 'requests' library to make an API call
- Using Flask + requests!
- Wrap-Up

Learning Outcomes

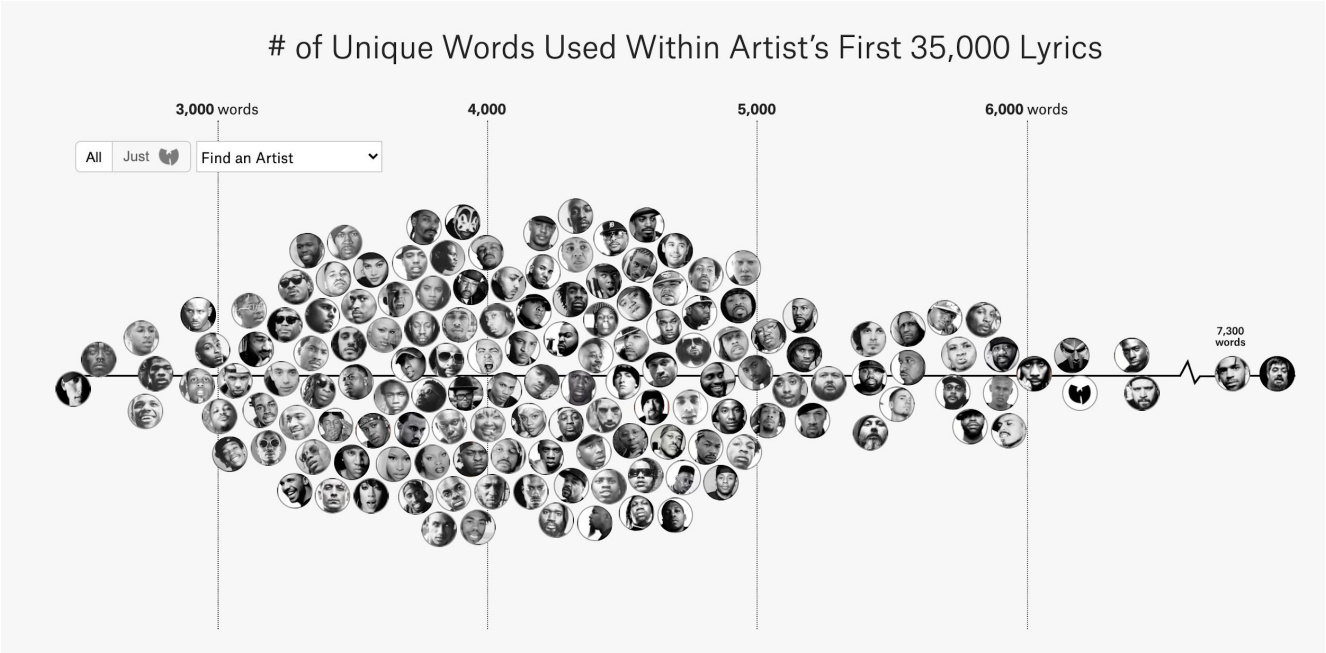
By the end of today, you should be able to...

1. **Explain** how APIs can be useful in retrieving data.
2. **Use** the Postman desktop program to make an API call.
3. **Use** the `requests` library to make an API call within Python code.

Which Hip-Hop artists have the largest vocabulary?

Hip-Hop Song Lyrics

Which Hip-Hop artists have the largest vocabulary? We can use data to answer this question - and it turns out, [someone already has](#)!



Activity (15 minutes)

Read this article and, in a group of 3, answer the following questions:

- How has the number of unique words used by hip-hop artists changed over time?
- At the end of the article, what commentary does Jay-Z provide in his song 'Moment of Clarity' on the phenomenon?
- If you had access to the song lyrics dataset used by the author, what other questions might you want to explore?
    - Vocabulary of top 10 hits of each artist
    - Top 5 most repeated words
    - Chart popularity vs. vocabulary

How do we access this data??

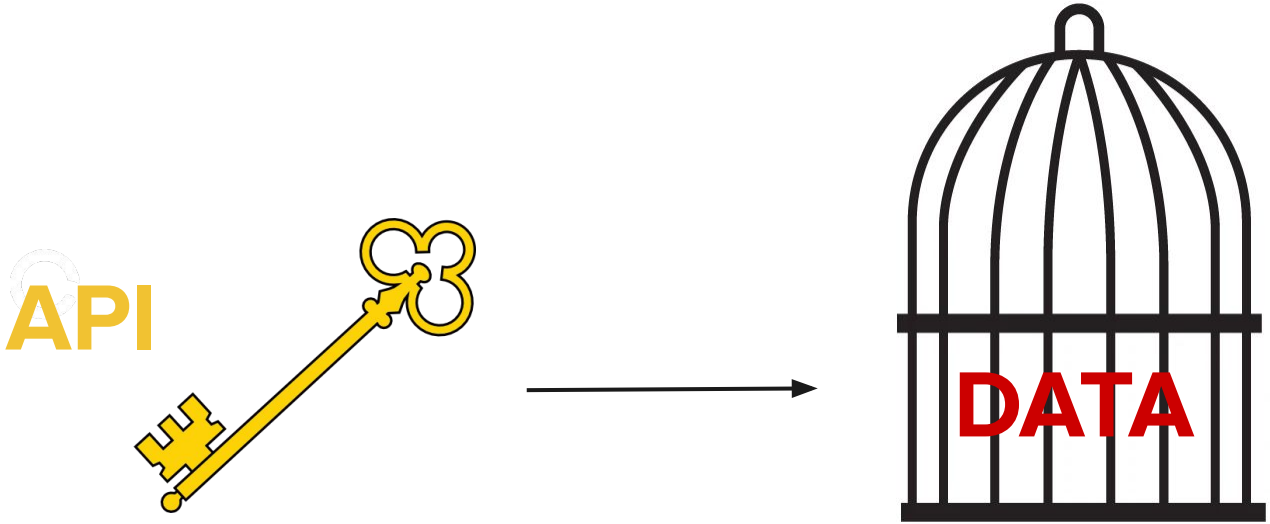
The Genius API

The article's author, Matt Daniels, used the Genius API to gather data for his analysis. The API gave him the song lyrics of each artist, and he wrote a program to count the number of unique words.

So... what is an API?

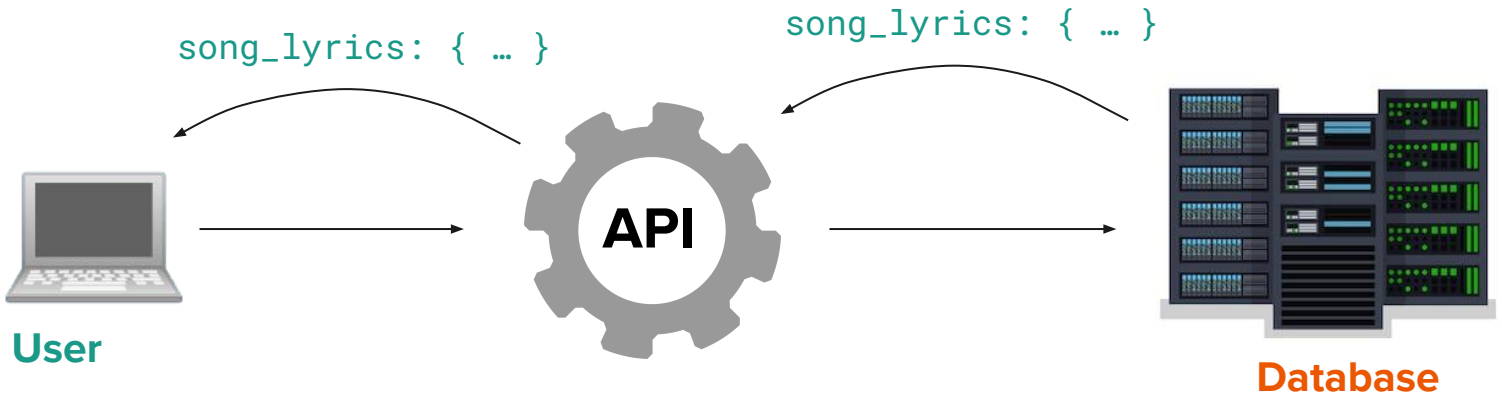
The Genius API

A company, in this case Genius, has a large collection of data that they want to share with the world. However, they can't just give anyone access to their database. So, they created an **API** to allow us to access the data.



APIs

API stands for **Application Programming Interface**. It's the **interface** we use to access someone else's database.

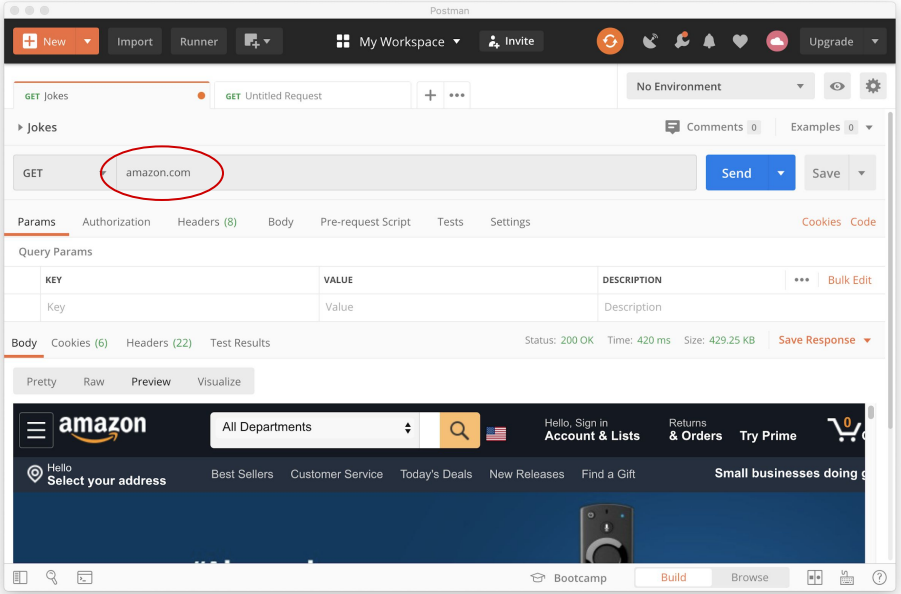


So, now that we know what an API is… how do we access one??

Introducing: Postman

Postman

Postman is a program that lets us make requests to other people's (and our own) APIs. It works just like an Internet browser, with some extra features!

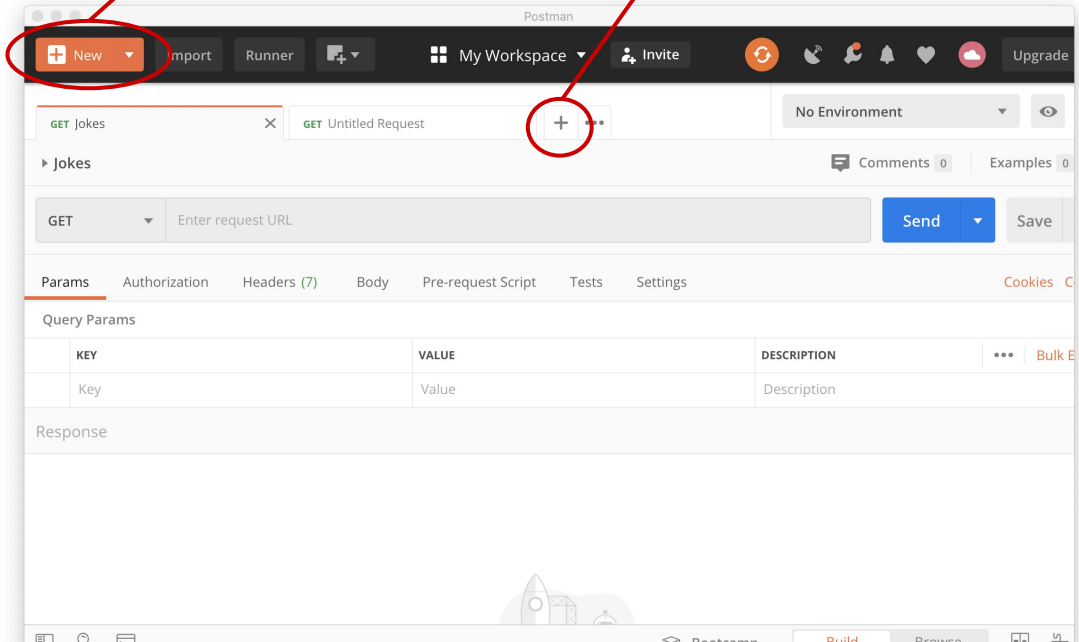


Postman

Click on the “New” or “+” button to make a new request.

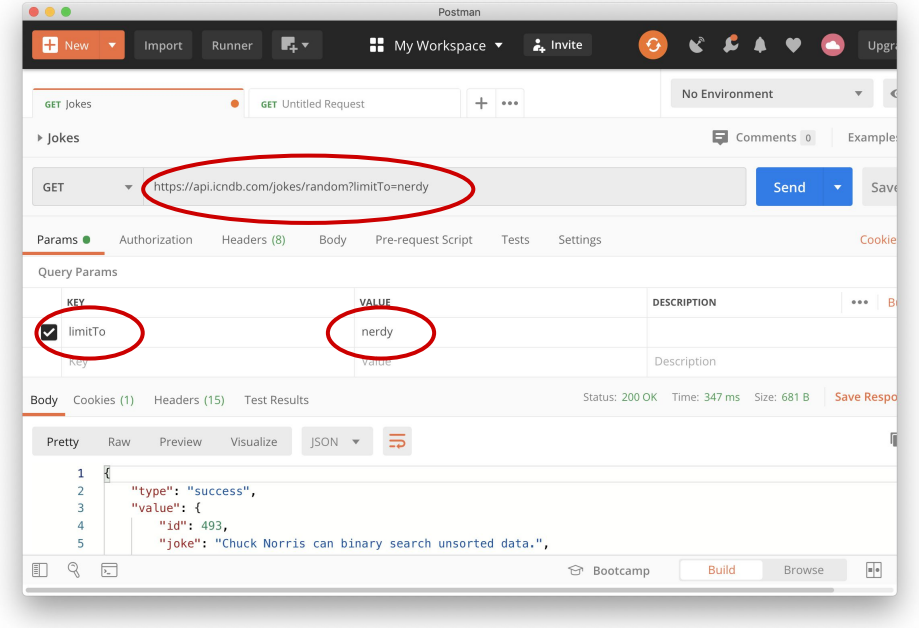
Create a named request

Create an untitled request



Postman

Type the URL into the “Enter request URL” box and press “Send”. Anything after the “?” is a **query parameter** and will be listed below.



Postman

Type in the URL of your request, choose the method (GET, POST, etc), and press “Send”.

Try using the following URL:

```
https://api.icndb.com/jokes/random
```

to get a random Chuck Norris joke!

Postman

We can use a “query parameter” (a type of key-value pair) to limit to only “nerdy” jokes:

```
https://api.icndb.com/jokes/random?limitTo=nerdy
```

Make sure you’re typing into Postman, not your browser!

What joke did you get?

Break - 10 min

How do we make API calls in our code??

Requests Library

We can use the **requests** library to make API calls in our Python code. Here's an example program that uses the Joke API:

```
import requests

params = { "limitTo": "nerdy" } # set the request's query parameters
result = requests.get(
    "http://api.icndb.com/jokes/random",
    params=params) # make the request
joke_json = result.json() # get the JSON data of the response
```

Requests Library

The `PrettyPrinter` class is used to print out the JSON data in a nicely formatted way. Let's see the difference!

```
import requests
from pprint import PrettyPrinter

pp = PrettyPrinter(indent=4) # make a PrettyPrinter object

params = { "limitTo": "nerdy" } # set the request's query parameters
result = requests.get(
    "http://api.icndb.com/jokes/random",
    params=params) # make the request
joke_json = result.json() # get the JSON data of the response
pp.pprint(joke_json) # print it out, nicely formatted :)
```

PrettyPrinter is a tool that prints JSON (the data we get from an API) in a nicely formatted, more readable way.

Requests Library

Without `PrettyPrinter`:

```
{'type': 'success', 'value': {'id': 471, 'joke': "Chuck Norris's keyboard doesn't
have a Ctrl key because nothing controls Chuck Norris.", 'categories': ['nerdy']}}
```

With `PrettyPrinter`:

```
{   'type': 'success',
    'value': {   'categories': ['nerdy'],
                 'id': 471,
                 'joke': 'Chuck Norris's keyboard doesn't have a Ctrl key '
                         'because nothing controls Chuck Norris.'}}
```

What's the difference?

Requests Library

Here's a **simplified look** at how to make an API request. Replace "KEY" and "VALUE" with any request parameters (optional), and replace the URL with the API's URL.

```
import requests

...

params = { "KEY": "VALUE" }
result = requests.get("http://my-api.co", params=params)
result_json = result.json()
# do something with result_json
```

this is what's actually making the API request

Activity (25 minutes)

Complete the TODOs in the Joke API Repl.lt program.

Flask Route

We can turn our program into a Flask route, to show the user a random joke:

```
import requests
from flask import Flask

app = Flask(__name__)

@app.route('/joke')
def make_joke():
    params = { "limitTo": "nerdy" } # set the request's query parameters
    result = requests.get( # make the API request
        "http://api.icndb.com/jokes/random", params=params)
    joke_json = result.json() # get the response's JSON data
    joke_text = joke_json["value"]["joke"] # get the joke text
    return joke_text # send the joke text to the browser
```

Activity (20 minutes)

Complete the TODOs in the Flask APIs Repl.It program.

JSON

What is JSON?

JSON stands for **JavaScript Object Notation**. Any time we make an API call, the **response data** is returned in JSON format.

Body | Cookies (1) | Headers (15) | Test Results | Status: 200 OK

Pretty | Raw | Preview | Visualize | JSON

```
1  {
2      "type": "success",
3      "value": {
4          "id": 493,
5          "joke": "Chuck Norris can binary search unsorted data.",
6          "categories": [
7              "nerdy"
8          ]
9      }
10 }
```

What is JSON?

In Python, we refer to **JSON data** as a **dictionary**. (In reality, they're slightly different, but their syntax/usage is exactly the same.)

A **dictionary** is just a collection of **key-value pairs**. The keys must be strings, but the values can be any data type. Notice how a value can contain a list, or even another dictionary!

```
my_info = {
    'name': 'Fluffy',
    'likes': ['tennis balls', 'walks'],
    'address': {
        'street': '555 Post St',
        'city': 'San Francisco'
    }
}
```

What is JSON?

What if we want to get Fluffy's street address? Type in the missing code!

```
my_info = {
    'name': 'Fluffy',
    'likes': ['tennis balls', 'walks'],
    'address': {
        'street': '555 Post St',
        'city': 'San Francisco'
    }
}

street_address =
```

JSON Practice Activity (15 minutes)

With a partner, complete the TODOs in the Dictionaries Practice Repl.it.

The Weather API

Weather API

For Homework 3, we'll be using the OpenWeatherMap API to build a project.

This API gives us data on the current temperature, humidity, sunrise/sunset times, description, etc. for a given city.

Follow the steps on this page to get your own API key!

Weather API - Activity (15 minutes)

With a partner, complete the TODOs in the Weather API Practice Repl.it. **Make sure you use your own API key!**

Lab Time: Work on Homework 3

Homework

Due today: Quiz 1 (if you are taking the quiz, leave the zoom room so I don't disturb you!)

Due on Thursday: Homework 3