



Curriculum

**SE Foundations** ^

Average: 137.49% v

You have a captain's log due before 2024-04-21 (in 1 day)! Log it now!  
(/captain\_logs/5596018/edit)

# 0x16. API advanced

Python

Scripting

Back-end

API

⚙ Weight: 1

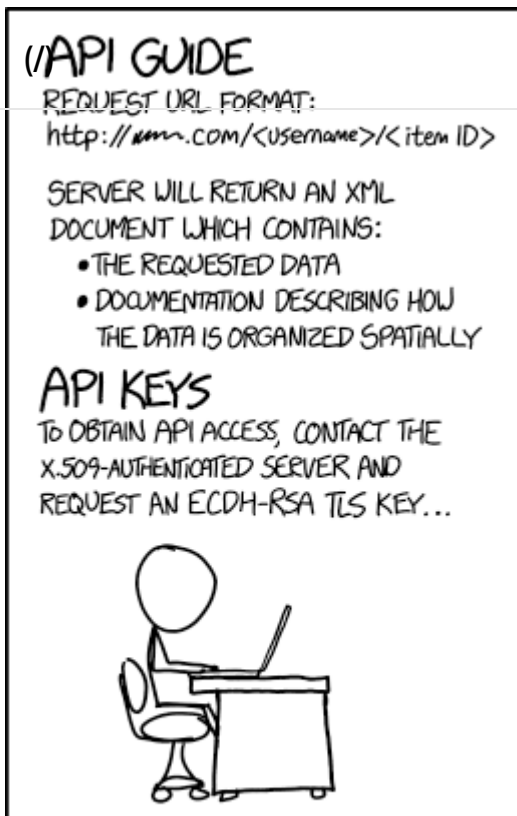
📅 Project over - took place from Mar 5, 2024 6:00 AM to Mar 6, 2024 6:00 AM

☒ An auto review will be launched at the deadline

## In a nutshell...

- **Auto QA review:** 4.0/17 mandatory & 7.0/7 optional
- **Altogether: 47.06%**
  - Mandatory: 23.53%
  - Optional: 100.0%
  - Calculation:  $23.53\% + (23.53\% * 100.0\%) == 47.06\%$





IF YOU DO THINGS RIGHT, IT CAN TAKE PEOPLE A WHILE TO REALIZE THAT YOUR "API DOCUMENTATION" IS JUST INSTRUCTIONS FOR HOW TO LOOK AT YOUR WEBSITE.

## Background Context

Questions involving APIs are common for interviews. Sometimes they're as simple as 'write a Python script that queries a given endpoint', sometimes they require you to use recursive functions and format/sort the results.

A great API to use for some practice is the Reddit API. There's a lot of endpoints available, many that don't require any form of authentication, and there's tons of information to be parsed out and presented. Getting comfortable with API calls now can save you some face during technical interviews and even outside of the job market, you might find personal use cases to make your life a little bit easier.

## Resources

Read or watch:

- [Reddit API Documentation \(/rltoken/b-4nD6hwEeNYTwYI5yWNwA\)](#)
- [Query String \(/rltoken/lufn\\_zrgmAQ00AO\\_PEI9bA\)](#)

## Learning Objectives

At the end of this project, you are expected to be able to explain to anyone ([/rltoken/uDfkZ\\_HQ\\_YnelvPnhnBOnw](#)), **without the help of Google**:



## General

- How to read API documentation to find the endpoints you're looking for
- How to use an API with pagination
- How to parse JSON results from an API
- How to make a recursive API call
- How to sort a dictionary by value

## Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

## Requirements

### General

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be interpreted/compiled on Ubuntu 20.04 LTS using `python3` (version 3.4.3)
- All your files should end with a new line
- The first line of all your files should be exactly `#!/usr/bin/python3`
- Libraries imported in your Python files must be organized in alphabetical order
- A `README.md` file, at the root of the folder of the project, is mandatory
- Your code should use the `PEP 8` style
- All your files must be executable
- The length of your files will be tested using `wc`
- All your modules should have a documentation ( `python3 -c 'print(__import__("my_module").__doc__)'` )
- You must use the `Requests` module for sending HTTP requests to the Reddit API

## Tasks

### 0. How many subs?

mandatory

Score: 20.0% (Checks completed: 20.0%)



Write a function that queries the Reddit API (`/r/token/b-4nD6hwEeNYTwYI5yWNwA`) and returns the number of subscribers (not active users, total subscribers) for a given subreddit. If an invalid subreddit is given, the function should return 0.

Hint: No authentication is necessary for most features of the Reddit API. If you're getting errors related to Too Many Requests, ensure you're setting a custom User-Agent.

#### Requirements:

- Prototype: `def number_of_subscribers(subreddit)`
- If not a valid subreddit, return 0.
- NOTE: Invalid subreddits may return a redirect to search results. Ensure that you are not following redirects.

```
wintermancer@lapbox ~/reddit_api/project $ cat 0-main.py
#!/usr/bin/python3
"""
0-main
"""
import sys

if __name__ == '__main__':
    number_of_subscribers = __import__('0-subs').number_of_subscribers
    if len(sys.argv) < 2:
        print("Please pass an argument for the subreddit to search.")
    else:
        print("{:d}".format(number_of_subscribers(sys.argv[1])))
wintermancer@lapbox ~/reddit_api/project $ python3 0-main.py programming
756024
wintermancer@lapbox ~/reddit_api/project $ python3 0-main.py this_is_a_fake_subreddit
0
```

#### Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x16-api_advanced`
- File: `0-subs.py`

☐ Done?

Check your code

Ask for a new correction

QA Review

## 1. Top Ten

mandatory

Score: 33.33% (Checks completed: 33.33%)

Write a function that queries the Reddit API (`/r/token/b-4nD6hwEeNYTwYI5yWNwA`) and prints the titles of the first 10 hot posts listed for a given subreddit.

#### Requirements:

- Prototype: `def top_ten(subreddit)`
- If not a valid subreddit, print None.
- NOTE: Invalid subreddits may return a redirect to search results. Ensure that you are not following redirects.



```
wintermancer@lapbox ~/reddit_api/project $ cat 1-main.py
```

```
#!/usr/bin/python3
```

```
"""
```

```
1-main
```

```
"""
```

```
import sys
```

```
if __name__ == '__main__':
```

```
    top_ten = __import__('1-top_ten').top_ten
```

```
    if len(sys.argv) < 2:
```

```
        print("Please pass an argument for the subreddit to search.")
```

```
    else:
```

```
        top_ten(sys.argv[1])
```

```
wintermancer@lapbox ~/reddit_api/project $ python3 1-main.py programming
```

```
Firestore founder's response to last week's "Firestore Costs increased by 7000%!"
```

```
How a 64k intro is made
```

```
HTTPS on Stack Overflow: The End of a Long Road
```

```
Spend effort on your Git commits
```

```
It's a few years old, but I just discovered this incredibly impressive video of researchers  
reconstructing sounds from video information alone
```

```
From the D Blog: Introspection, Introspection Everywhere
```

```
Do MVC like it's 1979
```

```
GitHub is moving to GraphQL for v4 of their API (v3 was a REST API)
```

```
Google Bug Bounty - The 5k Error Page
```

```
PyCon 2017 Talk Videos
```

```
wintermancer@lapbox ~/reddit_api/project $ python3 1-main.py this_is_a_fake_subreddit
```

```
None
```

```
wintermancer@lapbox ~/reddit_api/project $
```

## Repo:

- GitHub repository: alx-system\_engineering-devops
- Directory: 0x16-api\_advanced
- File: 1-top\_ten.py

☐ Done?

Check your code

Ask for a new correction

QA Review

## 2. Recurse it!

mandatory

Score: 16.67% (Checks completed: 16.67%)

Write a *recursive function* that queries the Reddit API (/r/token/b-4nD6hwEeNYTwYI5yWNwA) and returns a list containing the titles of all hot articles for a given subreddit. If no results are found for the given subreddit, the function should return None.

Hint: The Reddit API uses pagination for separating pages of responses.

Requirements:

- Prototype: `def recurse(subreddit, hot_list=[])`
- (/). Note: You may change the prototype, but it must be able to be called with just a subreddit supplied. AKA you can add a counter, but it must work without supplying a starting value in the main.
- If not a valid subreddit, return None.
- NOTE: Invalid subreddits may return a redirect to search results. Ensure that you are not following redirects.

Your code will NOT pass if you are using a loop and not recursively calling the function! This /can/ be done with a loop but the point is to use a recursive function. :)

```
wintermancer@lapbox ~/reddit_api/project $ cat 2-main.py
#!/usr/bin/python3
"""
2-main
"""
import sys

if __name__ == '__main__':
    recurse = __import__('2-recurse').recurse
    if len(sys.argv) < 2:
        print("Please pass an argument for the subreddit to search.")
    else:
        result = recurse(sys.argv[1])
        if result is not None:
            print(len(result))
        else:
            print("None")
wintermancer@lapbox ~/reddit_api/project $ python3 2-main.py programming
932
wintermancer@lapbox ~/reddit_api/project $ python3 2-main.py this_is_a_fake_subreddit
None
```

### Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x16-api_advanced`
- File: `2-recurse.py`

☐ Done?

☐ Check your code

☒ Ask for a new correction

☐ QA Review

### 3. Count it!

#advanced

Score: 100.0% (Checks completed: 100.0%)



Write a *recursive function* that queries the Reddit API (`/r/token/b-4nD6hwEeNYTwYI5yWNwA`), parses the title of all hot articles, and prints a sorted count of given keywords (case-insensitive, delimited by spaces. Javascript should count as javascript , but java should not).

## Requirements:

(/)


- Prototype: `def count_words(subreddit, word_list)`
- Note: You may change the prototype, but it must be able to be called with just a subreddit supplied and a list of keywords. AKA you can add a counter or anything else, but the function must work without supplying a starting value in the main.
- If `word_list` contains the same word (case-insensitive), the final count should be the sum of each duplicate (example below with `java`)
- Results should be printed in descending order, by the count, and if the count is the same for separate keywords, they should then be sorted alphabetically (ascending, from A to Z). Words with no matches should be skipped and not printed. Words must be printed in lowercase.
- Results are based on the number of times a keyword appears, not titles it appears in. `java java java` counts as 3 separate occurrences of `java`.
- To make life easier, `java.` or `java!` or `java_` should not count as `java`
- If no posts match or the subreddit is invalid, print nothing.
- NOTE: Invalid subreddits may return a redirect to search results. Ensure that you are NOT following redirects.

Your code will NOT pass if you are using a loop and not recursively calling the function! This /can/ be done with a loop but the point is to use a recursive function. :)

**Disclaimer:** number presented in this example *cannot be accurate now* - Reddit is hot articles are always changing.

```
bob@dylan $ cat 100-main.py
#!/usr/bin/python3
"""
100-main
"""
import sys

if __name__ == '__main__':
    count_words = __import__('100-count').count_words
    if len(sys.argv) < 3:
        print("Usage: {} <subreddit> <list of keywords>".format(sys.argv[0]))
        print("Ex: {} programming 'python java javascript'".format(sys.argv[0]))
    else:
        result = count_words(sys.argv[1], [x for x in sys.argv[2].split()])
bob@dylan $
bob@dylan $ python3 100-main.py programming 'react python java javascript scala no_results_for_this_one'
java: 27
javascript: 20
python: 17
react: 17
scala: 4
bob@dylan $ python3 100-main.py programming 'Java java'
java: 54
bob@dylan $ python3 100-main.py not_a_valid_subreddit 'python java javascript scala no_results_for_this_one'
bob@dylan $ python3 100-main.py not_a_valid_subreddit 'python java'
bob@dylan $
```



(/)  
**Repo:**

- GitHub repository: alx-system\_engineering-devops
- Directory: 0x16-api\_advanced
- File: 100-count.py

☒ Done!

Check your code

[>\\_ Get a sandbox](#)

QA Review

Copyright © 2024 ALX, All rights reserved.

