

Institute of  
Data



2020



# Data Science and AI

Module 2  
Part 2:

---

## APIs

---



## Agenda: Module 2 Part 2

- What is an API?
- APIs for data services
- APIs for analytic services
- APIs for visualisation services
- APIs for cognitive services
- Creating an API



## What is an API?

- Definition, examples
- Interfaces
- Authentication protocols
- Documentation



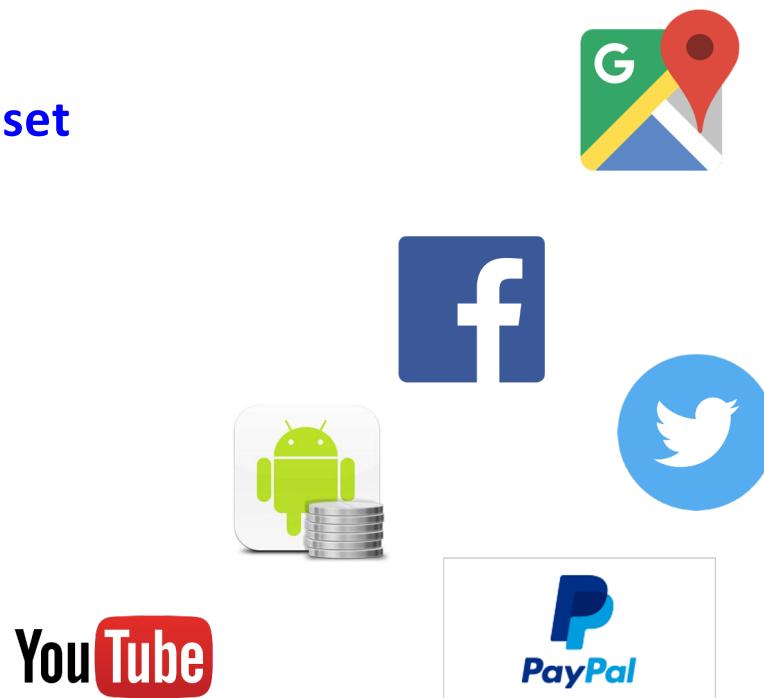
# What is an API?

- What does “API” stand for?
  - Application Programming Interface
- Examples?
  - automation in Microsoft Office
    - e.g. generating a Word document or an Outlook reminder from another application
  - high-level database drivers
    - e.g. PyMongo
  - programming libraries for mobile & wearable devices
  - programmable web services
  - other?



# Use Cases for APIs

- integrate remote data access
  - repetitive analyses of an **evolving dataset**
  - **up-to-the-moment** forecasting
- **integrate** familiar functionality
  - location sharing using Google Maps
  - simplified app login via Facebook
  - in-app purchases
  - in-app YouTube viewing





# Some Popular Web Service APIs

Name	Nature	URL
Twitter	Networking, marketing, trending	<a href="https://developer.twitter.com/en.html">https://developer.twitter.com/en.html</a>
Facebook	Networking, marketing	<a href="https://developers.facebook.com/tools/">https://developers.facebook.com/tools/</a>
Amazon S3	Cloud storage, Big Data analytics	<a href="https://aws.amazon.com/s3/">https://aws.amazon.com/s3/</a>
LinkedIn	Networking	<a href="https://developer.linkedin.com/">https://developer.linkedin.com/</a>
eBay	E-commerce	<a href="https://developer.ebay.com/">https://developer.ebay.com/</a>
Google API Console	Data access & analytics, e-commerce, etc.	<a href="https://developers.google.com/apis-explorer/#p/">https://developers.google.com/apis-explorer/#p/</a>
New York Times	News	<a href="http://developer.nytimes.com/">http://developer.nytimes.com/</a>



# Interfaces for Web Service APIs

- SOAP
  - *Simple Object Access Protocol*
  - early, widespread web service protocol
  - exposes components of application logic as services
  - XML
- REST
  - *Representational State Transfer*
  - now > 70% of public APIs
  - accesses data
  - variety of data formats, coupled with JSON
  - generally faster and uses less bandwidth
  - easier to integrate with existing websites

Overview of RESTful API

Description Languages:

[https://en.wikipedia.org/wiki/Overview\\_of\\_RESTful\\_API\\_Description\\_Languages](https://en.wikipedia.org/wiki/Overview_of_RESTful_API_Description_Languages)

roll your own:

<https://www.restapitutorial.com/>

<https://aws.amazon.com/api-gateway>



# HTTP

- HyperText Transfer Protocol
  - underlies RESTful APIs
  - 4 major methods
    - GET fetches data from web server
    - PUT edits data on web server
    - POST adds new data
    - DELETE removes data
- HTTP Status Codes
    - 1xx informational
    - 2xx success
    - 3xx redirection
    - 4xx client error
    - 5xx server error

<https://www.restapitutorial.com/httpstatuscodes.html>



# Elements of an API call

- ***endpoint***
  - URL of a server page that provides data or functionality via ***requests*** and ***responses***
- ***protocol***
  - the communication standard for passing requests to an endpoint
- ***authentication***
  - secure ***identification*** of user making request
  - if a developer creates an app for other users, the app needs to obtain ***authorisation*** from the owner of the API for both the developer's access *and* the user's access



# Authentication Protocols

- HTTP Basic Access Authentication
  - username + password
  - transmitted in header of HTTP request
  - weakly encoded, no encryption
- OAuth 1.0
  - uses encrypted tokens
- OAuth 2.0
  - simpler, more robust than OAuth 1.0



# OAuth 2.0

- token-based
  - e.g. ***client\_id*** & ***client\_secret***
  - allows a 3<sup>rd</sup>-party app to access a user's/developer's account **without knowing the account password**
  - allows an end-user to access an API via ***your*** app, using ***their*** token
- redirect URL
  - **registered** when app created
  - OAuth 2.0 service **returns user to this URL** after authorising (and issuing a user token)
  - protects access token from **interception**

<https://www.oauth.com/oauth2-servers/background/>



# Developer Access

- some API's have **a developer mode** that may allow access without requesting a user token
- options for connect/request include:
  - use developer's *user\_id* and *password*
  - use *app\_id*, developer's *client\_id*, developer's *secret*
- access granted **may** include
  - read developer's posts, comments, profile, etc.
  - post to developer's account
  - read other users' posts, comments, profiles, etc.



# Python Libraries: Utilities

## requests

- HTTP library (“elegant and simple”)
- <http://www.python-requests.org/en/latest/>
- returns JSON-formatted byte strings

## json

- JSON ↔ lists, dictionaries
- <https://docs.python.org/2/library/json.html>

## untangle, xmldict

- parses XML to Pythonic data structures

## BeautifulSoup (bs4)

- parses HTML, XML to Pythonic data structures



# Python Libraries: API Wrappers

- simplify usage of APIs by introducing a Python API into the loop
- use data types & structures familiar to Python developers

*[pyfacebook](#)*

*[linkedin](#)*

*[praw](#)* (Reddit)

*[bucketstore](#)* (Amazon S3)

*[python-forecastio](#)* (weather)

*[foursquare](#)* (location-based networking)

*[GooPyCharts](#)* (Google Charts)

*[indeed](#)* (indeed.com)

*[kiteconnect](#)* (stock trading)

*[pymaps](#)* (Google Maps)

*[pymed](#)* (PubMed)

*[pyspotify](#)* (Spotify)

*[newsapi](#)*

*[rottentomatoes](#)* (crowd-based movie reviews)

*[sportradar](#)* (sport APIs)

*[tesserocr](#)* (OCR)

*[bowshock](#)* (NASA)

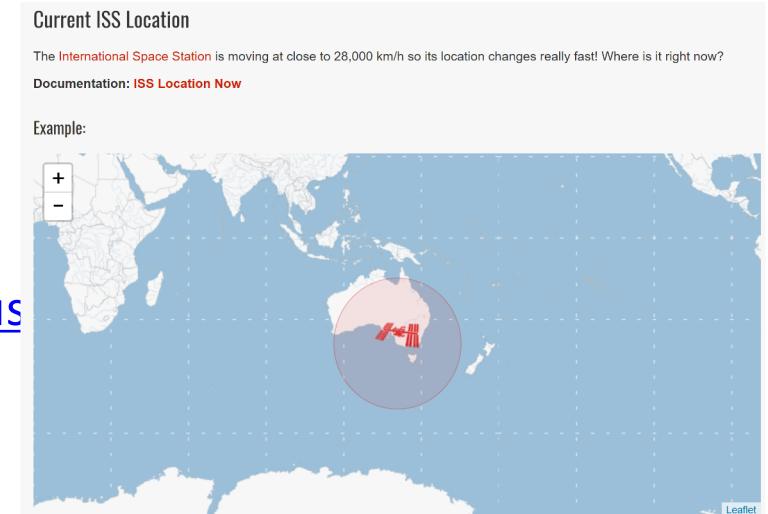
*[geopy](#)* (geocoding)

<https://github.com/realpython/list-of-python-api-wrappers>



# Lab 2.2.1: Querying the ISS

- Purpose:
  - To become familiar with basic API requests and responses
- Resources:
  - API for the International Space Station:  
**OpenNotify**  
<http://open-notify.org/Open-Notify-API/>
  - HTTP response codes  
<https://www.restapitutorial.com/httpstatus>
- Materials:
  - ‘Lab 2.2.1.ipynb’





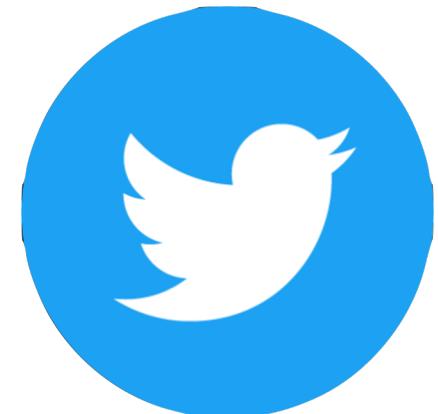
## Extracting Data from APIs

- Twitter API
- Reddit API
- Facebook API
- Google Public Data and BigQuery API



## Twitter API

- Twitter API structure
- API Usage restrictions
- Developer / App approval



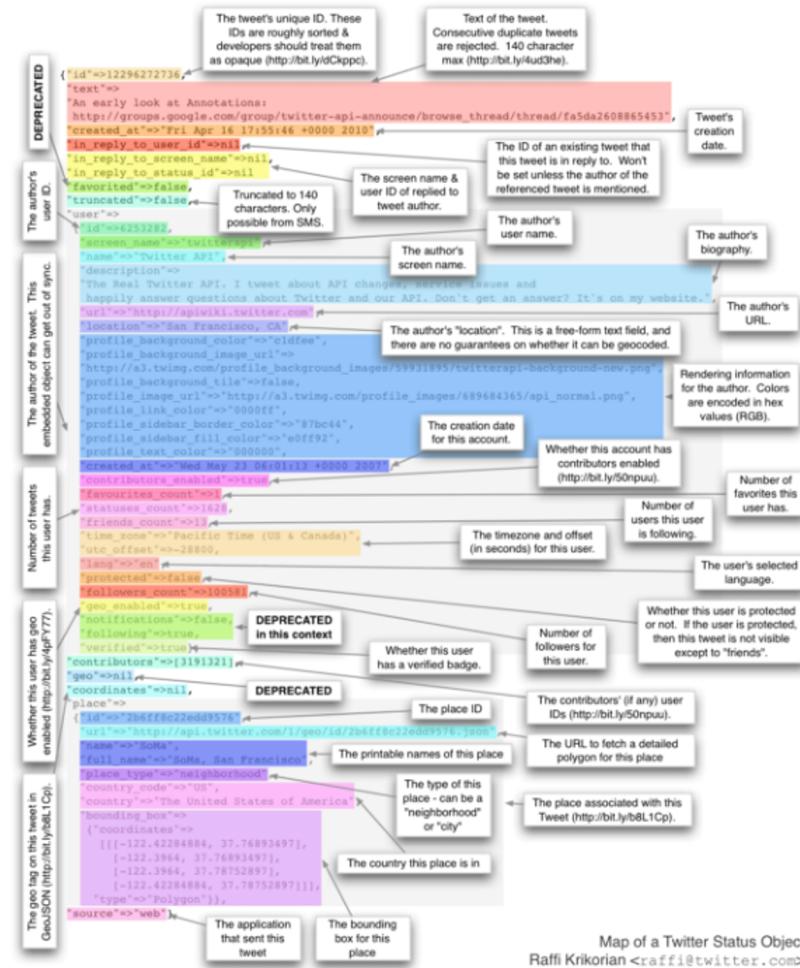


# Twitter API

- Tweets are **status updates**
- main API object = **status**
  - root-level attributes:
    - *id, created\_at, text, ...*
  - child objects:
    - *user, entities, extended\_entities, ...*
    - *place* (if Tweet was geo-tagged)

<https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object.html>

<http://socialmedia-class.org/twittertutorial.html>



Map of a Twitter Status Object  
Raffi Krikorian <[raffi@twitter.com](mailto:raffi@twitter.com)>  
18 April 2010



# Twitter API

1. If you haven't got one, open a Twitter user account
2. Create a Twitter app (<https://developer.twitter.com/en/apps>)
3. Register the app for API access
4. Store your credentials
  - for accessing your account:
    - user name
    - password
  - for authenticating your app:
    - user agent (information describing your app)
    - client ID (a unique identifier for your app)
    - client secret (secure token for authorising your app to access the API)



# Twitter App Restrictions

- Terms & Conditions:

C. **Geographic Data.** Your license to use Twitter Content in this Agreement does not allow you to (and you will not allow others to) aggregate, cache, or store location data and other geographic information contained in the Twitter Content, except in conjunction with the Twitter Content to which it is attached. Your license only allows you to use such location data and geographic information to identify the location tagged by the Twitter Content. Any use of location data or geographic information on a standalone basis or beyond the license granted herein is a breach of this Agreement.



# Twitter App Approval

## New developer requirements to protect our platform

[https://blog.twitter.com/developer/en\\_us/topics/tools/2018/new-developer-requirements-to-protect-our-platform.html](https://blog.twitter.com/developer/en_us/topics/tools/2018/new-developer-requirements-to-protect-our-platform.html)



### Application under review.

Thanks! We've received your application and are reviewing it. We'll be in touch soon.

We review applications to ensure compliance with our Terms of Service and Developer policies. [Learn more.](#)

To help us understand how you use your existing apps, please [edit each of your apps](#) and add a description of your app's use case where it says "Tell us how this app will be used".

You'll receive an email when the review is complete. While you wait, check out our [documentation](#), explore our [tutorials](#), or check out our [community forums](#).



## Lab 2.2.2: Mining Social Media with Twitter

- Purpose:
  - To develop skills in using a more complex API
- Resources:
  - Python library for Twitter API: *Tweepy*
    - <http://docs.tweepy.org>
- Materials:
  - ‘Lab 2.2.2.ipynb’





## Reddit API

- Introduction to Reddit
- API structure
- Developer access
- Reddit API: Using Python





# Reddit

- why Reddit?
  - good example of a social media product
  - rich content
  - large user base
  - highly structured API
  - immediately accessible

<https://www.reddit.com/wiki/faq>

The screenshot shows the Reddit homepage with the 'Popular' sort selected. The interface includes a search bar, a navigation bar with 'VIEW' and 'SORT HOT AUSTRALIA', and a sidebar with 'Otherwized' and '1 karma'. The main content area displays several posts with their upvote counts, titles, and descriptions. A sidebar on the right features a 'r/popular' section, a VMware advertisement, and a 'TRENDING COMMUNITIES' section with links to r/FromPuppyToDog, r/Hookit, r/thisismylifenow, and r/couldntagreemore.

Post Title	Upvotes	Comments	Link
He made a homemade toy for his kids....and they loved it.	32.5k	661	<a href="#">gfycat.com/NewFri...</a>
Ubisoft logo , my creation with onion.	11.8k	231	<a href="#">i.reddit.it/y0bkeq...</a>
Ninja Warrior's First Jump (alternate version in comments!)	8.9k	80	<a href="#">gfycat.com/Vacant...</a>
Hey Reddit: Want to write better? Eliminate grammatical mistakes, wipe out wordiness, and let your ideas shine. See for yourself why over 10 million users are hooked on Grammarly's free writing app.	544	0	<a href="#">https://www.grammarly.com/</a>
Smooth boi.	12.8k	101	<a href="#">i.imgur.com/JGPhon...</a>
We've all done it	7.5k	73	<a href="#">i.reddit.it/9lxq02...</a>
This shark, older than the United States.	14.3k	661	<a href="#">i.reddit.it/1wu1z2...</a>
A study has found that students appreciate when instructors tell jokes in science class, but that female and male students differ in what topics they find funny or offensive. Funny humor tends to increase student attention to course content, instructor relatability, and student sense of belonging.	13.7k	0	<a href="#">Social Science asunow.asu.edu/201808...</a>



# Reddit API

- Account endpoints:
  - *me, me/friends, me/prefs, ...*
- Links & comments endpoint:
  - *comment, vote, report, ...*
- Listing endpoints:
  - categories
    - *hot, new, random, ...*
  - navigation (pagination) and filtering
    - *before, after, count, show*
- and many more ...

The screenshot shows a dark-themed interface for the Reddit API methods. At the top is a white Reddit logo. Below it is a navigation bar with tabs: 'API methods' (selected), 'by section' (highlighted in blue), and 'by oauth scope'. There are two buttons: 'by section' and 'by oauth scope'. The main content area is divided into sections: 'account', 'captcha', 'emoji', and 'flair'. Each section lists several API endpoints along with their corresponding OAuth scopes. Most endpoints have the 'oauth' scope applied.

Section	Endpoint	OAuth Scope	
account	/api/v1/me	oauth	
	/api/v1/me/blocked	oauth	
	/api/v1/me/friends	oauth	
	/api/v1/me/karma	oauth	
	/api/v1/me/prefs	oauth	
	/api/v1/me/trophies	oauth	
	/prefs/blocked	oauth	
	/prefs/friends	oauth	
	/prefs/messaging	oauth	
	/prefs/trusted	oauth	
	/prefs/where	oauth	
	captcha	/api/needs_captcha	oauth
		emoji	/api/v1/subreddit/emoji.json
/api/v1/subreddit/emoji/emoji_name	oauth		
/api/v1/subreddit/emoji_asset	oauth		
flair	/api/v1/subreddit/flair	ad_3.0, oauth	
	/api/clearflairtemplates	oauth	
	/api/deleteflair	oauth	
	/api/deleteflairtemplate	oauth	
	/api/flair	oauth	

<https://www.reddit.com/dev/api>



# Reddit API: Developer Access

1. Open a Reddit user account
2. Create a Reddit app
3. Register the app for API access
4. Store your credentials
  - for accessing your account:
    - user name
    - password
  - for authenticating your app:
    - user agent (information describing your app)
    - client ID (a unique identifier for your app)
    - client secret (secure token for authorising your app to access the API)



# Reddit API: Using Python

- install PRAW package
- import praw
- create a connection object (to Reddit API)
- invoke API methods on the connection object
  - send requests that GET or PUT data to/from Reddit objects
- do something with data!

<https://www.reddit.com/r/popular/>

<https://www.reddit.com/wiki/faq>

[https://praw.readthedocs.io/en/stable/getting\\_started/quick\\_start.html](https://praw.readthedocs.io/en/stable/getting_started/quick_start.html)



## Lab 2.2.3: Mining Social Media with Reddit (Optional homework)

- Purpose:
  - To develop skills in using a media-rich API
- Resources:
  - Python library for Reddit API: *PRAW*  
[https://praw.readthedocs.io/en/stable/getting\\_started/quick\\_start.html](https://praw.readthedocs.io/en/stable/getting_started/quick_start.html)
- Materials:
  - ‘Lab 2.2.3.ipynb’





# Discussion

- Questions



# HOMEWORK

- Continue working with Lab 2.2.2 / 2.2.3:
  - choose one or more Twitter or Reddit API members and write code to demonstrate how they could be used to reveal something interesting



## Facebook API

- developer tools
- app review process





# Facebook API

## Developer

### Tools



#### [Graph API Explorer](#)

Test, create, and authenticate API calls and debug responses.



#### [Sharing Debugger](#)

Preview how your content will look when it's shared to Facebook.



#### [Access Token Debugger](#)

See detailed info for an access token.

use Graph API Explorer for developing requests to use in API calls from your (Python) application code



# Facebook Graph API Explorer

- discover object model
  - graph structure
  - field names
- test code before moving to Python
- experiment with permissions

Graph API Explorer      Application: [?] Allen's first app

Access Token: EAAa5jhGKG1QBAH4aY4ey9ILPNSxylptK3v5DrRB7l8uiGqkTSYBcwHEIHTO6f7vvjZBfVxybagalrPTcOpqrVJloxA8l1G3ZBINl6Ab6hutNE! [Get Token](#)

GET → /v3.1 /me?fields=posts{comments}

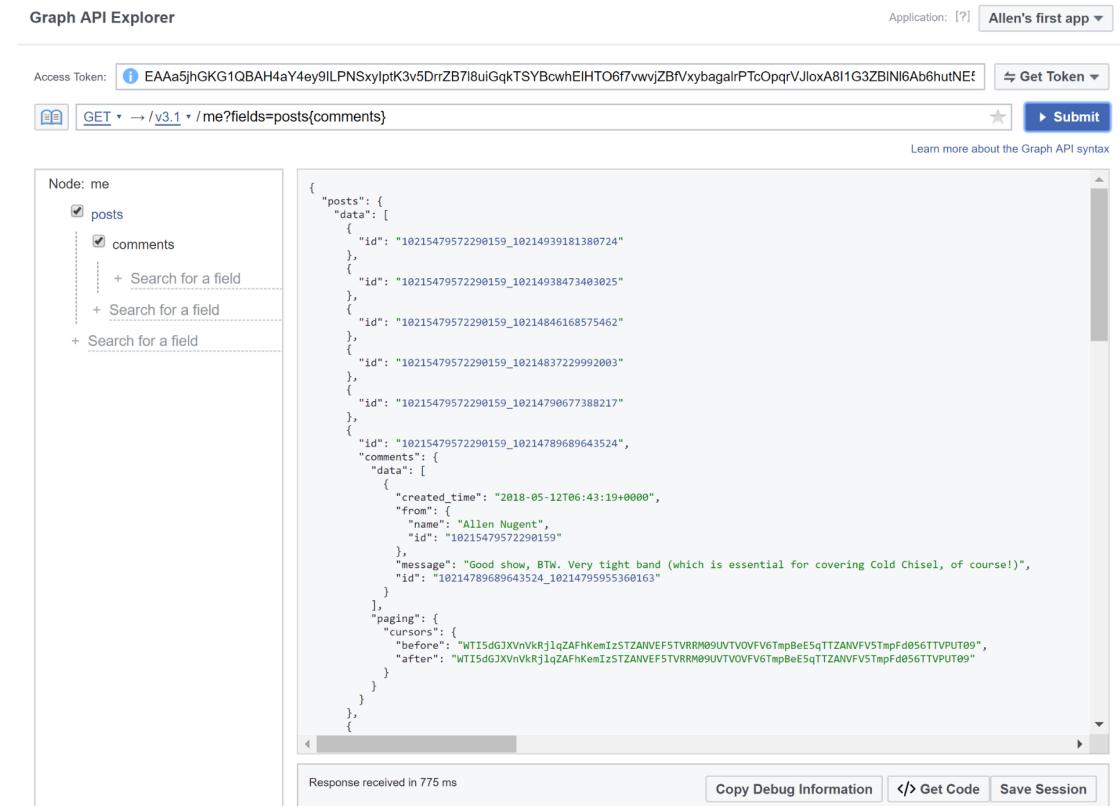
Node: me

posts  
 comments  
+ Search for a field  
+ Search for a field  
+ Search for a field

{ "posts": { "data": [ { "id": "10215479572290159\_10214939181388724" }, { "id": "10215479572290159\_10214938473403025" }, { "id": "10215479572290159\_10214846168575462" }, { "id": "10215479572290159\_10214837229992003" }, { "id": "10215479572290159\_10214790677388217" }, { "id": "10215479572290159\_10214789689643524" }, "comments": { "data": [ { "created\_time": "2018-05-12T06:43:19+0000", "from": { "name": "Allen Nugent", "id": "10215479572290159" }, "message": "Good show, BTW. Very tight band (which is essential for covering Cold Chisel, of course!)!", "id": "10214789689643524\_1021479595360163" } ], "paging": { "cursors": { "before": "WTI5dGJXvnVkrJ1qZAFhKemIzSTZANVF5TVRRM09UVTVOVF6TmpBeE5qTTZANVFV5TmpFd056TTVPUT09", "after": "WTI5dGJXvnVkrJ1qZAFhKemIzSTZANVF5TVRRM09UVTVOVF6TmpBeE5qTTZANVFV5TmpFd056TTVPUT09" } } }, { "id": "10215479572290159\_10214789689643524" } ] } }

Learn more about the Graph API syntax      [Submit](#)

Response received in 775 ms      [Copy Debug Information](#)      [Get Code](#)      [Save Session](#)





# Facebook App Review

- apps must be submitted to Facebook for review before they can go live
- as of May 2018, permission is required before any endpoint (other than the user's own pages) can be accessed from Graph API Explorer

The screenshot shows the Facebook Graph API Explorer interface. At the top, there is a header with a book icon, a 'GET' dropdown set to 'v3.1', a URL path '/GameofThrones?fields=id,name', a star icon, and a 'Submit' button. Below the header, the left panel shows a node named 'GameofThrones' with two selected fields: 'id' and 'name'. A search bar below the node name contains the placeholder 'Search for a field'. The right panel displays an error response in JSON format:

```
{  
  "error": {  
    "message": "#10 To use 'Page Public Content Access', your use of this endpoint must be reviewed and approved by Facebook.",  
    "type": "OAuthException",  
    "code": 10,  
    "fbtrace_id": "EVnHdKVqIc1"  
  }  
}
```

Below the JSON response, a link reads 'Learn more about the Graph API syntax'.



# Google Cloud Platform

- public data sets / BigQuery
- APIs based on data science products





# Google Cloud Platform

Google Cloud SDK	<ul style="list-style-type: none"><li>• <a href="https://cloud.google.com/sdk/gcloud/">https://cloud.google.com/sdk/gcloud/</a></li><li>• <a href="https://cloud.google.com/sdk/docs/initializing">https://cloud.google.com/sdk/docs/initializing</a></li></ul>
Google Cloud Platform	<ul style="list-style-type: none"><li>• <a href="https://github.com/GoogleCloudPlatform/python-docs-samples">https://github.com/GoogleCloudPlatform/python-docs-samples</a></li><li>• <a href="https://googlecloudplatform.github.io/google-cloud-python/">https://googlecloudplatform.github.io/google-cloud-python/</a></li><li>• <a href="https://googlecloudplatform.github.io/google-cloud-python/latest/">https://googlecloudplatform.github.io/google-cloud-python/latest/</a></li></ul>
Google API Client Libraries	<ul style="list-style-type: none"><li>• <a href="https://developers.google.com/api-client-library/">https://developers.google.com/api-client-library/</a></li></ul>
Google BigQuery	<ul style="list-style-type: none"><li>• <a href="https://cloud.google.com/bigquery/public-data/">https://cloud.google.com/bigquery/public-data/</a></li><li>• <a href="https://cloud.google.com/bigquery/docs/quickstarts/quickstart-web-ui">https://cloud.google.com/bigquery/docs/quickstarts/quickstart-web-ui</a></li><li>• <a href="https://cloud.google.com/bigquery/docs/reference/libraries">https://cloud.google.com/bigquery/docs/reference/libraries</a></li><li>• <a href="https://cloud.google.com/bigquery/create-simple-app-api">https://cloud.google.com/bigquery/create-simple-app-api</a></li><li>• <a href="https://github.com/GoogleCloudPlatform/google-cloud-python/tree/master/bigquery">https://github.com/GoogleCloudPlatform/google-cloud-python/tree/master/bigquery</a></li></ul>



# Google Public Data sets

- accessible via Google BigQuery
- free for 1<sup>st</sup> TB / month
- subject areas:
  - genomics
  - medicine & epidemiology
  - geo imagery (Earth science, weather, etc.)
  - transport & service utilisation
  - annotated images
  - etc.
- <https://cloud.google.com/public-datasets/>



# Google BigQuery

Quickstart to  
BigQuery Web UI:

<https://cloud.google.com/bigquery/docs/quickstarts/quickstart-web-ui>

The screenshot shows the Google BigQuery web interface. On the left, there's a sidebar with 'Query history', 'Saved queries', 'Job history', 'Transfers', and a 'Resources' section containing a search bar and a project pinned to it. The main area is the 'Query editor' where a SQL query is written:

```
1 SELECT
2   name, gender,
3   SUM(number) AS total
4 FROM
5   `bigquery-public-data.usa_names.usa_1910_2013`
6 GROUP BY
7   name, gender
8 ORDER BY
9   total DESC
10 LIMIT
11 100
```

Below the editor, it says 'Processing location: US'. There are buttons for 'Run query', 'Save query', 'Save view', and 'Options'. A note says 'This query will process 99.95 MB when run.' with a checkmark. The 'Query results' section shows the completed query with three rows of data:

Row	name	gender	total
1	James	M	4924235
2	John	M	4818746
3	Robert	M	4703680

At the bottom, there are buttons for 'Rows per page' (set to 50), '1 - 50 of 100', and navigation arrows. There are also links for 'SAVE AS' and 'EXPLORE IN DATA STUDIO'.



# BigQuery API: Authentication

## Service accounts

- for client apps that you will run
  - e.g. dev/test, batch processing pipelines
- authentication via your service credentials

## User accounts

- for apps you create for other end-users
  - e.g. data products
- authentication via end-users credentials
  - app can only access BigQuery tables that the end-user is authorised to access
  - end-user gets billed for queries

<https://cloud.google.com/bigquery/docs/authentication/>



# BigQuery API: Authentication – cont'd

GCP CONSOLE    COMMAND LINE

1. Go to the [Create service account key](#) page in the GCP Console.

[GO TO THE CREATE SERVICE ACCOUNT KEY PAGE](#)

2. From the Service account drop-down list, select **New service account**.

3. Enter a name into the **Service account name** field.

4. From the **Role** drop-down list, select **Project > Owner**.

**Note:** The **Role** field authorizes your service account to access resources. You can view and change this field later using [GCP Console](#). If you are developing a production application, specify more granular permissions than **Project > Owner**. For more information, see [granting roles to service accounts](#).

5. Click **Create**. A JSON file that contains your key downloads to your computer.

Google Cloud Platform    MyReallyBigQuery

[Create service account key](#)

Service account  
New service account

Service account name: bigquery-api-service    Role: BigQuery Admin

Service account ID: bigquery-api-service @myreallybigquery.iam.gserviceaccount.co

Key type  
Downloads a file that contains the private key. Store the file securely because this key cannot be recovered if lost.

JSON  
Recommended

P12  
For backward compatibility with code using the P12 format

[Create](#)    [Cancel](#)

<https://cloud.google.com/docs/authentication/production>



# BigQuery API: Authentication – cont'd

The screenshot shows the Google Cloud Platform API Credentials page. The top navigation bar includes the Google Cloud Platform logo, the project name "MyReallyBigQuery", and a search bar. On the left, there's a sidebar with icons for "APIs & Services", "Compute Engine", and "Cloud Storage". The main content area is titled "Credentials" and contains tabs for "Credentials", "OAuth consent screen", and "Domain verification". A prominent blue button labeled "Create credentials" with a dropdown arrow is visible. Below the tabs, a message encourages creating credentials to access APIs. A table lists a single service account key:

ID	Creation date	Service account
75516912d806a1ecdc78fa935cadb396cf9d11c6	21 Aug 2018	bigquery-api-service



# Using the Google Authentication Key

**Option 1:** Set GOOGLE\_APPLICATION\_CREDENTIALS environment variable

- Linux / MacOS

```
$ export GOOGLE_APPLICATION_CREDENTIALS="[PATH]"
```

- Windows

```
$ set GOOGLE_APPLICATION_CREDENTIALS="[PATH]"
```

**Option 2:** Pass the path to the service account key in code

```
from google.cloud import storage  
storage_client = storage.Client.from_service_account_json('[PATH]')
```

*where '[PATH]' is the full file path of the json key file*



# Google BigQuery API: Top-Level Object

`client` object:

- `connection`
  - authenticated connection to the BigQuery service
  - determines credentials
    - implicitly from the environment,
    - or directly via `from_service_account_json` and `from_service_account_p12`
- `project`
  - top-level container
  - tied to billing
  - can provide default access control across all its datasets
  - access control list (ACL)
    - grants reader / writer / owner permission to one or more entities
    - must be managed using the Google Developer Console (not API)



# BigQuery API Object Hierarchy

```
bigquery
  .projects
  .datasets
    .get, .delete, .insert, .list, .update, ...
  .tabledata
  .tables
  .jobs
    .get, .cancel, .insert, .list, .query, ...
  ...
```

<https://developers.google.com/apis-explorer/#p/bigquery/v2/>



## Lab 2.2.4: Big Data Analytics with BigQuery

- Purpose:
  - (1) To learn how to the Google BigQuery Web UI for discovering public data sets and performing basic analytics.
  - (2) To become proficient with the Google BigQuery API for wrangling Google's public datasets.
- Materials:
  - 'Lab 2.2.4.ipynb'





## Lab 2.2.4 – cont'd

- Python packages :
  - pyarrow (pip)
  - google-cloud-bigquery (conda-forge)
  - google-cloud-storage (conda-forge)
- Resources:
  - Google BigQuery Public Datasets <https://cloud.google.com/bigquery/public-data/>
  - BigQuery UI <https://cloud.google.com/bigquery/docs/quickstarts/quickstart-web-ui>
  - Python client for BigQuery API <https://github.com/GoogleCloudPlatform/google-cloud-python/tree/master/bigquery>



# Discussion

- Extracting data using APIs
  - applications?



# Analytics-Based APIs

- **Google**
  - Google Analytics
    - <https://developers.google.com/analytics/>
  - Google Cloud Vision
    - <https://cloud.google.com/vision/>
  - Google Cloud AI
    - <https://cloud.google.com/products/ai/>
- **IBM Watson**
  - Developer Cloud
    - <https://www.ibm.com/watson/developercloud/>
    - <https://github.com/watson-developer-cloud/python-sdk>
  - Mashups
    - <https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=SP&infotype=PM&htmlfid=LBS03048USEN&attachment=LBS03048USEN.PDF>



# Analytics-Based APIs – cont'd

- AWS
  - Boto3
    - low-level (“client”) and high-level (“resource”) APIs for all AWS products
    - <https://aws.amazon.com/sdk-for-python/>
  - API Explorer
    - <https://developers.google.com/apis-explorer/#search/analytics/analytics/v3/>
- Azure
  - Code samples, Cognitive Services API, etc.
    - <https://docs.microsoft.com/en-us/python/azure/?view=azure-python>
  - Python API Browser
    - <https://docs.microsoft.com/en-au/python/api/?view=azure-python>



# Machine Vision APIs

- use cases:
  - autonomous vehicles
  - industrial control & QA
  - face recognition
  - number plate recognition
  - biometric identity verification
  - print & handwriting transcription
  - image annotation
    - detecting and labelling objects or themes in an image



# Creating APIs

- Why would a data scientist/engineer want to create their own API?
  - for building an interface to your data product
  - for enforcing control over how your application's data and services can be used
  - for isolating the IP that your data product is based on
- References:
  - <https://www.fullstackpython.com/application-programming-interfaces.html>



# Discussion

## More Open Data APIs

- List of Open APIs (Wikipedia)  
[https://en.wikipedia.org/wiki/List\\_of\\_open\\_APIs](https://en.wikipedia.org/wiki/List_of_open_APIs)
- List of Open Data APIs (Programmable Web)  
<https://www.programmableweb.com/category/open-data/api>
- todmotto Public APIs  
<https://github.com/toddmotto/public-apis>



# HOMEWORK

1. Investigate a data or analytic API for one of the following:
  - AWS
  - Microsoft Azure
  - IBM Cloud
2. Create a Jupyter notebook that demonstrates some basic operations (e.g. transporting, querying, or visualising data).

## NOTES:

- The offerings of these platforms are myriad and complex. It may not be obvious which API you need to use at first, so try to start with published code examples.
- APIs (and the libraries that wrap them) change. Online examples may not work as documented.



# Questions?



# End of Presentation!