

# FB Graph API

# Outline

- Review: Defining a class
- A class that does something useful
- The FB Graph API
- Authorization with oauth
- Our workaround for not running a webserver
- The facebook module
- Installing the facebook module

# Review Exercise

- Define a class, any class
  - `__init__` method
  - At least one other method
- Write code to make at least two instances
- Invoke methods on those instances

# FB: A REST API

- But have to send an “authorization key” as a POST parameter
  - Details of POST parameters not important; we will use a module that takes care of that for us.
- <https://graph.facebook.com/?<path>>
- Some useful paths
  - /me gets info about current user
  - /me/feed gets their recent posts
  - /me/photos gets their recent photos
  - /245188182322906 gets info about our facebook group
  - 245188182322906/feed gets recent posts to our facebook group

# Facebook's Graph API Explorer

The screenshot shows the Facebook Graph API Explorer interface. The browser address bar displays the URL: `https://developers.facebook.com/tools/explorer?method=GET&path=245188182322906%2F&version=`. The navigation bar includes links for Developers, Apps, Products, Docs, Tools, and Support, along with a search bar labeled "Search in docs".

The main section is titled "Graph API Explorer". It includes a dropdown for "Application" set to "Graph API Explorer" and a dropdown for "Locale" set to "English (US)".

The "Access Token" field contains the token: `CAACEdEose0cBAHpGRidlqrXkP6BjP3iDIIL3tIDl4fBC5fbWTdNQzlrwUj2Eg1dk2JKC`. There are buttons for "Debug", "Get Access Token", and "Get App Token".

Below the token field, there are tabs for "Graph API" and "FQL Query". The "Graph API" tab is active, showing a "GET" method and a path of `/245188182322906/`. A "Submit" button is located to the right of the path field.

A link "Learn more about the Graph API syntax." is provided below the path field.

The response area shows the JSON data for the edge `245188182322906/`. The JSON structure is as follows:

```
{
  "id": "245188182322906",
  "owner": {
    "name": "Jackie Cohen",
    "id": "1246950181"
  },
  "name": "SI 106 - Winter 2014",
  "description": "This is a Facebook group for SI 106 - Programs, Information, and People, for Winter 2014. Please join if you're in the class (please don't if you aren't)!",
  "venue": {
    "street": ""
  },
  "privacy": "CLOSED",
  "icon": "https://fbstatic-a.akamaihd.net/rsrc.php/v2/y0/r/XCrOg4YmGg4.png",
  "updated_time": "2014-03-16T21:03:16+0000",
  "email": "245188182322906@groups.facebook.com"
}
```

# **OAUTH**

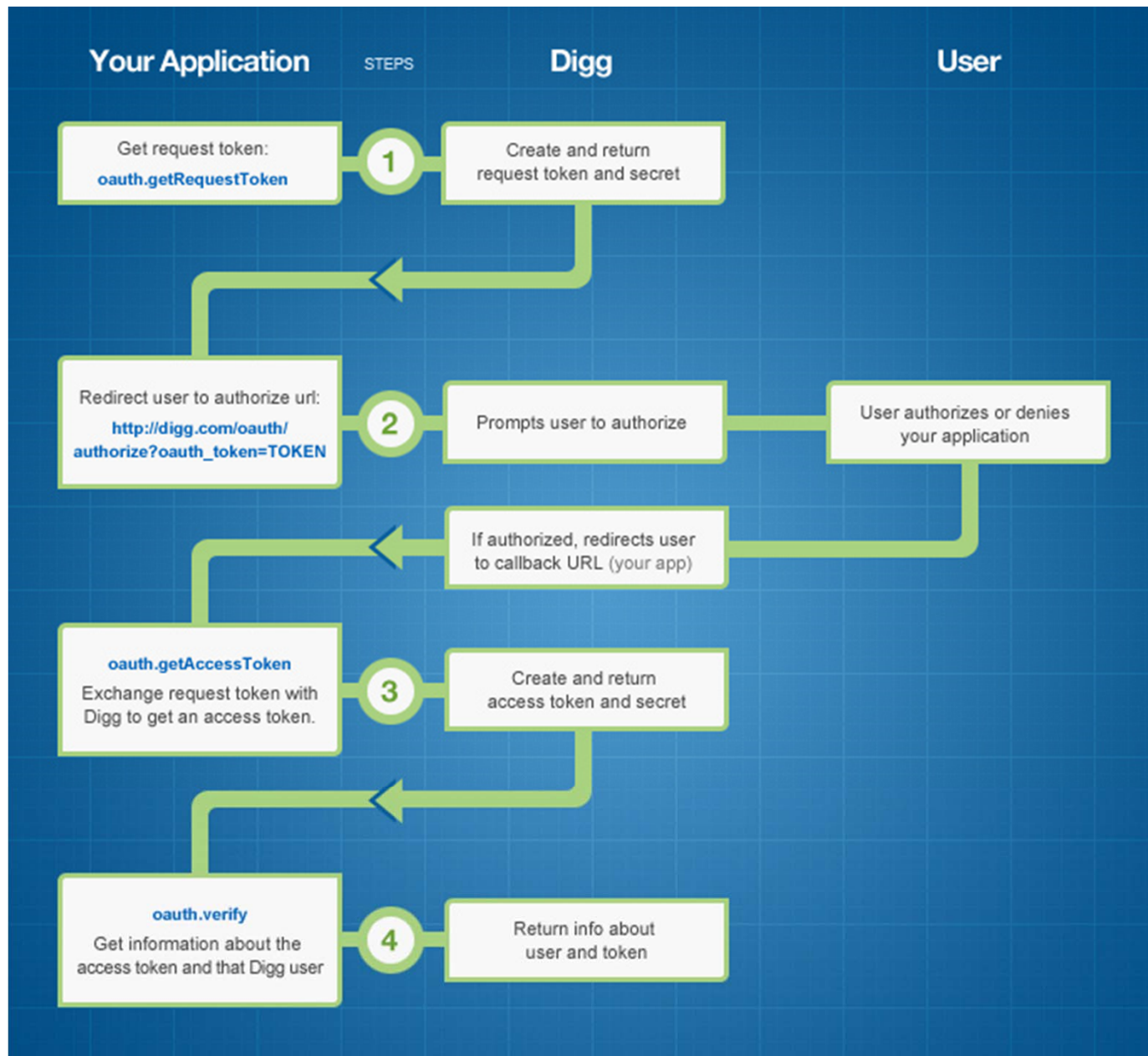
# Purpose

- User interacts with site A
- Authentication
  - Prove you own an account on site B
- (limited) Authorization
  - Give site A ability to access things on site B, acting on behalf of user
- Don't reveal your site B password to site A
  - maybe not even username

# oAuth Terminology

- User: the person who has to log in
- Consumer (site A)
  - the application that will access data from elsewhere
  - e.g., your code running on your desktop
- Service Provider (site B)
  - The place the user has to log in to
  - e.g., FaceBook, Digg, Twitter





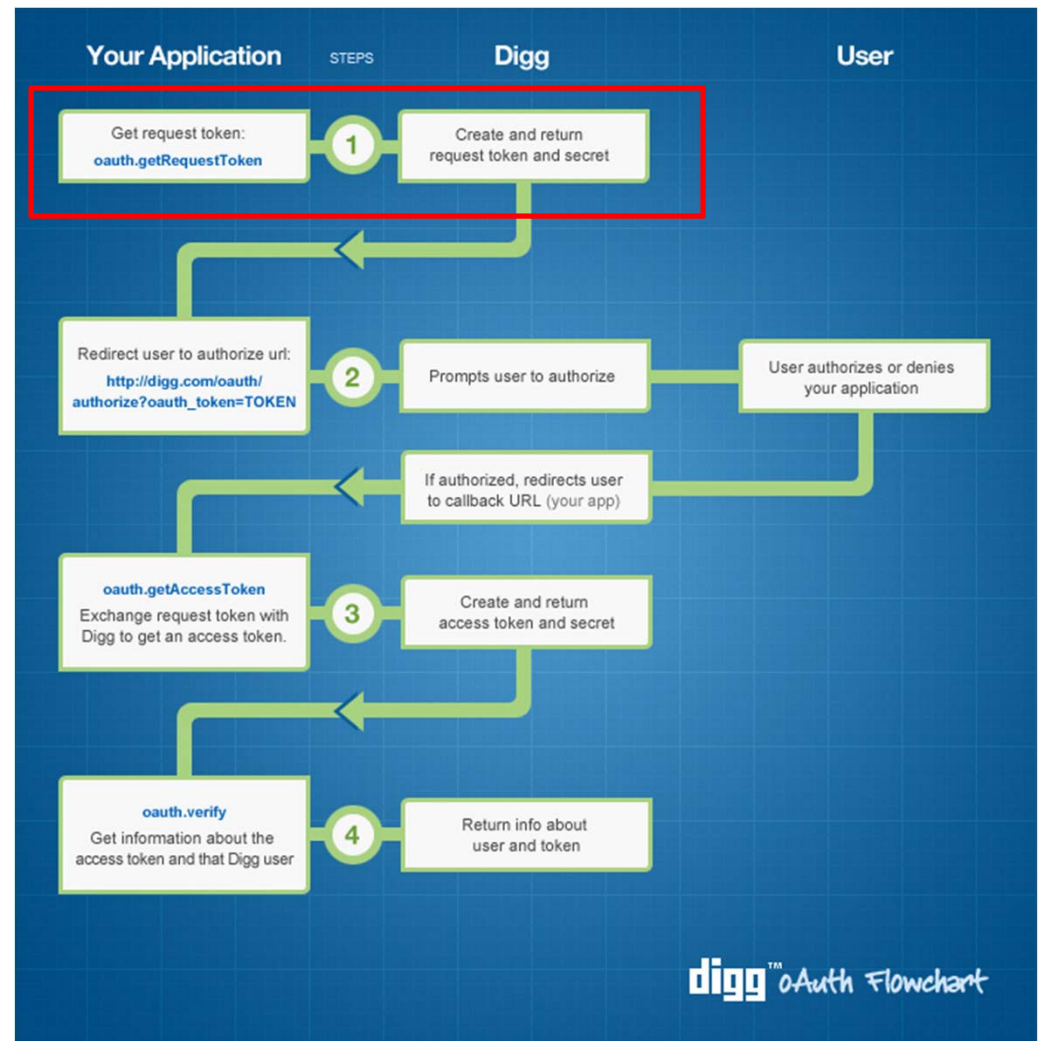
from <http://developers.digg.com/authentication>;  
similarly <https://developer.yahoo.com/oauth/guide/oauth-auth-flow.html>

# Step 0: User Connects to Consumer

- Consumer decides it needs user to authenticate to Service Provider

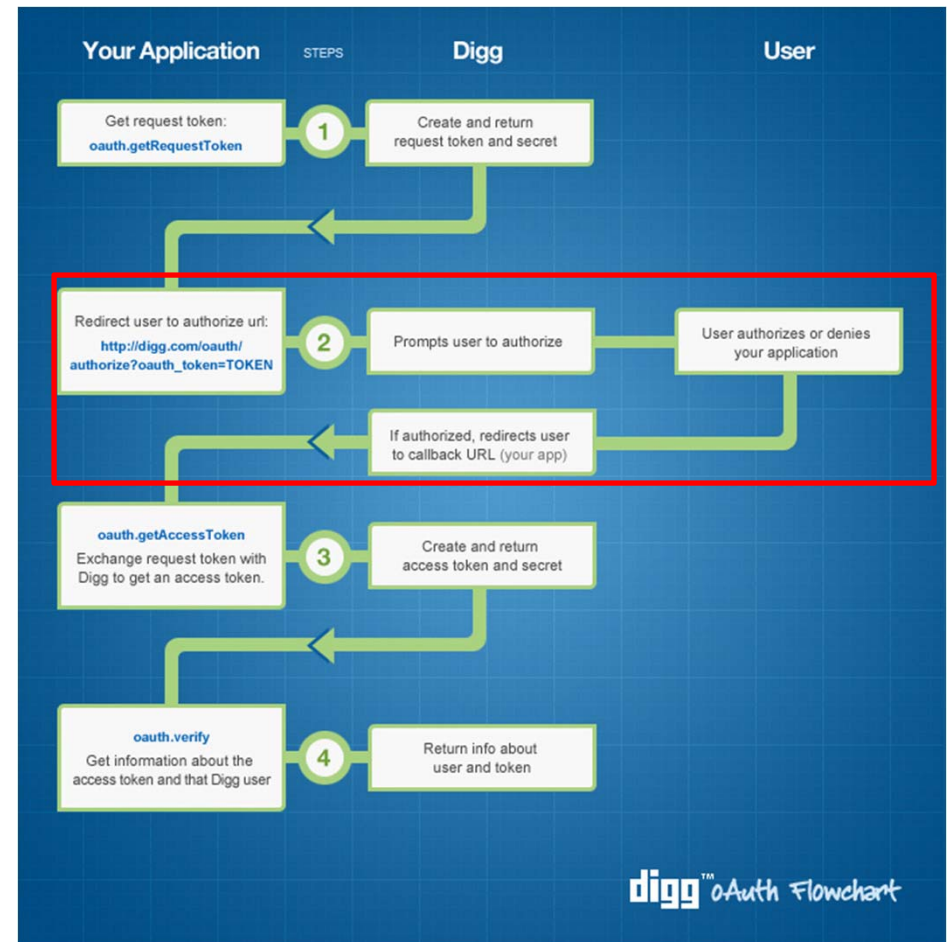
# Step 1: Consumer Connects to Service Provider

- Please create and return `oauth_token` and `oauth_secret`



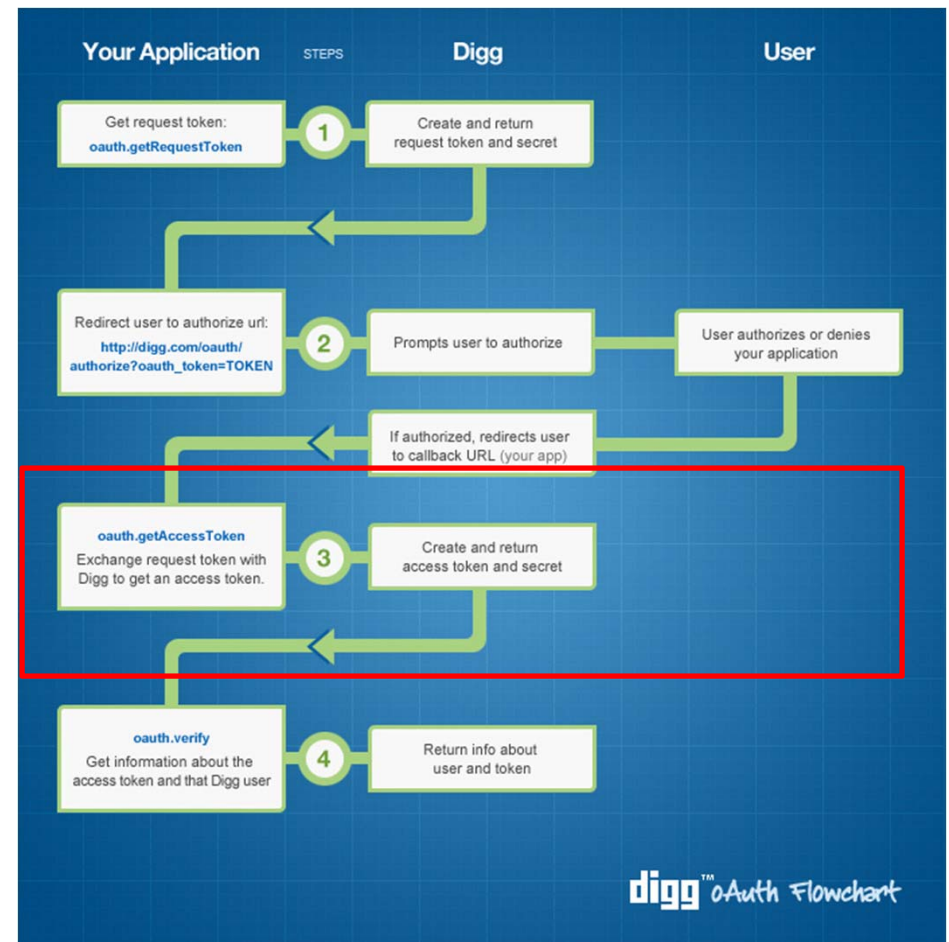
# Step 2: User Authenticates to Service Provider

- Redirect User to Digg URL
  - Encode within that Digg URL a URL at your site to redirect to after user authenticates
- When Digg redirects user to your site
  - authorization code also sent as query parameter



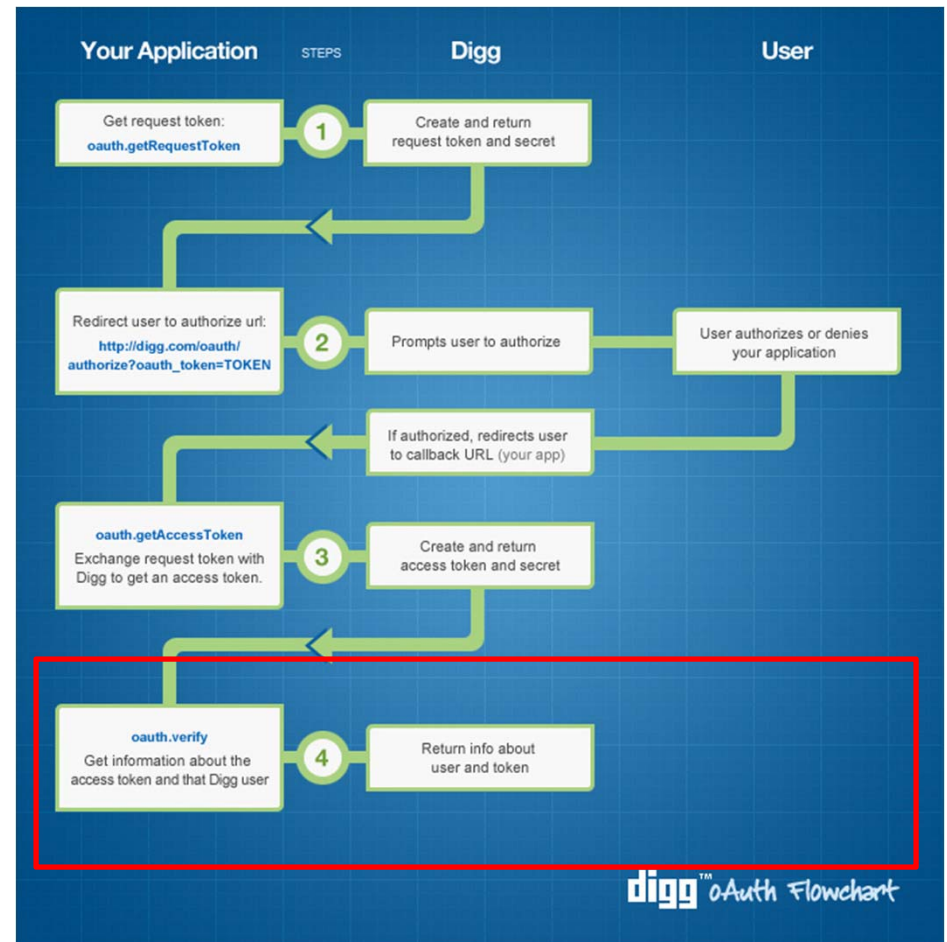
# Step 3: Consumer gets a token

- Consumer sends auth code to Service Provider
  - With some info about the Consumer (application)
- Get back an access token



## Step 4: use access token

- Consumer sends access token along with requests for data
  - Digg keeps track of access tokens it has issued
  - Digg decides permissions based on who the access token was for



# The Problem For Us

- We are not running a web server
- The step where service redirects user to our site, passing a token, won't work for us

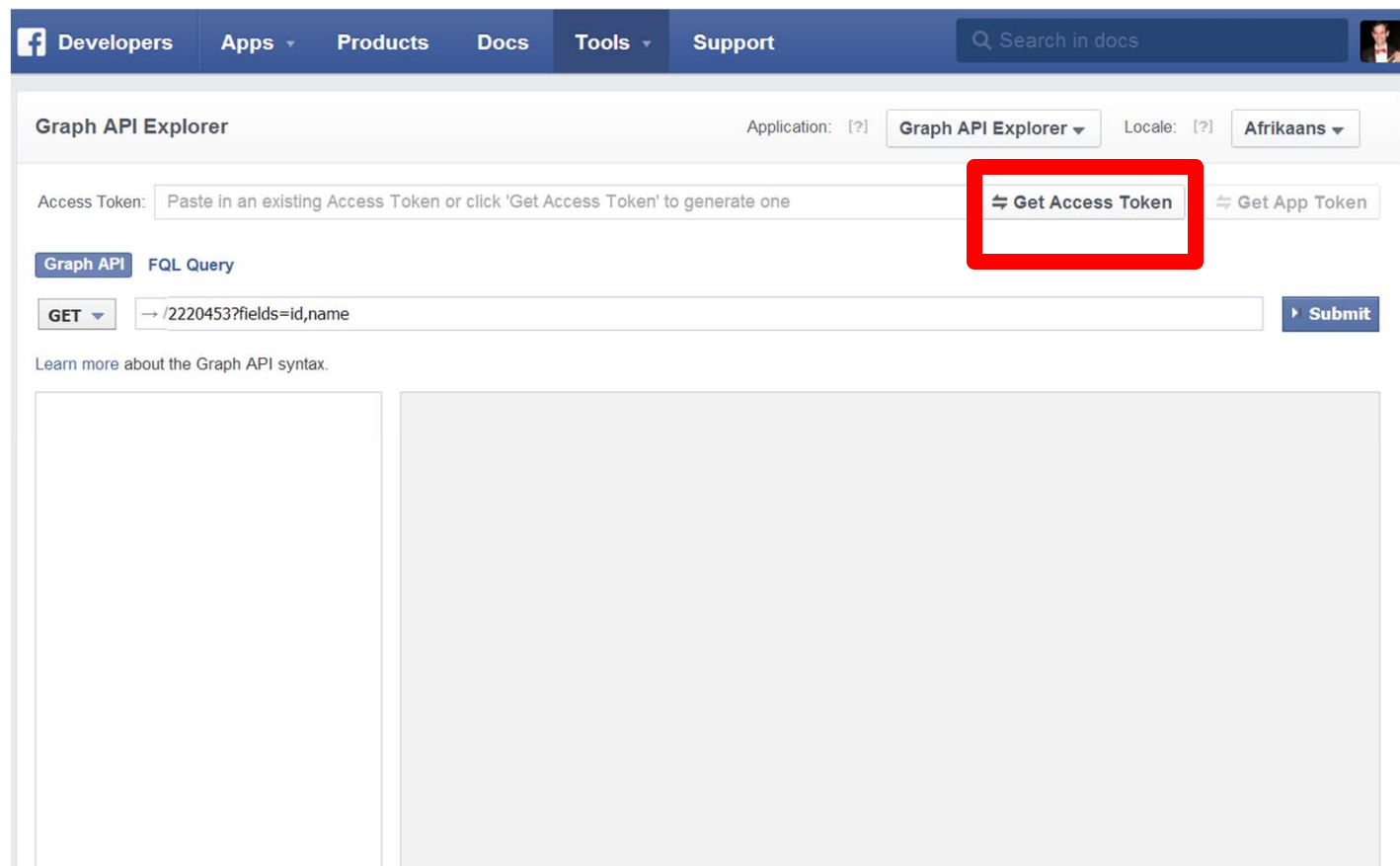
# Our Workarounds

- For FB
  - Use the FB developer tool to get the token manually
  - Paste it into your code (or terminal window)
- For Twitter (next week)
  - Open a web browser for the login; copy and paste token from the browser
- For your project
  - Read the API documentation for the particular service
  - If oAuth required, do something similar to what we've done for FB or Twitter



# Getting the Token

- <https://developers.facebook.com/tools/explorer>



The screenshot shows the Facebook Graph API Explorer interface. At the top, there is a navigation bar with links for Developers, Apps, Products, Docs, Tools, and Support. Below this, the main header includes the application name 'Graph API Explorer', a search bar, and a locale dropdown set to 'Afrikaans'. The 'Access Token' section contains a text input field with the placeholder 'Paste in an existing Access Token or click 'Get Access Token' to generate one'. To the right of this field are two buttons: 'Get Access Token' and 'Get App Token'. The 'Get Access Token' button is highlighted with a red rectangle. Below the 'Access Token' section, there are tabs for 'Graph API' and 'FQL Query'. The 'Graph API' tab is active, showing a 'GET' method and a query path '/2220453?fields=id,name'. A 'Submit' button is located to the right of the query input. At the bottom, there is a link to 'Learn more about the Graph API syntax.' and two large empty rectangular areas for displaying the results of the API call.

Graph API FQL Query

GET → /2220453...

Learn more about the Graph API

### Select Permissions

User Data Permissions Friends Data Permissions **Extended Permissions**

<input type="checkbox"/> ads_management	<input type="checkbox"/> ads_read	<input type="checkbox"/> create_event
<input type="checkbox"/> create_note	<input type="checkbox"/> email	<input checked="" type="checkbox"/> export_stream
<input type="checkbox"/> manage_friendlists	<input type="checkbox"/> manage_notifications	<input type="checkbox"/> manage_pages
<input type="checkbox"/> photo_upload	<input type="checkbox"/> publish_actions	<input type="checkbox"/> publish_checkins
<input type="checkbox"/> publish_stream	<input type="checkbox"/> read_friendlists	<input type="checkbox"/> read_insights
<input type="checkbox"/> read_mailbox	<input type="checkbox"/> read_page_mailboxes	<input type="checkbox"/> read_requests
<input checked="" type="checkbox"/> read_stream	<input type="checkbox"/> rsvp_event	<input type="checkbox"/> share_item
<input type="checkbox"/> sms	<input type="checkbox"/> status_update	<input type="checkbox"/> video_upload
<input type="checkbox"/> xmpp_login		

Basic permissions included by default.

**Get Access Token** Clear Cancel

# Cut and paste the access token

Graph API Explorer Application: [?] Graph API Explorer ▾ Locale: [?]

Access Token: CAACEdEose0cBAAEdQFNKO7Mox4ADyGzzdtnhnIHWtrQSI4RD2Nah51wLN1C1a ✕ Debug ⇌ Get Access Token

**Graph API** FQL Query

**GET** ▾ → /me

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

See `fboauth.py` for sample code

# The facebook module

- Defines class GraphAPI
- `__init__` takes an auth key
- Method `get_object()` takes a string
  - String is in format that FB defines
    - `"/me"` gets info about current user
    - `"/me/feed"` gets their recent posts
  - `get_object` method does a lot of stuff behind the scenes
    - Calls `requests.get`
    - Passes the `auth_key` in the way FB wants
    - Gets the response
    - Calls `json.loads` on it
    - Returns a dictionary

# But You Have To Get Your Computer Prepped to Use It

- Install the FB module
- Install pip [already done]
- Use pip to install the requests module [already done]

# Use pip to Install requests (should have done this already)

- On Mac
  - `pip install requests`
- On Windows
  - `/c/Python27/Scripts/pip install requests`
- Note: if you get permission errors during installation, try running the install commands with a prefix of `sudo` (super user do)
  - E.g., `sudo pip install requests`

# Installing the FB module

- Pip install facebook-sdk
- On Windows
  - `/c/Python27/Scripts/pip install facebook-sdk`
- Note: if you get permission errors during installation, try running the install commands with a prefix of `sudo` (super user do)
  - E.g., `sudo pip install facebook-sdk`



# Now fboauth.py Should Run!

- Note that the `get_object()` method does a lot of things that we've done manually before
  - Calls `requests.get()`
  - Reads the response as text
  - Calls `json.loads()`
- What you get back is a python dictionary

# Exercises

- Print out the text of all the messages returned in the feed
- Print out the names of everyone who left a comment on any of the posts