Social Media Apps Programming

Twitter API

1051SMAP13
TLMXM1A (8648) (M2143) (Fall 2016)
(MIS MBA) (2 Credits, Elective) [Full English Course]
Wed 8,9 (15:10-17:00) B310

Min-Yuh Day, Ph.D.
Assistant Professor
Department of Information Management
Tamkang University

http://mail.tku.edu.tw/myday

2016-12-28
## Course Schedule (1/3)

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2016/09/14</td>
<td>Course Orientation and Introduction to Social Media and Mobile Apps Programming</td>
</tr>
<tr>
<td>2</td>
<td>2016/09/21</td>
<td>Introduction to Android / iOS Apps Programming</td>
</tr>
<tr>
<td>3</td>
<td>2016/09/28</td>
<td>Developing Android Native Apps with Java (Eclipse) (MIT App Inventor)</td>
</tr>
<tr>
<td>4</td>
<td>2016/10/05</td>
<td>Developing iPhone / iPad Native Apps with Swift (XCode)</td>
</tr>
<tr>
<td>5</td>
<td>2016/10/12</td>
<td>Mobile Apps using HTML5/CSS3/JavaScript</td>
</tr>
<tr>
<td>6</td>
<td>2016/10/19</td>
<td>jQuery Mobile</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>2016/10/26</td>
<td>Create Hybrid Apps with PhoneGap</td>
</tr>
<tr>
<td>8</td>
<td>2016/11/02</td>
<td>jQuery Mobile/PhoneGap</td>
</tr>
<tr>
<td>9</td>
<td>2016/11/09</td>
<td>jQuery Mobile/PhoneGap</td>
</tr>
<tr>
<td>10</td>
<td>2016/11/16</td>
<td>Midterm Exam Week (Midterm Project Report)</td>
</tr>
<tr>
<td>11</td>
<td>2016/11/23</td>
<td>Case Study on Social Media Apps Programming and Marketing in Google Play and App Store</td>
</tr>
<tr>
<td>12</td>
<td>2016/11/30</td>
<td>Invited Talk [B302b]: Challenges in Natural Language Processing: Question Answering and Dialog System Invited Speaker: Prof. Yoshinobu Kano, Associate Professor, Faculty of Informatics, Shizuoka University, Japan</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>13</td>
<td>2016/12/07</td>
<td>Google Cloud Platform</td>
</tr>
<tr>
<td>14</td>
<td>2016/12/14</td>
<td>Google App Engine and Google Map API</td>
</tr>
<tr>
<td>15</td>
<td>2016/12/21</td>
<td>Facebook API (Facebook JavaScript SDK) (Integrate Facebook with iOS/Android Apps)</td>
</tr>
<tr>
<td>16</td>
<td>2016/12/28</td>
<td>Twitter API</td>
</tr>
<tr>
<td>17</td>
<td>2017/01/04</td>
<td>Final Project Presentation</td>
</tr>
<tr>
<td>18</td>
<td>2017/01/11</td>
<td>Final Exam Week (Final Project Presentation)</td>
</tr>
</tbody>
</table>
Outline

• Twitter Developers
  – Twitter Platform Objects
• Twitter for Websites
• Twitter Search API
• Twitter REST API
• Twitter Streaming API
https://twitter.com/search?src=typd&q=Starbucks
What can your business do...in just 140 characters?

http://www.youtube.com/watch?v=BGirUZq1WtQ
An espresso favorite—all dressed up just for you. The new #TuxedoMocha. Here ‘till the new year. 🎉
https://www.facebook.com/Starbucks
Twitter vs. Facebook

https://twitter.com/Starbucks/

https://www.facebook.com/pg/Starbucks/likes/?ref=page_internal
Documentation

Getting Started
With the Twitter Platform is easy. Jump right into the API resource documentation, explore the developer console, or manage your apps. Have a question? Read the FAQ. Don’t know what to build? Check out our Case Studies.

A field guide to Twitter Platform objects
Explore the Twitter platform’s variety of flora and fauna with this field guide to the most frequently observed wild objects.

Twitter cards

Embedded Timelines
Tweets from @twitter/more-twitter-accounts
Embeddable timelines make it possible to display public timelines on your website with a single line of code.

Embedded Tweets
A field guide to Twitter Platform objects

Like any ecosystem, the Twitter platform has a variety of flora and fauna. Use this field guide to better understand the most frequently observed wild objects.

https://dev.twitter.com/docs/platform-objects
Tweets

Tweets are the basic atomic building block of all things Twitter.

Source: https://dev.twitter.com/docs/platform-objects
Users

Users can be anyone or anything.

Source: https://dev.twitter.com/docs/platform-objects
Entities

Entities provide metadata and additional contextual information about content posted on Twitter.

Source: https://dev.twitter.com/docs/platform-objects
Places

Places are specific, named locations with corresponding geo coordinates.

Source: [https://dev.twitter.com/docs/platform-objects](https://dev.twitter.com/docs/platform-objects)
Tweets

Tweets are the basic atomic building block of all things Twitter. Users tweet Tweets, also known more generically as "status updates." Tweets can be embedded, replied to, favorited, unfavorited and deleted.

Field Guide

Consumers of Tweets should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a null field, an empty set, and the absence of a field as the same thing. Please note that Tweets found in Search results vary somewhat in structure from this document.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotations</td>
<td>Object</td>
<td>Unused. Future/beta home for status annotations.</td>
</tr>
</tbody>
</table>

Related API Resources

- GET favorites
- GET search
Tweets

Field Guide

Consumers of Tweets should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a nulled field, an empty set, and the absence of a field as the same thing. Please note that Tweets found in Search results vary somewhat in structure from this document.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotations</td>
<td>Object</td>
<td>Unused. Future/beta home for status annotations.</td>
</tr>
<tr>
<td>contributors</td>
<td>Collection of</td>
<td>Nullable. An collection of brief user objects (usually only one) indicating users who contributed to the authorship of the tweet, on behalf of the official tweet author. Discussion</td>
</tr>
<tr>
<td>coordinates</td>
<td>Coordinates</td>
<td>Nullable. Represents the geographic location of the Tweet as reported by the user or client application. The inner coordinates array is formatted as geoJSON (longitude first, then latitude). Example:</td>
</tr>
</tbody>
</table>

```
"coordinates":
{
  "coordinates":
  [
    -75.14310264,
    40.05701649
  ],
  "type": "Point"
}
```

Related API Resources

- GET favorites
- GET search
- GET statuses/home_timeline
- GET statuses/mentions
- GET statuses/retweeted_by_me
- GET statuses/retweeted_to_me
- GET statuses/retweets/id
- GET statuses/retweets_of_me
- GET statuses/show/id
- GET statuses/user_timeline

Tags

- Tweets (28)
- Timeline (12)
- Entities (33)
- Timelines (16)
- field descriptions (9)
The user who posted this Tweet. Perspectival attributes embedded within this object are unreliable. See Why are embedded objects stale or inaccurate?

Example:

```json
{　'statuses_count': 3080,　'favourites_count': 22,
　'protected': false,　'profile_text_color': '#437792',
　'profile_image_url': '...',　'name': 'Twitter API',
　'profile_sidebar_fill_color': '#999999',　' listed_count': 9252,
　'followings': true, 'profile_background_tile': false,
　'utc_offset': '-28800', 'description': 'The Real Twitter API. I tweet about API changes, service issues and happily answer questions about Twitter and our API. Don’t get an answer? It’s on my website.', 'location': 'San Francisco, CA',
　'contributors_enabled': true, 'verified': true,
　'profile_link_color': '#0000ee', 'followers_count': 6658299,
　'url': 'http://dev.twitter.com', 'default_profile': false,
　'profile_sidebar_border_color': '#000000',
　'screen_name': 'twitterapi', 'default_profile_image': false,
　'notifications': false, 'display_url': null,
　'show_all_inline_media': false, 'geo_enabled': true,
　'profile_use_background_image': true, 'friends_count': 32,
　'id_str': '6251282', 'entities': {'hashtags': [], 'urls': []},
　'user_mentions': [], 'expanded_url': null, 'is_translator': false,
　'lang': 'en', 'time_zone': 'Pacific Time (US &amp; Canada)',
　'created_at': 'Wed, May 23 06:01:13 +0000 2007',
　'profile_background_color': '#eef2f7', 'id': 6251282,
　'follow_request_sent': false,
　'profile_background_image_url_https': '...',
　'profile_background_image_url': '...',
　'profile_image_url_https': '...'}
```

**withheld_in_countries**

Array of String

When present, indicates a list of uppercase two-letter country codes of this content is withheld from. See New Withheld Content Exists in API Responses.

Example:

```
[　'withheld_in_countries': ['US', 'BE', 'FR', 'DE']　]
```
Users

Updated on Mon, 2013-05-20 07:28

Users can be anyone or anything. They tweet, follow, create lists, have a home_timeline, can be mentioned, and can be looked up in bulk.

**twitterapi** Twitter API

The Real Twitter API. I tweet about API changes, service issues and happily answer questions about Twitter and our API. Don't get an answer? It's on my website.

http://dev.twitter.com

Followers **988,982** Following **33**

Field Guide

Consumers of Users should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a nulled field, an empty set, and the absence of a field as the same thing.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contributors_enabled</td>
<td>Boolean</td>
<td>Indicates that the user has an account with “contributor mode” enabled, allowing for Tweets issued by the user to be co-authored by another account. Read: write.</td>
</tr>
</tbody>
</table>

Related API Resources

- GET account/verify_credentials
- GET friends/following

https://dev.twitter.com/docs/platform-objects/users
Nullable. If possible, the user’s most recent tweet or retweet. In some circumstance, this data cannot be provided and this field will be omitted, null, or empty. Perspectiveal attributes within tweets embedded within users cannot always be relied upon. See Why are embedded objects stale or inaccurate?.

Example:

```json
"status": {
  "coordinates": null,
  "favorited": false,
  "truncated": false,
  "created_at": "Tue Apr 17 16:38:18 +0000 2012",
  "id_str": '1922909046646754304',
  "entities": {
    "urls": [
    ],
    "hashtags": [
    ],
    "user_mentions": [
    {
      "name": "Micah McVicker",
      "id_str": "166661446",
      "id": 166661446,
      "indices": [
        0,
        14
      ],
      "screen_name": "MicahMcVicker"
    }
    ],
  },
  "in_reply_to_user_id_str": '166661446',
  "contributors": null,
  "text": '@MicahMcVicker make sure you’re using include_rts=true and no other filters, then walking your timeline by since_id and max_id',
  "retweeted": false,
  "in_reply_to_status_id": "192290470427246594",
  "id": 1922909046646754304,
  "geo": null
}
Entities

Updated on Mon, 2013-12-16 15:46

Entities provide metadata and additional contextual information about content posted on Twitter. Entities are never divorced from the content they describe. In API v1.1, entities will be returned wherever Tweets are found in the API. Entities are instrumental in resolving URLs.

Read Entities in Twitter Objects for a more comprehensive guide to how entities are used throughout Twitter objects.

Field Guide

Consumers of Entities should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a nulled field, an empty set, and the absence of a field as the same thing.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hashtags</td>
<td>Array of Object</td>
<td>Represents hashtags which have been parsed out of the Tweet text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;hashtags&quot;:{&quot;indices&quot;:[32,36],&quot;text&quot;:&quot;lol&quot;}</td>
</tr>
<tr>
<td>media</td>
<td>Array of Object</td>
<td>Represents media elements uploaded with the Tweet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;media&quot;:{&quot;type&quot;:&quot;photo&quot;, &quot;sizes&quot;:{&quot;thumb&quot;:{&quot;h&quot;:150, &quot;w&quot;:150}, &quot;large&quot;:{&quot;h&quot;:238, &quot;w&quot;:226}, &quot;medium&quot;:{&quot;h&quot;:238, &quot;w&quot;:226}, &quot;small&quot;:{&quot;h&quot;:238, &quot;w&quot;:226}}}</td>
</tr>
</tbody>
</table>

Natural habitat

Entities are most often found within Tweets.

Tags

- Tweets (28)
- Entities (33)

https://dev.twitter.com/docs/platform-objects/entities
Entities

Field Guide

Consumers of Entities should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a nulled field, an empty set, and the absence of a field as the same thing.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hashtags</td>
<td>Array of Object</td>
<td>Represents hashtags which have been parsed out of the Tweet text. Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;hashtags&quot;: [{&quot;indices&quot;: [32, 36], &quot;text&quot;: &quot;lol&quot;}]</td>
</tr>
<tr>
<td>media</td>
<td>Array of Object</td>
<td>Represents media elements uploaded with the Tweet. Example:</td>
</tr>
<tr>
<td>urls</td>
<td>Array of Object</td>
<td>Represents URLs included in the text of a Tweet or within textual fields of a user object. Tweet Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;urls&quot;: [{&quot;indices&quot;: [32, 52], &quot;url&quot;: &quot;<a href="http://t.co/jowwrtzk">http://t.co/jowwrtzk</a>&quot;, &quot;display_url&quot;: &quot;youtube.com/watch?v=ong5sju2026&quot;, &quot;expanded_url&quot;: &quot;<a href="http://www.youtube.com/watch?v=ong5sju2026%22%7D">http://www.youtube.com/watch?v=ong5sju2026&quot;}</a>]</td>
</tr>
</tbody>
</table>

Tags

- Tweets (28)
- Entities (35)
- Mentions (20)
- t.co (43)
- field descriptions (9)
- hashtags (4)
- finding media (4)
- finding links (2)

https://dev.twitter.com/docs/platform-objects/entities
Places

Places are specific, named locations with corresponding geo coordinates. They can be attached to Tweets by specifying a place_id when tweeting. Tweets associated with places are not necessarily issued from that location but could also potentially be about that location. Places can be searched for. Tweets can also be found by place_id. See About Geo Place Attributes for more information.

Field Guide

Consumers of Places should tolerate the addition of new fields and variance in ordering of fields with ease. Not all fields appear in all contexts. It is generally safe to consider a nulled field, an empty set, and the absence of a field as the same thing.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>Contains a hash of variant information about the place. See About Geo Place Attributes.</td>
</tr>
<tr>
<td>bounding_box</td>
<td>Object</td>
<td>A bounding box of coordinates which encloses this place.</td>
</tr>
</tbody>
</table>

Natural habitat

Places can be found throughout the natural universe, but typically only appear attached to Tweets on Twitter.

Related API Resources

- GET geoid/place_id

https://dev.twitter.com/docs/platform-objects/places
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>country_code</td>
<td>String</td>
<td>Shortened country code representing the country containing this place.</td>
<td>'country_code': 'FR'</td>
</tr>
<tr>
<td>full_name</td>
<td>String</td>
<td>Full human-readable representation of the place's name.</td>
<td>'full_name': 'Paris, Paris'</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>ID representing this place. Note that this is represented as a string, not an integer.</td>
<td>'id': '7238f93a3e899af6'</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Short human-readable representation of the place's name.</td>
<td>'name': 'Paris'</td>
</tr>
<tr>
<td>place_type</td>
<td>String</td>
<td>The type of location represented by this place.</td>
<td>'place_type': 'city'</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>URL representing the location of additional place metadata for this place.</td>
<td>'url': '<a href="http://api.twitter.com/1/geo/id/7238f93a3e899af6.json">http://api.twitter.com/1/geo/id/7238f93a3e899af6.json</a>'</td>
</tr>
</tbody>
</table>

https://dev.twitter.com/docs/platform-objects/places
Twitter for Websites

Twitter Buttons

Select the best button for you, whether you want to distribute content, communicate with your followers, or discuss a topic using a #hashtag.

- Tweet (16.3K)
- Follow @twitter (10M+ followers)
- Tweet #LetsFly
- Tweet to @twitterapi

Create a button for your website

Read the documentation for the Tweet Button

Read the documentation for the Follow Button

Embedded Tweets

Tweets are dynamic and interactive media with authors, mentions, @people, #topics, pictures & videos. Now you can embed them on your website with just one line of code.

Let's Fly! blog.twitter.com/2011/12/lets-f...

1:43 AM - 8 Dec 2011

Let's Fly | Twitter Blogs

Update - 3:15pm PT More news: Read about enhanced profile pages, on the advertising blog, and embeddable Tweets, on the dev blog. Today we introduce a new version of Twitter. We've simplified...

Twitter @twitter

3,228 RETWEETS 1,488 FAVORITES

Learn how to embed a Tweet

https://dev.twitter.com/docs/twitter-for-websites
Follow Button

The Follow Button is a small widget which allows users to easily follow a Twitter account from any webpage. The Follow Button uses the same implementation model as the Tweet Button, and its integration is just as simple.

By using the Follow Button, you agree to the Developer Rules of the Road.

Overview

The Follow Button is a small widget which allows users to easily follow a Twitter account from any webpage. The Follow Button uses the same implementation model as the Tweet Button, and its integration is just as simple.
Follow Button

Overview
The Follow Button is a small widget which allows users to easily follow a Twitter account from any webpage. The Follow Button uses the same implementation model as the Tweet Button, and its integration is just as simple.

User Interaction
User interaction flow

https://dev.twitter.com/docs/follow-button
Twitter buttons

Add buttons to your website to help your visitors share content and connect on Twitter.

Choose a button

- Share a link
- Follow
- Hashtag
- Mention

Button options

- User: @iMyday
- Show username
- Large button
- Opt-out of tailoring Twitter
- Language: English

Preview and code

Try out your button, then copy and paste the code below into the HTML for your site.

```html
<a href="https://twitter.com/iMyday" class="twitter-follow-button">Follow @iMyday</a>
```

https://about.twitter.com/resources/buttons#follow
Follow @iMyday

<a href="https://twitter.com/iMyday" class="twitter-follow-button" data-show-count="false">Follow @iMyday</a>

<script>!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0],p=/^http:.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+'://platform.twitter.com/widgets.js';fjs.parentNode.insertBefore(js,fjs);}}(document, 'script', 'twitter-wjs');</script>
Test Twitter Button on jsbin.com
Follow @iMyday
Using the Twitter Search API

Updated on Wed, 2013-10-02 06:34

The Twitter Search API is part of Twitter's v1.1 REST API. It allows queries against the indices of recent or popular Tweets and behaves similarly to, but not exactly like the Search feature available in Twitter mobile or web clients, such as Twitter.com search.

Before getting involved, it's important to know that the Search API is focused on relevance and not completeness. This means that some Tweets and users may be missing from search results. If you want to match for completeness you should consider using a Streaming API instead.

A detailed reference on this API endpoint can be found at GET search/tweets.

How to build a query

The best way to build a query and test if it's valid and will return matched Tweets is to first try it at twitter.com/search. As you get a satisfactory result set, the URL loaded in the browser will contain the proper query syntax that can be reused in the API endpoint. Here's an example:

1. We want to search for tweets referencing @twitterapi account. First, we run the search on twitter.com/search. As you get a satisfactory result set, the URL loaded in the browser will contain the proper query syntax that can be reused in the API endpoint. Here's an example:
2. Check and copy the URL loaded. In this case, we got: https://twitter.com/search?q=%40twitterapi
3. Replace "https://twitter.com/search" with "https://api.twitter.com/1.1/search/tweets.json" and you will get: https://api.twitter.com/1.1/search/tweets.json?q=%40twitterapi
4. Execute this URL to do the search in the API

Please note that now API v1.1 requires that the request must be authenticated, check Authentication & Authorization documentation for more details on how to do it. Also note that the search results at twitter.com may return historical results while the Search API usually only serves tweets from the past week.

Related open issues

Search API: "-filter:links" not affecting results

Related Questions

- Why are the Tweets I'm looking for not in Twitter Search, the Search API, or Search widgets?

Related Case Studies

- Esri enriches maps with Tweets and the Streaming API

Tags

- Search (34)
- finding tweets (49)

https://dev.twitter.com/docs/using-search
The Twitter REST API

REST API version 1.1

The most recent version of the Twitter REST API.

- API v1.1 Resources
- Rate Limiting in API v1.1
- Authenticating
- Announcement

REST API version 1

Version 1 of the REST API is now deprecated and will cease functioning in the coming months. Migrate to version 1.1 today.

- Review the deprecated version 1 API

https://dev.twitter.com/docs/api
The Streaming APIs

Overview

The set of streaming APIs offered by Twitter give developers low latency access to Twitter’s global stream of Tweet data. A proper implementation of a streaming client will be pushed messages indicating Tweets and other events have occurred, without any of the overhead associated with polling a REST endpoint.

Twitter offers several streaming endpoints, each customized to certain use cases.

| Public streams | Streams of the public data flowing through Twitter. Suitable for following specific users or topics, and data mining. |
| User streams | Single-user streams, containing roughly all of the data corresponding with a single user’s view of Twitter. |
| Site streams | The multi-user version of user streams. Site streams are intended for servers which must connect to Twitter on behalf of many users. |

Differences between Streaming and REST

Connecting to the streaming API requires keeping a persistent HTTP connection open. In many cases this involves thinking about your application differently than if you were interacting with the REST API. For an example, consider a web application which accepts user requests, makes one or more requests to Twitter’s API, then formats and prints the result to the user, as a response to the user’s initial request:

https://dev.twitter.com/docs/streaming-apis
The REST API provides simple interfaces for most Twitter functionality.

https://dev.twitter.com/docs
Twitter Streaming API

The Streaming API is a family of powerful real-time APIs for Tweets and other social events.

https://dev.twitter.com/docs
Differences between Streaming and REST API

Twitter REST API

https://dev.twitter.com/docs/streaming-apis
Differences between Streaming and REST API

Twitter Streaming API

https://dev.twitter.com/docs/streaming-apis
Exploring the Twitter API

Updated on Thu, 2013-01-17 07:58

Explore the API on your own machine using Twitter for Mac’s developer console or our command-line Twurl tool. Explore the API here by using this console provided by @apigee.

https://dev.twitter.com/console
Create a new app

My applications

Looks like you haven't created any applications yet!

Create a new application

https://dev.twitter.com/apps
Create a new app

Create an application

**Application Details**

**Name:**

iMydayMobileApp

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

**Description:**

iMyday Mobile App

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

**Website:**

http://mail.tku.edu.tw/myday/app

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don't have a URL yet, just put a placeholder here but remember to change it later.)

**Callback URL:**

Where should we return after successfully authenticating? For *Anywhere applications*, only the domain specified in the callback will be used. *OAuth 1.0a* applications should explicitly specify their `oauth_callback` URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

https://dev.twitter.com/apps/new
Create a new app

Rules of the Road

Twitter maintains an open platform that supports the millions of people around the world who are sharing and discovering what’s happening now. We want to empower our ecosystem partners to build valuable businesses around the information flowing through Twitter. At the same time, we aim to strike a balance between encouraging interesting development and protecting both Twitter’s and users’ rights.

So, we’ve come up with a set of Developer Rules of the Road (“Rules”) that describes the policies and philosophy around what type of innovation is permitted with the content and information shared on Twitter.

The Rules will evolve along with our ecosystem as developers continue to innovate and find new, creative ways to use the Twitter API, so please check back periodically to see the current version. Don’t do anything prohibited by the Rules and talk to us if you think we should make a change or give you an exception.

If your application will eventually need more than 1 million user tokens, or you expect your embedded Tweets and embedded timelines to exceed 10 million daily impressions, you will need to talk to us directly about your access to the Twitter API as you may be subject to additional terms. Furthermore, applications that attempt to replicate Twitter’s core user experience (as described in Section 1.5 below) will need our permission to

---

Yes, I agree

By clicking the “I Agree” button, you acknowledge that you have read and understand this agreement and agree to be bound by its terms and conditions.

CAPTCHA

This question is for testing whether you are a human visitor and to prevent automated spam submissions.

Goshen

Create your Twitter application

https://dev.twitter.com/apps/new
Create a new app

iMydayMobileApp

Organization
Information about the organization or company associated with your application. This information is optional.

Organization: None
Organization website: None

OAuth settings
Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level: Read-only
About the application permission model

Consumer key: LKhzPN3e5JH600
Consumer secret: TMszvTNnTNDrUnY7hb
Create a new app

OAuth settings
Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

<table>
<thead>
<tr>
<th>Access level</th>
<th>Read-only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consumer key</td>
<td>1KHzfPN3oJ8EOQ</td>
</tr>
<tr>
<td>Consumer secret</td>
<td>7NswzTNuTNDvVf6jhb52I</td>
</tr>
<tr>
<td>Request token URL</td>
<td><a href="https://api.twitter.com/oauth/request_token">https://api.twitter.com/oauth/request_token</a></td>
</tr>
<tr>
<td>Authorize URL</td>
<td><a href="https://api.twitter.com/oauth/authorize">https://api.twitter.com/oauth/authorize</a></td>
</tr>
<tr>
<td>Access token URL</td>
<td><a href="https://api.twitter.com/oauth/access_token">https://api.twitter.com/oauth/access_token</a></td>
</tr>
<tr>
<td>Callback URL</td>
<td>None</td>
</tr>
<tr>
<td>Sign in with Twitter</td>
<td>No</td>
</tr>
</tbody>
</table>

Your access token
Use the access token string as your "oauth_token" and the access token secret as your "oauth_token_secret" to sign requests with your own Twitter account. Do not share your oauth_token_secret with anyone.

| Access token       | 227716153---Ky65DQp715thnDTC |
| Access token secret| gTTTZ6CG8IApgAP31q7k      |
| Access level       | Read-only                   |
Create a new app

Consumer key IKHzfFM3ejM6O********
Consumer secret TMsvzTNuTNDrUnY7hb5ZIarXqZDnsKW********
Request token URL https://api.twitter.com/oauth/request_token
Authorize URL https://api.twitter.com/oauth/authorize
Access token URL https://api.twitter.com/oauth/access_token
Update Twitter’s App Settings

iMydayMobileApp

Application Details

Name: iMydayMobileApp
Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description: iMyday Mobile App
Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website: http://mail.tku.edu.tw/myday/app
Your application’s publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.

Application Icon
Change icon:
Update Twitter’s App Settings

Application Icon

Change icon:

Maximum size of 700k. JPG, GIF, PNG.

Application Type

Access:

- Read only
- **Read and Write**
- Read, Write and Access direct messages

What type of access does your application need? Note: @Anywhere applications require read & write access.
Find out more about our Application Permission Model.

Callback URL:

Where should we return after successfully authenticating? For @Anywhere applications, only the domain specified in the callback will be used. OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

- **Allow this application to be used to Sign in with Twitter**

When enabled your application can be used to “Sign in with Twitter”. When disabled your application will not be able to use /oauth/authorize and any request to it will instead redirect the user to /oauth/authorize

Organization

Organization name:
Update Twitter’s App Settings

Access:
- Read only
- Read and Write
- Read, Write and Access direct messages

What type of access does your application need? Note: @Anywhere applications require read & write access. Find out more about our Application Permission Model.

Callback URL:

Where should we return after successfully authenticating? For @Anywhere applications, only the domain specified in the callback will be used. OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Allow this application to be used to Sign in with Twitter

When enabled your application can be used to “Sign in with Twitter”. When disabled your application will not be able to use /oauth/authorize and any request to it will instead redirect the user to /oauth/authorize

Organization

Organization name:

The organization or company behind this application, if any.

Organization website:

The organization or company behind this application’s web page, if any.

Update this Twitter application’s settings
OAuth Settings Updated

OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

<table>
<thead>
<tr>
<th>Access level</th>
<th>Read and write</th>
</tr>
</thead>
</table>

Consumer key: 1KHsFPj3eJH60...
Consumer secret: TMavzTNaTNDrUmY7hb52Ia...
Request token URL: https://api.twitter.com/oauth/request_token
Authorize URL: https://api.twitter.com/oauth/authorize
Access token URL: https://api.twitter.com/oauth/access_token
Callback URL: None
Sign in with Twitter: Yes

Your access token

Use the access token string as your "oauth_token" and the access token secret as your "oauth_token_secret" to sign requests with your own Twitter account. Do not share your oauth_token_secret with anyone.

Access token: 22716...Ky65DQp715thnDTO12G...
Access token secret: gTTT26CDg5i4pgAP81...
Access level: Read-only

Recreate my access token
Twitter REST API v1.1 Resources

REST API v1.1 Resources

Timelines
Timelines are collections of Tweets, ordered with the most recent first.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET statuses/mentions_timeline</td>
<td>Returns the 20 most recent mentions (tweets containing a user’s @screen_name) for the authenticating user. The timeline returned is the equivalent of the one seen when you view your mentions on twitter.com. This method can only return up to 800 tweets. See Working with Timelines for...</td>
</tr>
<tr>
<td>GET statuses/user_timeline</td>
<td>Returns a collection of the most recent Tweets posted by the user indicated by the screen_name or user_id parameters. User timelines belonging to protected users may only be requested when the authenticated user either “owns” the timeline or is an approved follower of the owner. The timeline...</td>
</tr>
<tr>
<td>GET statuses/home_timeline</td>
<td>Returns a collection of the most recent Tweets and retweets posted by the authenticating user and the users they follow. The home timeline is central to how most users interact with the Twitter service. Up to 800 Tweets are obtainable on the home timeline. It is more volatile for users that follow...</td>
</tr>
<tr>
<td>GET statuses/retweets_of_me</td>
<td>Returns the most recent tweets authored by the authenticating user that have been retweeted by others. This timeline is a subset of the user’s GET statuses/user_timeline. See Working with Timelines for instructions on traversing timelines.</td>
</tr>
</tbody>
</table>

Tweets
Tweets are the atomic building blocks of Twitter, 140-character status updates with additional associated metadata. People tweet for a variety of reasons about a multitude of topics.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

https://dev.twitter.com/docs/api/1.1
GET search/tweets

Returns a collection of relevant Tweets matching a specified query.

Please note that Twitter’s search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface.

In API v1.1, the response format of the Search API has been improved to return Tweet objects more similar to the objects you’ll find across the REST API and platform. You may need to tolerate some inconsistencies and variance in perspectival values (fields that pertain to the perspective of the authenticating user) and embedded user objects.

To learn how to use Twitter Search effectively, consult our guide to Using the Twitter Search API. See Working with Timelines to learn best practices for navigating results by since_id and max_id.

Resource URL
https://api.twitter.com/1.1/search/tweets.json

Parameters

- q
  - required
  - A UTF-8, URL-encoded search query of 1,000 characters maximum, including operators. Queries may additionally be limited by complexity.
  - Example Values: @noradio

- geocode
  - optional
  - Returns tweets by users located within a given radius of the given latitude/longitude. The location is preferentially taking from the Geotagging API, but will fall back to their Twitter profile. The parameter value is specified by

Related open issues

- Search API: Count param does not work when result type is set to mixed

Resource Information

- Rate Limited?: Yes
- Requests per rate limit window: 180/user, 450/app
- Authentication: Required
- Response Formats: json
- HTTP Methods: GET
- Resource family: search
- Response Object: Tweets
- API Version: v1.1

https://dev.twitter.com/docs/api/1.1/get/search/tweets
GET search/tweets

Example Request

Use the OAuth tool in this page sidebar to generate the OAuth signature for this request.

GET

https://api.twitter.com/1.1/search/tweets.json?q=freebandnames&since_id=24012619984051000&max_id=250126199840518145&result_type=mixed&count=4

```json
{
  "statuses": [
    {
      "coordinates": null,
      "favorited": false,
      "truncated": false,
      "created_at": "Mon Sep 24 03:35:21 +0000 2012",
      "id_str": "250075927172759552",
      "entities": {
        "urls": [
          {
            "hashtags": [
              {
                "text": "freebandnames",
                "indices": [
                  20,
                  34
                ]
              }
            ]
          },
          "user_mentions": [
            
          ],
          "in_reply_to_user_id_str": null,
          "contributors": null,
          "text": "Aggressive Ponytail #freebandnames",
          "metadata": {
            "iso_language_code": "en",
            "result_type": "recent"
          },
          "request_count": 0
        ]
      }
    }
  ]
}
```
GET search/tweets

https://api.twitter.com/1.1/search/tweets.json?q=%23freebandnames&since_id=24012619984051000&max_id=250126199840518145&result_type=mixed&count=4
OAuth Tool

via the API to geocode arbitrary locations; however you can use this geocode parameter to search near geocodes directly. A maximum of 1,000 distinct "sub-regions" will be considered when using the radius modifier.

Example Values: 37.781157,-122.398720,lmi

lang
optional

Restricts tweets to the given language, given by an ISO 639-1 code. Language detection is best-effort.

Example Values: eu

locale
optional

Specify the language of the query you are sending (only ja is currently effective). This is intended for language-specific consumers and the default should work in the majority of cases.

Example Values: ja

result_type
optional

Optional. Specifies what type of search results you would prefer to receive. The current default is "mixed." Valid values include:

* mixed: Include both popular and real time results in the response.
* recent: return only the most recent results in the response
* popular: return only the most popular results in the response.

Example Values: mixed, recent, popular

count
optional

The number of tweets to return per page, up to a maximum of 100. Defaults to 15. This was formerly the "rpp" parameter in the old Search API.

Example Values: 100

until
optional

Returns tweets generated before the given date. Date should be formatted as YYYY-MM-DD. Keep in mind that the search index may not go back as far as the date you specify here.

Example Values:

Related Documentation
- GET users/search
- Using the Twitter Search API
- GET geo/search
- Finding Tweets about Places

Related Questions
- Why are the Tweets I'm looking for not in Twitter Search, the Search API, or Search widgets?

Tags
- Search (34)
- finding tweets (49)

https://dev.twitter.com/docs/api/1.1/get/search/tweets#oauth-tool
OAuth Tool

This tool will generate the OAuth signature for the example request

Generate OAuth signature

https://dev.twitter.com/docs/api/1.1/get/search/tweets#oauth-tool
See OAuth signature for this request
OAuth Signing Results

Important: This will only be valid for a few minutes. Also remember the cURL command will actually execute the request.

Signature base string

Authorization header
Authorization: OAuth oauth_consumer_key="1KHzeFM3eDjJ", oauth_nonce="121df0a26f314c57c", oauth_signature="2q5vnDcf1276pC", oauth_signature_method="HMAC-SHA1", oauth_timestamp="1388735", oauth_token="227161---Ky65DQp7158hnD7012GZUIPej", oauth_version=1.0"

cURL command
curl --get 'https://api.twitter.com/1.1/search/tweets.json' --data 'count=50&max_id=0&0a26f314c57c&&signature_method=HMAC-SHA1, oauth_timestamp="13869716", oauth_token="227161---Ky65DQp7158hnD7012GZUIPej", oauth_version=1.0" --verbose
By using Web Intents, you agree to the [Developer Rules of the Road](https://dev.twitter.com/docs/intents).

Make it easy to bring interactivity to Tweets that you display on the web.

Web Intents provide optimized flows for working with Tweets & Twitter Users: Tweet, Reply, Retweet, Favorite, and Follow. They make it possible for users to interact with Twitter content in the context of your site, without leaving the page or having to authorize a new app just for the interaction. Web Intents are mobile friendly, and super easy to implement.
Web Intents

Get Started

Web intents can be invoked flexibly through a light combination of Javascript and HTML and are meant to be opened in a new window.

The easiest way to use intents is to include this SCRIPT tag on any web page you wish to invoke an intent. If you’ve already setup the Tweet Button, you’re already prepared for Web Intents.

When combined with standard anchor tags and familiar iconography like the examples below, this Javascript will automatically open a window of the appropriate size when clicked. You only need to load platform.twitter.com/widgets.js once.

```html
1. <script type="text/javascript" src="/platform.twitter.com/widgets.js"></script>
2. <p><a href="https://twitter.com/intent/tweet?in_reply_to=51113028241989932"&gt;Reply</a></p>
```

Meet the Web Intents

Tweet Intents

- Tweet or Reply to a Tweet
- Retweet a Tweet
- Favorite a Tweet

User Intents

- Follow or Display a User

Tweet or Reply to a Tweet

https://twitter.com/intent/tweet

This intent makes it easy for a user to tweet nearly anything from your site with a stylish #newtwitter-inspired Tweet Composer. Though easy to implement with minimal configuration, you can provide optional parameters to support a variety of scenarios including allowing for careful pre-composition of Tweet text, adding @mentioning, link sharing, and...
JavaScript Interfaces for Twitter for Websites

By using Twitter Javascript, you agree to the Developer Rules of the Road.

If you're integrating your site with Twitter using Twitter for Websites and Web Intents, you can enhance your application using JavaScript functions and events.

Twitter for Websites products—Tweet buttons, Follow buttons, embedded Tweets and timelines—are all loaded using a JavaScript utility named widgets-js. When adding a Twitter widget to your page, this JavaScript file is included in the HTML embed code, or you can directly include http://platform.twitter.com/widgets.js in your page, using the following code:

```
1. window.twtrr = (function (d,s,id) {
2.    var t, js, fjs = d.getElementsByTagName(s)[0];
3.    if (d.getElementsByTagName(s).length === 0) return;
4.    js = d.createElement(s); js.id = id;
5.    js.src = 'https://platform.twitter.com/widgets.js';
6.    fjs.parentNode.insertBefore(js, fjs);
7.    return window.twtrr = || (t = { _e: []}, ready: function(f){ t._e.push(f) } ));
8. })(document, 'script', 'twitter-wjs');
```

By default, widgets-js will find mark-up in a page and convert basic, functional mark-up into rich interactive widgets. In addition, there are a number of functions of widgets-js that allow developers to work with Twitter content dynamically, after the page has loaded:

- Late initialization of widgets
- Factory methods for dynamic widgets

https://dev.twitter.com/docs/intents/events
JavaScript Interfaces for Twitter for Websites

1. `window.twttr = (function(d,s,id) {`
2. `var t, js, fjs = d.getElementsByTagName(s)[0];`
3. `if (d.getElementById(id)) return; js = d.createElement(s); js.id = id;`
4. `js.src = "https://platform.twitter.com/widgets.js"; fjs.parentNode.insertBefore(js, fjs);`
5. `return window.twttr || (t = { _e: [], ready: function(f){ t._e.push(f) } });`
6. `}(document, "script", "twitter-wjs"));`

https://dev.twitter.com/docs/intents/events
Twitter API Getting Started

Updated on Sat, 2012-08-25 11:22

Twitter is an information network and communication mechanism that produces more than 200 million tweets a day. The Twitter platform offers access to that corpus of data, via our APIs. Each API represents a facet of Twitter, and allows developers to build upon and extend their applications in new and creative ways. It's important to note that the Twitter APIs are constantly evolving, and developing on the Twitter Platform is not a one-off event.

Twitter for Websites

Twitter for Websites (TfW) is a suite of products that enables websites to easily integrate Twitter. TfW is ideal for site developers looking to quickly and easily integrate very basic Twitter functions. This includes offerings like the Tweet button, which lets a user tweet about something she found engaging on your site to all of her friends. Her friends see that tweet, click through to your site, and some percentage of them will tweet that same content to their friends - a great distribution loop. Likewise, the Follow button allows a user to quickly follow your account back on Twitter. That particular user will then see updates from your account funneled back into her stream, creating a new channel of engagement with your user base.

Search API

The Search API designed for products looking to allow a user to query for Twitter content. This may include finding a set of tweets with specific keywords, finding tweets referencing a specific user, or finding tweets from a particular user. This API will also provide your product access to data around Trends. If you need to query the Search API at extreme velocities and are hitting rate limits, then you should be working with the Streaming API. The latter API is covered in detail below.

REST API

The REST API enables developers to access some of the core primitives of Twitter including timelines, status updates, and user information. Additionally, the REST API can be used to perform actions on behalf of a logged-in user.
REST API SSL certificate updates

At the end of 2013, all Browsers and Certificate Authorities will no longer support 1024 bits RSA certificates to be compliant to National Institute of Standards and Technology (NIST) guidelines.

The SSL certificate currently used on api.twitter.com is signed with the older Verisign G2 root CA certificate.

Due to NIST guidelines, api.twitter.com will change to a new certificate on Dec 10th, 2013. The new certificate will be signed with VeriSign Class 3 Secure Server CA - G3, which has the 2048 bits key length needed to meet recommended security levels.

This means that all HTTP clients used by your application must trust the new root certificate, otherwise you won’t be able to connect in the API. To ensure proper SSL certificate verification across all of Twitter’s services, your software should include all Verisign Root Certificates in its CAFile or other respective keystores. The root certificates are available at the following link:

- Verisign (https://www.symanetcorp.com/page.jsp?id=roots)

For more guidelines on using SSL with the Twitter API, see our Guide to Connecting with SSL. If you’re continuing to have issues with the transition, you can join in on this discussion topic.

Update [Dec 10th, 2013]: the new certificates were deployed.

https://dev.twitter.com/blog/rest-api-ssl-certificate-updates
Connecting to Twitter API using SSL

The preferred method of connecting to the Twitter API is over HTTPS. While both HTTPS and Non-HTTPS connections to the Twitter API are supported, communicating over SSL preserves user privacy by protecting information between the user and the Twitter API as it travels across the public Internet.

Using OAuth isn't enough

While OAuth is mandated and protects the user from having their password captured in transit by substituting an OAuth token for the user's credentials, it's not enough to ensure complete privacy.

Cipher Selection

Twitter's servers and your client will negotiate a cipher spec upon connection. When possible, it's best to use the Twitter supplied cipher default (currently RC4) for session encryption. While other ciphers may offer better performance or security (and may be supported by both your client and Twitter's servers) the preferred cipher as negotiated by our servers is typically the best available for communication. We do not recommend overriding the negotiated selection in your code.

Verification

Validating and/or "pinning" the Twitter SSL Certificate in your code

Twitter's SSL Certificates for api.twitter.com are signed by Verisign. For Assets, such as those on s3.twimg.com (through s3.amazonaws), those certificates are signed by Verisign and/or Contando depending on the geographically closest CDN server.

Your application should ensure that the certificate chain returned for all Twitter servers is signed by one of our approved vendors (Verisign EV for twitter.com, Verisign for api.twitter.com, and Digicert for others) and not other CA roots.

https://dev.twitter.com/docs/security/using-ssl
Connecting to Twitter API using SSL

Verification

Validating and/or "pinning" the Twitter SSL Certificate in your code

Twitter’s SSL Certificates for api.twitter.com are signed by Verisign. For assets, such as those on s14.twimg.com (through s14.twimg.com), those certificates are signed by Verisign and/or Contendo depending on the geographically closest CDN server.

Your application should ensure that the certificate chain returned for all Twitter servers is signed by one of our approved vendors (Verisign EV for twitter.com, Versign for api.twitter.com, and DigiCert for others) and not other CA roots.

As of this writing, api.twitter.com’s certificate is:

```
1. Certificate:
   2. data:
   3.         Version: 3 (0x2)
   4.         Serial Number:
   6. Signature Algorithm: sha1WithRSAEncryption
   7. Issuer: C=US, O=Verisign, Inc., OU=Verisign Trust Network, OU=Terms of use at
      https://www.verisign.com/privacy (c)09, CR=Verisign Class 3 Secure Server CA - G2
   8. Validity
   9.   Not Before: May 2 00:00:00 2012 GMT
11.   Subject: C=US, ST=California, L=San Francisco, O=Twitter, Inc., OU=Twitter
      Security, CN=api.twitter.com
```


Your code should trust both the Verisign “G2” and “G3” root certificates. After May 3rd, 2013, Twitter’s certificates will be signed by the G3 root as Verisign is deprecating the G2 root. Prepare for this by trusting the correct root certificates.

Validate against the minimum number of root certificates

Don’t rely on the local operating system to validate the certificate if possible. This can be tampered with by malware, local IT staff, or other bad actors. Validate against the known vendors for api.twitter.com as listed above. Don’t include more certificates in your application’s trusted CA Root store from vendors that Twitter hasn’t listed.

https://dev.twitter.com/docs/security/using-ssl
Connecting to Twitter API using SSL

1. Certificate:
2. Data:
3. Version: 3 (0x2)
4. Serial Number:
6. Signature Algorithm: sha1WithRSAEncryption
7. Issuer: C=US, O=VeriSign, Inc., OU=VeriSign Trust Network, OU=Terms of use at https://www.verisign.com/rpa (c)09, CN=VeriSign Class 3 Secure Server CA - G2
8. Validity
9. Not Before: May 2 00:00:00 2012 GMT
11. Subject: C=US, ST=California, L=San Francisco, O=Twitter, Inc., OU=Twitter Security, CN=api.twitter.com

https://dev.twitter.com/docs/security/using-ssl
Connecting to Twitter API using SSL

Code Examples

When communicating with Twitter, it’s always best to use well-tested and predefined libraries to communicate with the API. If you have to write your own code to communicate with our servers, the following code examples in popular languages will help you to make proper SSL connections with full certificate validation.

Curl / LibCurl

CURL is frequently used for testing web applications and communication with the API. If you’re using libcurl in your applications, you will typically be required to supply your own CACert file, which is a list of accepted X509 root certificates for validation.

Twitter recommends that developers using TLS/SSL connections using libcurl verify that the CURLOPT_SSL_VERIFYPEER option is set to boolean value True (or integer value 1) and ensure that failure to return a successful verification results in a failed connection error rather than the establishment of an untrusted and unsecured connection. We also recommend that the SSL verifydepth be set to maximum (9) when possible.

From the command line, append the capath (or cafile) option to include a file containing the Verisign root CA certificate:

1. curl -3 --capath <file> --ssl https://api.twitter.com

PHP

PHP makes use of libcurl for SSL communication. The following will increase security on connections, assuming you have the Verisign root key stored in `ca-bundle.crt`:

1. curl_setopt($connection, CURLOPT_SSL_VERIFYPEER, True);
2. curl_setopt($connection, CURLOPT_SSL_VERIFYHOST, 2);
3. curl_setopt($connection, CURLOPT_CAINFO, 'path:/ca-bundle.crt');

Ruby

Place the Verisign (and all other) certs in `/etc/ssl/certs`, and use this code to ensure proper SSL verification on connect:

1. require 'net/http'
2. require 'net/https'
3. require 'uri'
4.
5. RootCA = '/etc/ssl/certs'
6. url = URI.parse 'https://api.twitter.com/yourrequestgoeshere'

https://dev.twitter.com/docs/security/using-ssl
Connecting to Twitter API using SSL

From the command line, append the capath (or cafie) option to include a file containing the Verisign root CA certificate:

```
1. curl -3 -capath <file> --ssl https://api.twitter.com
```

### PHP

PHP makes use of libcurl for SSL communication. The following will increase security on connections, assuming you have the Verisign root key stored in `ca-bundle.crt`:

```
1. curl_setopt($connection, CURLOPT_SSL_VERIFYHOST, 2);
2. curl_setopt($connection, CURLOPT_SSL_VERIFYPEER, true);
3. curl_setopt($connection, CURLOPT_CAINFO, "path:/ca-bundle.crt");
```

### Ruby

Place the Verisign (and all other) certs in /etc/ssl/certs, and use this code to ensure proper SSL verification on connect.

```
1. require 'net/http'
2. require 'net/https'
3. require 'uri'
4.
5. RootCA = '/etc/ssl/certs'
6. url = URI.parse 'https://api.twitter.com/yourrequestgoeshere'
7. http = Net::HTTP.new(url.host, url.port)
8. http.ca_path = RootCA
9. http.verify_mode = OpenSSL::SSL::VERIFY_PEER
10. http.verify_depth = 9
11. request = Net::HTTP::Get.new(url.path)
12. # handle oauth here, or whatever you need to do...
13. response = http.request(request)
14.
15. # ... process response ...
```

### Python

See this Stack Overflow post regarding SSL verification under Python. Depending on the Python version and SSL Library you are using, verification may be supported natively in the library, or you may have to extract the certificate from the connection and perform additional verification steps.

[https://dev.twitter.com/docs/security/using-ssl](https://dev.twitter.com/docs/security/using-ssl)
Summary

- Twitter Developers
  - Twitter Platform Objects
- Twitter for Websites
- Twitter Search API
- Twitter REST API
- Twitter Streaming API
References

• Twitter Developers, https://dev.twitter.com/