API Blender

A Uniform Interface to Social Platform APIs

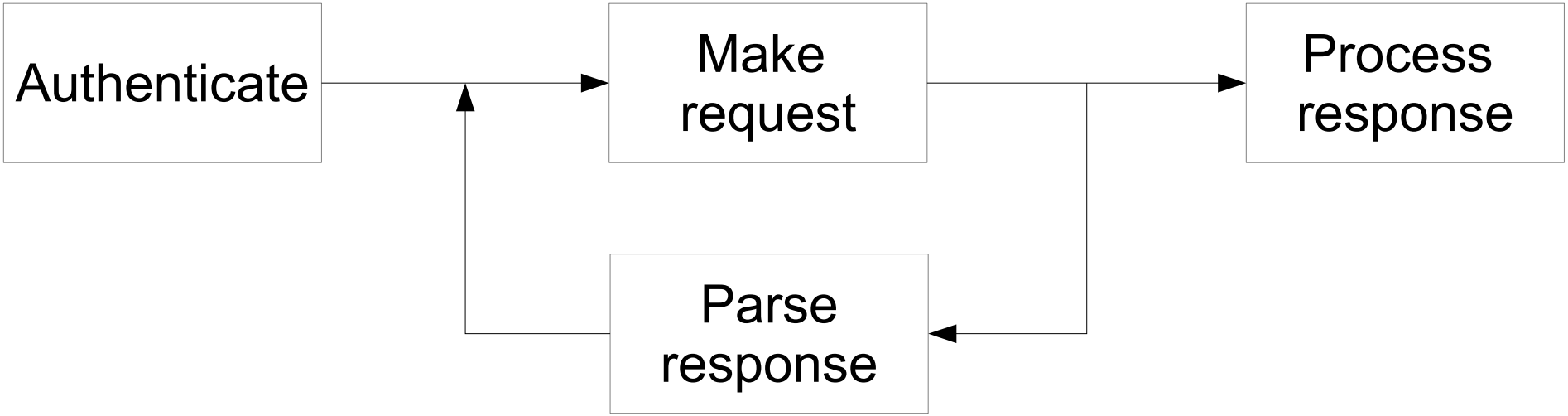
{georges.gouriten, pierre.senellart}
@telecom-paristech.fr

Outline

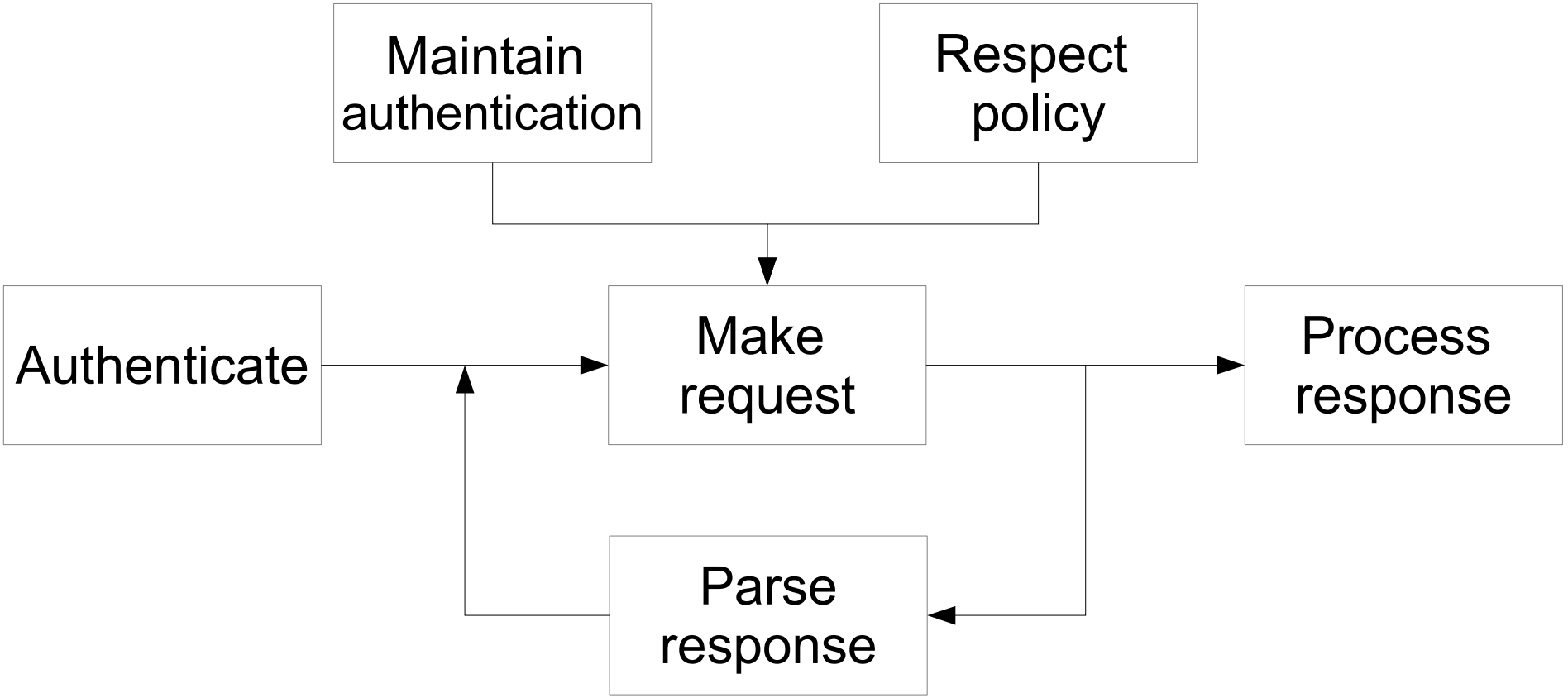
Interacting with social platforms ...

is usually painful, ...

but API Blender makes it easy!



Generic interactions with a Web API



Generic interactions with a Web API


facebook


g+


flickr


YouTube

Graph API - Facebook Dev

developers.facebook.com/docs/reference/api/

facebook DEVELOPERS

Documentation Support Blog Apps Gerard Sophistian

Getting Started
Core Concepts
Social Design
Social Plugins
Open Graph
Social Channels
Authentication
Graph API
Advanced Topics
SDK Reference
Tools

Concepts
Batch Requests
Permissions
Real-time Updates

Objects
Achievement(Instance)
Album

Graph API

Core Concepts › Graph API

At Facebook's core is the social graph; people and the connections they have to everything they care about. The Graph API presents a simple, consistent view of the Facebook social graph, uniformly representing objects in the graph (e.g., people, photos, events, and pages) and the connections between them (e.g., friend relationships, shared content, and photo tags).

Every object in the social graph has a unique ID. You can access the properties of an object by requesting https://graph.facebook.com/ID. For example, the official page for the Facebook Platform has id 19292868552, so you can fetch the object at https://graph.facebook.com/19292868552:

```
{
   "name": "Facebook Platform",
   "website": "http://developers.facebook.com",
   "username": "platform",
   "founded": "May 2007",
   "company_overview": "Facebook Platform enables anyone to build...",
   "mission": "To make the web more open and social.",
   "products": "Facebook Application Programming Interface (API)...",
   "likes": 449921,
   "id": 19292868552,
   "category": "Technology"
}
```

Alternatively, people and pages with usernames can be accessed using their username as an ID. Since "platform" is the username for the page above, https://graph.facebook.com/platform will return what you expect. All responses are JSON objects.

Documentation | Twitter

https://dev.twitter.com/docs

twitter developers

API Health Blog Discussions Documentation Sign in

Home

Documentation

Getting Started with the Twitter Platform is easy. Jump right into the API resource documentation, explore the developer console, or create an app.

Getting Started

- Getting Started with the Twitter Platform
- Twitter API FAQ
- JSON Parsing & Tweet IDs
- The t.co URL Wrapper
- Connecting to the API using SSL
- Developer Console

Twitter for Websites

Easily embed Twitter functionality, encourage your users to share content on Twitter, and increase your audience reach with Twitter for Websites.

- Tweet Button
- Follow Button

The REST API

The REST API provides simple interfaces for most Twitter functionality.

- Comprehensive Resource Documentation
- Using the Twitter Search API and GET search
- Tweet Entities
- Things to know about the REST API
- History of the REST and Search APIs
- Rate Limiting & Rate Limiting FAQ
- Error Codes & Responses
- Explore the API with Tools & Consoles
- Finding Tweets about Places
- Working with Timelines

Google+ API - Google+ Pla

https://developers.google.com/+/api/

Google Developers

Search Google Developers

Georges.Gouriten@gmail.com Sign out

Google+ Platform +1 6.7k

Overview
Plugins
Hangouts API
REST API
Overview
People
Activities
Comments
OAuth
Downloads
Release Notes
Discussions
Blog
Best Practices
Office Hours

Google+ API

The Google+ API is the programming interface to Google+. You maximum engagement using Google+ features from within your

Note: The Google+ API currently provides read-only access t

Quota

Applications are limited to a courtesy usage quota. This should application. To see the courtesy limit and to request a higher lim

Authorization

Many API calls require that the user of your application grant pe data. To learn more, see OAuth.

API Calls

Most of the Google+ API follows a RESTful API design, meanin user, you might send an HTTP request like:

API v2.0 – Audience - YouT

https://developers.google.com/youtube/2.0/developers_guide_protocol_audience

Google Developers

Search Google Developers

Georges.Gouriten@gmail.com Sign out

YouTube +1 731

rform functions normally executed on the YouTube website. The API enables your application to search for YouTube videos and video responses. In addition, the API lets your application upload videos to YouTube or update existing videos. Your ons, user profiles and more. Finally, your application can submit authenticated requests to enable users to create playlists, c entities.

who are writing client applications that interact with YouTube. It provides examples of basic API operations using raw HTTP ers may prefer to read the language-specific developer guides that explain how to use the client libraries for those languages

d the general principles behind the Google Data APIs protocol. Google Data APIs provide a simple, standard protocol for a APIs are based on the Atom 1.0 and RSS 2.0 syndication formats as well as the Atom Publishing Protocol.

sections:

different authentication methods available for associating API operations with a specific user account. This section also on for the YouTube Data API and other Google Data APIs. Throughout this documentation, explanations of specific API ction requires user authentication.

section provides a sample API response and explains how to extract information about a single video from a list of videos lustrates how the XML elements in a YouTube Data API response correspond to the video information typically displayed lains the API request format for updating an individual video entry.

Flickr Services

www.flickr.com/services/api/

flickr from YAHOO!

Signed in as netiru (1 new) Help Sign Out

Home You Organize & Create Contacts Groups Explore Upload Search

The App Garden

Create an App | API Documentation | Feeds | What is the App Garden?

The Flickr API is available for non-commercial use by outside developers. Commercial use is possible by prior arrangement.

Read these first:

- Developer Guide
- Overview
- Encoding
- User Authentication

- Dates
- Tags
- URLs
- Buddyicons

- Flickr APIs Terms of Use

- API Keys
- Developers' mailing list

Photo Upload API

API Methods

activity

- flickr.activity.userComments
- flickr.activity.userPhotos

auth

- flickr.auth.checkToken
- flickr.auth.getFrob
- flickr.auth.getFullToken
- flickr.auth.getToken

auth.oauth

- flickr.auth.oauth.checkToken
- flickr.auth.oauth.getAccessToken

blogs

- flickr.blogs.getList

flickr.photos.search

Return a list of photos matching some criteria. Only photos visible to the calling user will be returned. To return private or semi-private photos, the caller must be authenticated with 'read' permissions, and have permission to view the photos. Unauthenticated calls will only return public photos.

Authentication

This method does not require authentication.

Arguments

api_key (Required)
Your API application key. See here for more details.

user_id (Optional)
The NSID of the user who's photo to search. If this parameter isn't passed then everybody's public photos will be searched. A value of "me" will search against the calling user's photos for authenticated calls.

tags (Optional)
A comma-delimited list of tags. Photos with one or more of the tags listed will be returned. You can exclude results that match a term by prepending it with a - character.

tag_mode (Optional)
Either 'any' for an OR combination of tags, or 'all' for an AND combination. Defaults to 'any' if not specified.

text (Optional)
A free text search. Photos who's title, description or tags contain the text will be returned. You can exclude results that match a term by prepending it with a - character.

min_upload_date (Optional)
Minimum upload date. Photos with an upload date greater than or equal to this value will be returned. The date can be in the form of a unix timestamp or mysql datetime.

max_upload_date (Optional)
Maximum upload date. Photos with an upload date less than or equal to this value will be returned. The date can be in the form of a unix timestamp or mysql datetime.

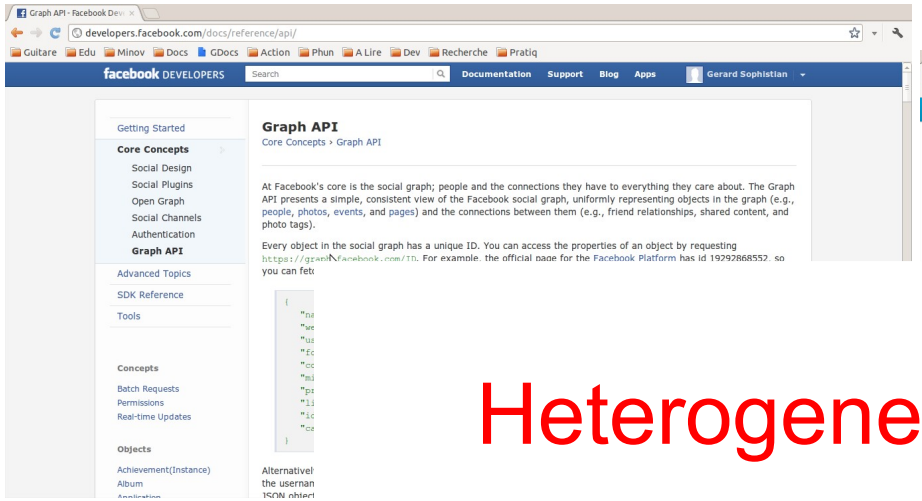
Graph API

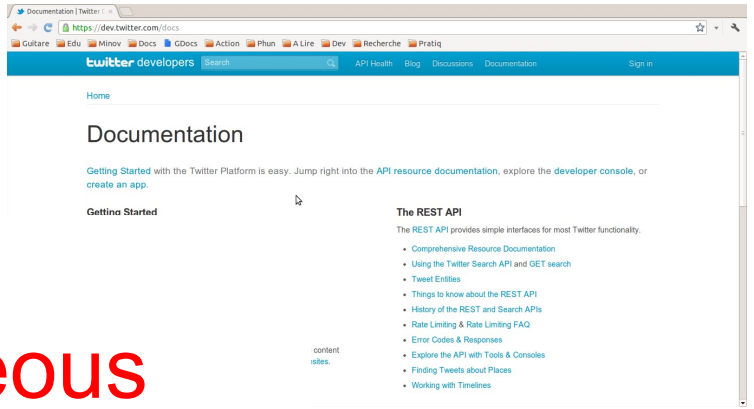
Core Concepts › Graph API

At Facebook's core is the social graph; people and the connections they have to everything they care about. The Graph API presents a simple, consistent view of the Facebook social graph, uniformly representing objects in the graph (e.g., people, photos, events, and pages) and the connections between them (e.g., friend relationships, shared content, and photo tags).

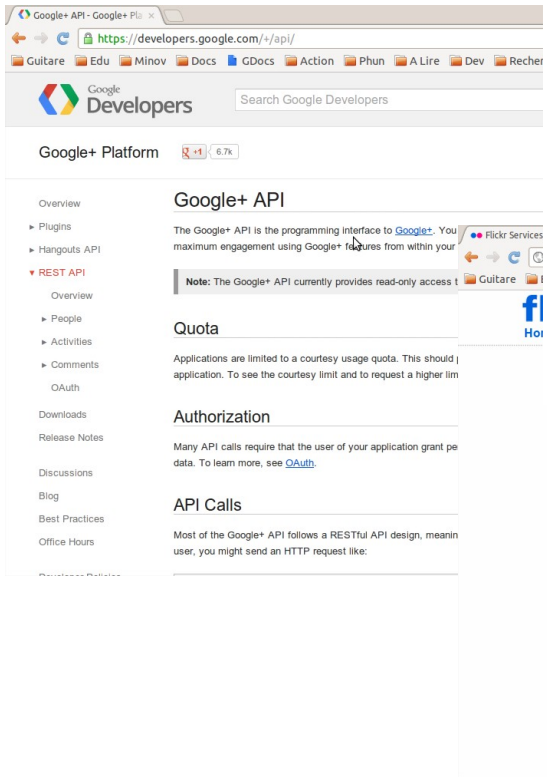
Every object in the social graph has a unique ID. You can access the properties of an object by requesting `https://graph.facebook.com/ID`. For example, the official page for the Facebook Platform has id 19292868552, so you can fetch the object at https://graph.facebook.com/19292868552:

```
{
   "name": "Facebook Platform",
   "website": "http://developers.facebook.com",
   "username": "platform",
   "founded": "May 2007",
   "company_overview": "Facebook Platform enables anyone to build...",
   "mission": "To make the web more open and social.",
   "products": "Facebook Application Programming Interface (API)...",
   "likes": 449921,
   "id": 19292868552,
   "category": "Technology"
}
```

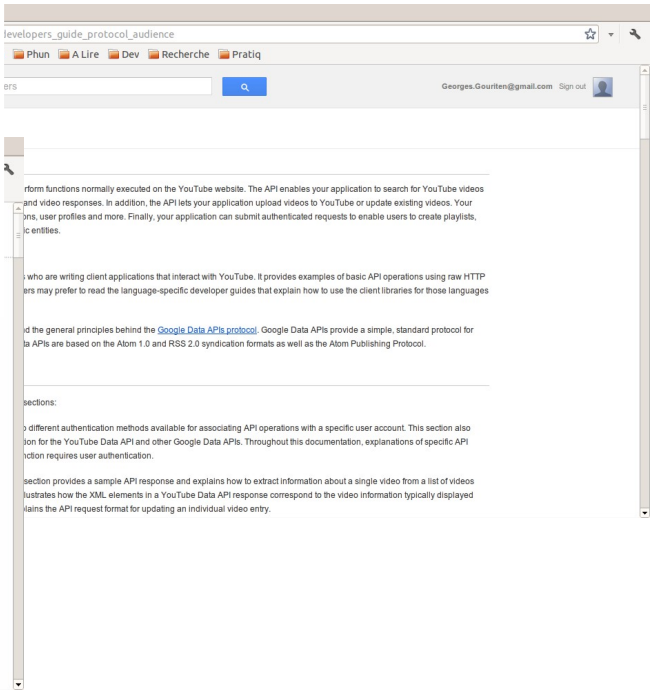




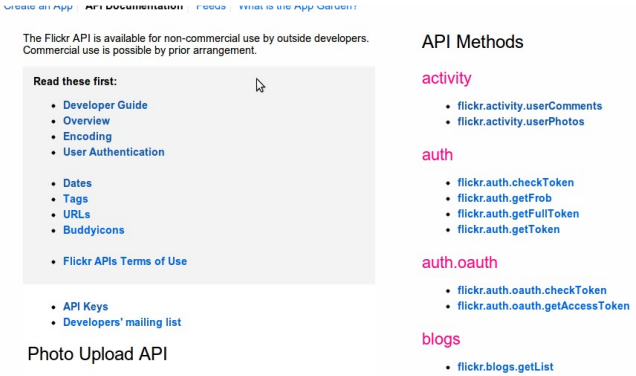
Heterogeneous

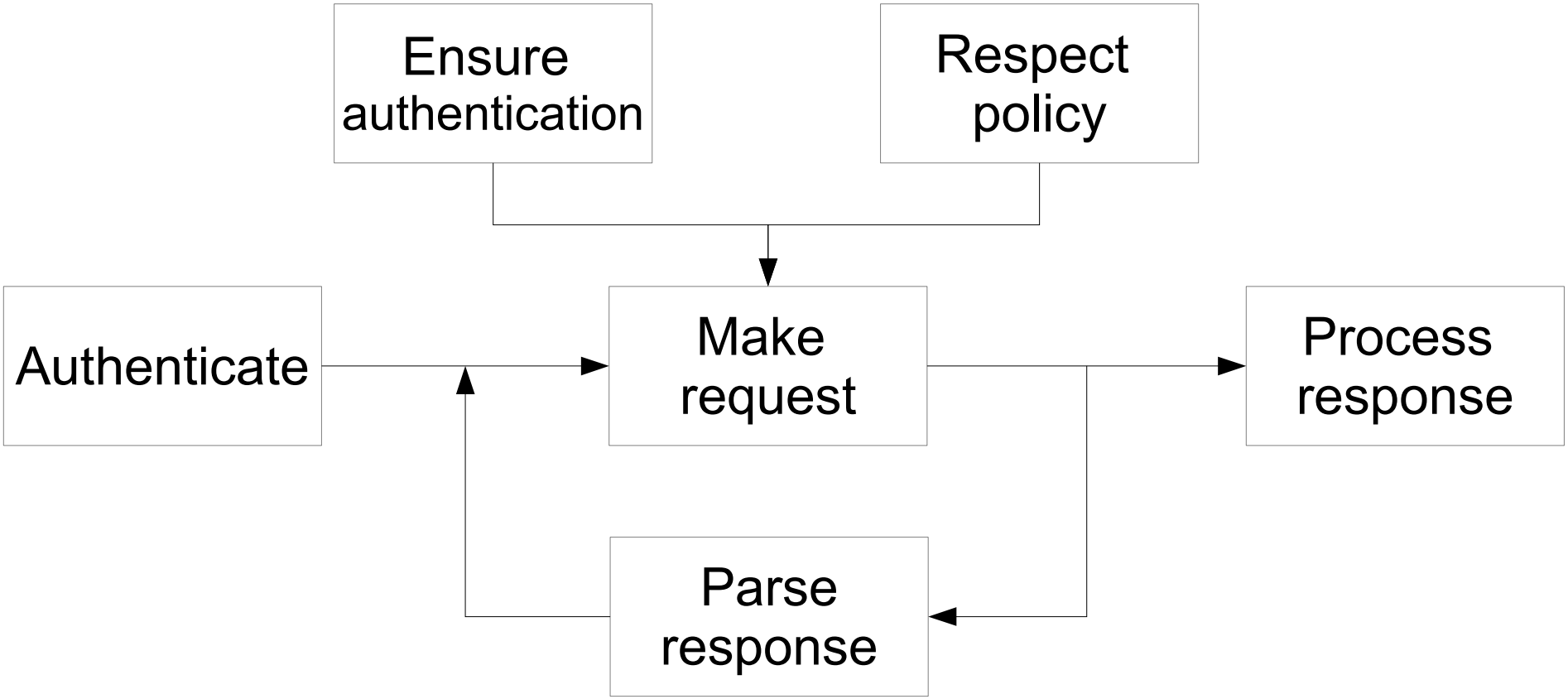


Sometimes obscure



Error-prone





Generic interaction with a Web service

Home-made development

What most people do

Redundant efforts

What about collective efforts?

Top-down

WSDL
WADL

Error 404

Bottom-up

Spring Social
SPORE-DL

Promising direction

API Blender

Social Web APIs description files

Python implementation

API Blender

Social Web APIs description files

Python implementation

Lightweight social Web API description format

```
{
    “name”: "twitter-search",
    “host”: "search.twitter.com",
    “port”: 80,
    “authentication”: authentication_object,
    “policy”: policy_object,
    “interactions”: [interaction_object]
}
```

Simple Authentication

```
{
    "type": "simple",
    "path": "/oauth/access_token",
    "parameters": {
        "client_id": "x2836",
        "client_secret": "z45725",
        "grant_type": "client_credentials"
    }
}
```

Oauth Authentication


```
{
    “type”: “oauth”,
    “consumer_key”: “x4565482s”,
    “consumer_secret”: “tsouintsouin”,
    “request_token_url”: “https://api.twitt...”,
    “access_token_url”: “https://api.twitt...”,
    “authorize_url”: “https://api.twitt...”
}
```

Policy

```
{
    "active": true,
    "requests_per_hour": 400,
    "too_many_calls_response_code": 420,
    "too_many_calls_waiting_seconds": 600
}
```

Interaction

```
{   "name": "search",
    "description": "search for tweets",
    "request": {
        "url_root_path": "/search.json",
        "method": "GET",
        "url_params": [
            [ "q", "string", false, null ],
            [ "rpp", "int", true, 100 ],
        ]
    },
    "response": {
        "expected_status_code": 200,
        "serialization_format": "JSON",
        "extractor": extractor_object   } }
```

Extractor

```
{
    “data.field1”: “mydata.there.fieldx”,
    “data.field2”: “mydata.there.fieldy”
}
```

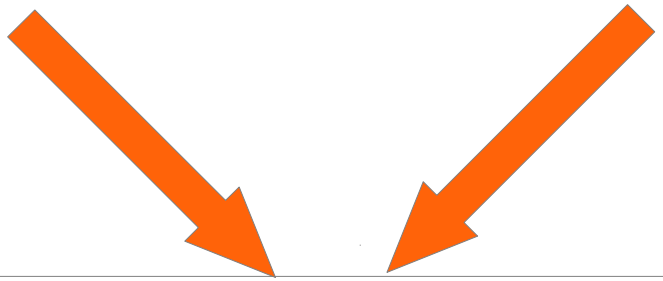
```
"data": {
    "field1": <field1_value>,
    "field2": <field2_value>
}
```

```
{   "data.field1": "mydata.there.fieldx",
    "data.field2": "mydata.there.fieldy"   }
```

```
"mydata": {
    "there": {
        "fieldx": <field1_value>
        "fieldy": <field2_value>
    }
}
```

```
“results”: {
    “from_user”: ...
}
```

```
“data”: {
    “from”: {
            “name”:  ...
    }
}
```



```
“posts”: {
    “user_name”: ...
}
```

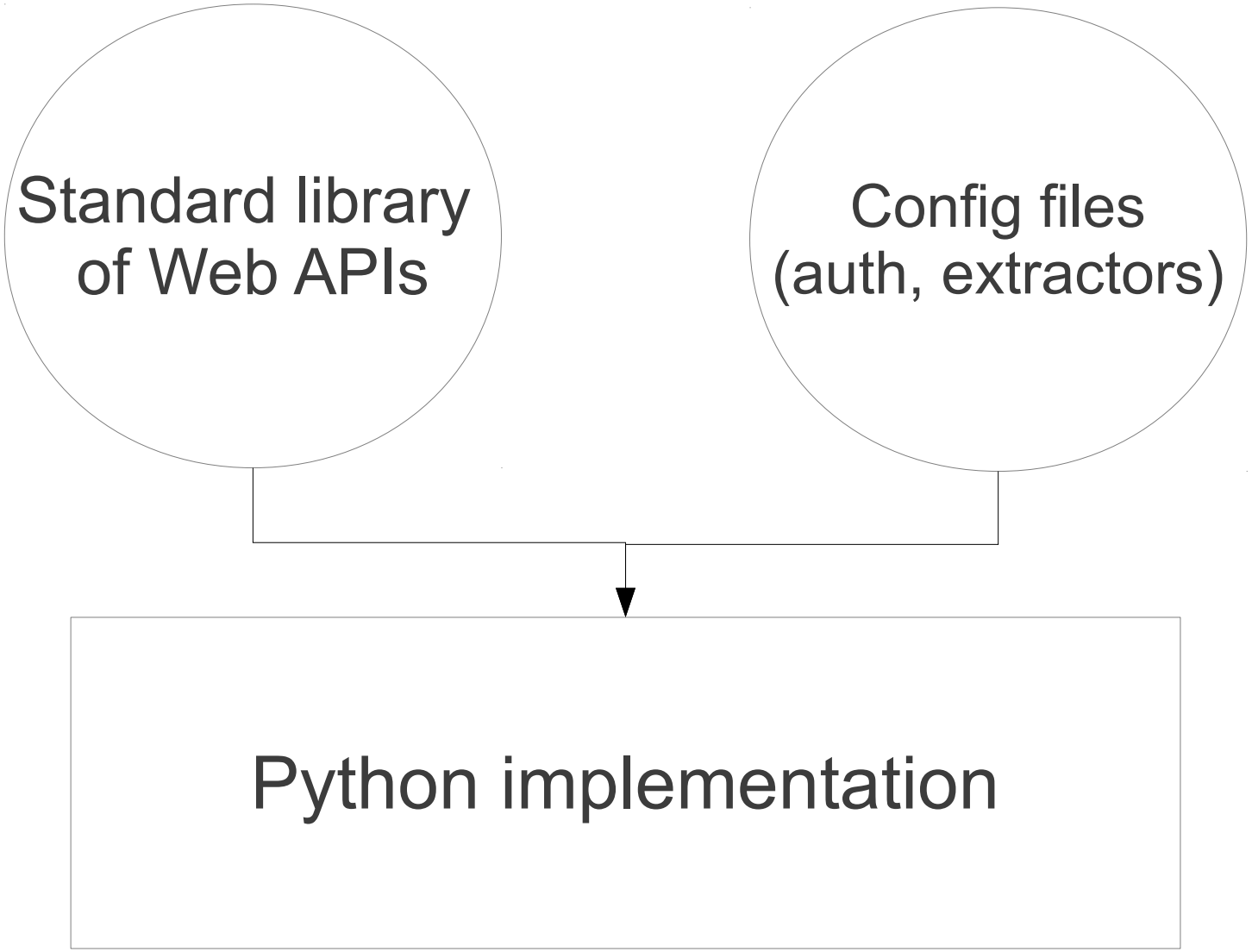
Lightweight social Web API description format

```
{
    “name”: "twitter-search",
    “host”: "search.twitter.com",
    “port”: 80,
    “authentication”: authentication_object,
    “policy”: policy_object,
    “interactions”: [interaction_object]
}
```

API Blender

Social Web APIs description files

Python implementation



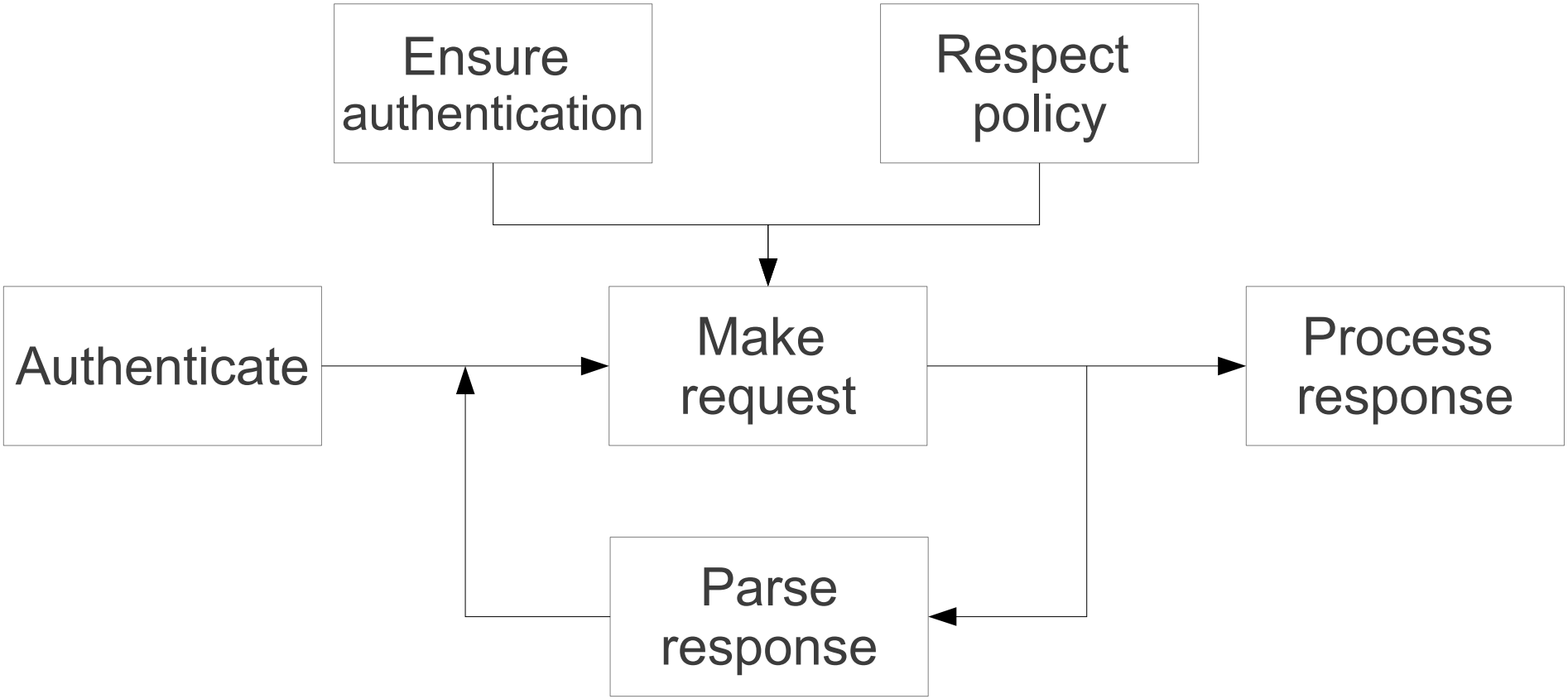
github.com/netiru/apiblender

How the apiblender works (1)

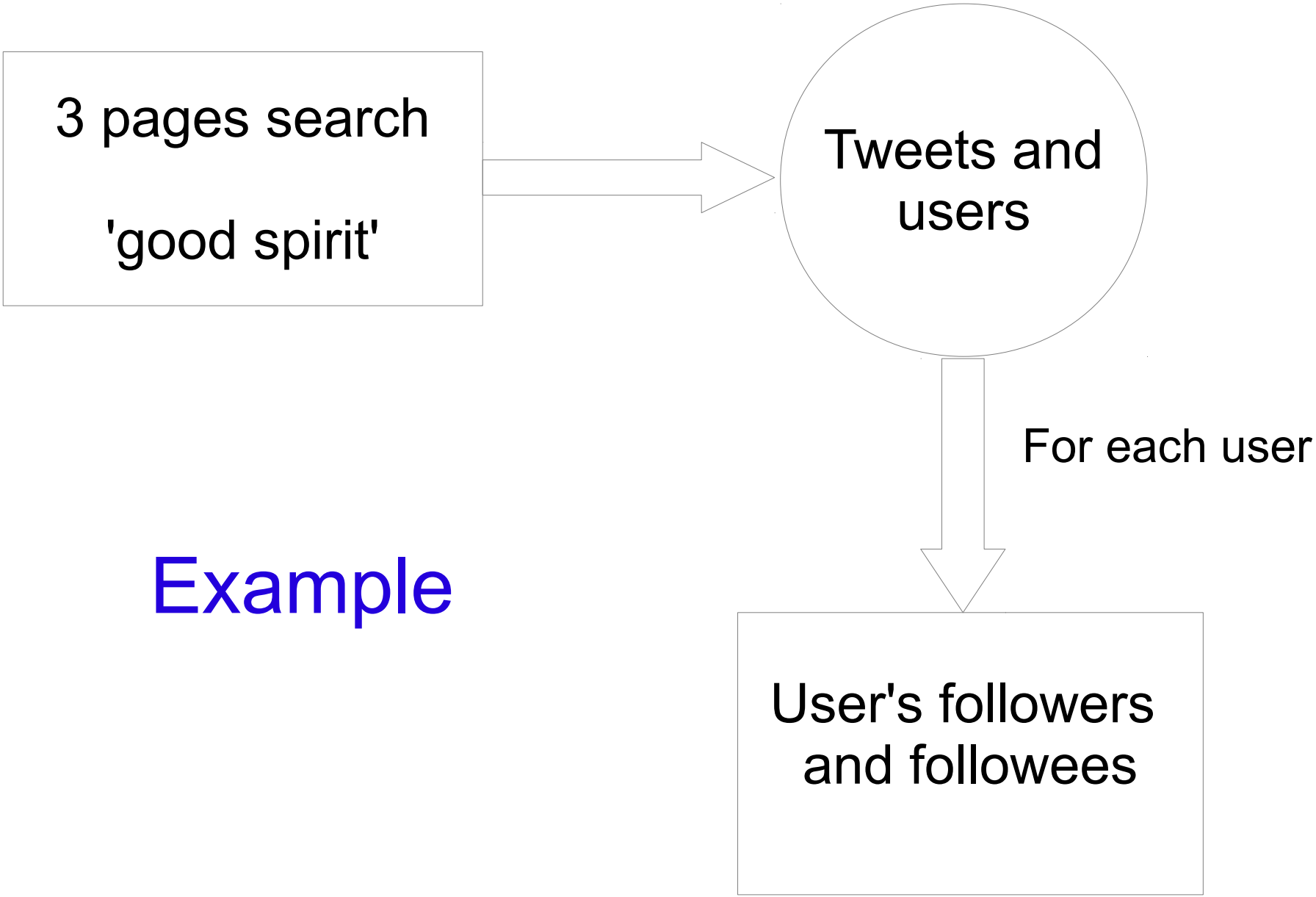
```
import apiblender

blender = apiblender.Blender()

blender.list_servers()
        .load_server('flickr')

        .list_interactions()
        .load_interaction('photos_search')
```

How the apiblender works (2)

```
blender.list_url_params()
blender.set_url_params({'tags': 'good spirit'})

result = blender.blend()

result['raw_content']
result['prepared_content']
result['headers']
```



Generic interaction with a Web service


3 pages search
'good spirit'
Tweets and users
For each user
User's followers and followees

Example

Interactions made easy (1)

```
blender.load_server("twitter-search")
blender.load_interaction("search")

users = set()
for p in range(1,3):
  blender.set_url_params({"q": 'good spirit',
                          "page": p  })
  response=blender.blend()
  results=response['raw_content']['results']
  for twitt in results:
    users.add(twitt['from_user'])
```

Interactions made easy (2)

```
blender.load_server("twitter-generic")

for user in users:
    blender.load_interaction('followers')
    blender.set_url_params({"screen_name":
                            user})
    followers = blender.blend()
```

demo

Summary

Policy management
Persistent authentication
Data integration

User-friendly interactions
chaining

Coming soon

> 5 APIs (and more methods)

Request chaining libraries

It's open source!

github.com/netiru/apiblender/