REST

Restful API

Une API web est dite “**Restful**” si elle a les caractéristiques suivantes:

- Les requêtes sont envoyées sous forme de requêtes HTTP:
  - Méthodes HTTP: GET, POST, PUT, PATCH, DELETE..
- Un **Endpoint**: http://example.com/api/ressources
- Les requêtes doivent être envoyées avec un MIME*/Content-type spécifique tel: JSON, XML, HTML, etc.

* Multipurpose Internet Mail Extensions

Contraintes d’une API REST

Une API Restful doit respecter certaines contraintes d’architecture:

1. **Archi client-serveur**: chacun évolue indépendamment.
2. **Serveur sans états (stateless)**: pas de sessions.
3. **Utilisation du cache**: le client sait combien de temps il peut garder les données qu’il reçoit avant expiration.
4. **Une interface uniforme**: un seul id, contient l’URL des ressources suivantes..
5. **Archi en couches**: Réduire la complexité de l’architecture globale de l’API.
6. **Code à la demande**: étendre les fonctionnalités du client.

Ressources

En REST tout est resources, on interagit avec les ressources à travers différents endpoints (URLs):

- http://example.com/products
- http://example.com/products/1
- http://example.com/products/1/description

Action sur les ressources

Pour interagir avec une ressource, il suffit d'utiliser la méthode HTTP (verbe) adéquate :

- GET http://example.com/products/1 (Récupérer)
- POST http://example.com/products (Créer)
- PUT http://example.com/products/1 (Mettre à jour)
- DELETE http://example.com/products/1 (Effacer)

API Endpoints

Tweets	Retweets	Likes (formerly favorites)
• POST statuses/update	• POST statuses/retweet/:id	• POST favorites/create/:id
• POST statuses/destroy/:id	• POST statuses/unretweet/:id	• POST favorites/destroy/:id
• GET statuses/show/:id	• GET statuses/retweets/:id	• GET favorites/list
• GET statuses/oembed	• GET statuses/retweets_of_me	
• GET statuses/lookup	• GET statuses/retweeters/ids	

API de Twitter

https://developer.twitter.com/en/docs/tweets/post-and-engage/overview

API Endpoints

Resource URL

https://api.twitter.com/1.1/statuses/update.json

Resource Information

Response formats	JSON
Requires authentication?	Yes (user context only)
Rate limited?	Yes

Parameters

Name	Required	Description	Default Value	Example
status	required	The text of the status update. URL encode as necessary. t.co link wrapping will affect character counts.		

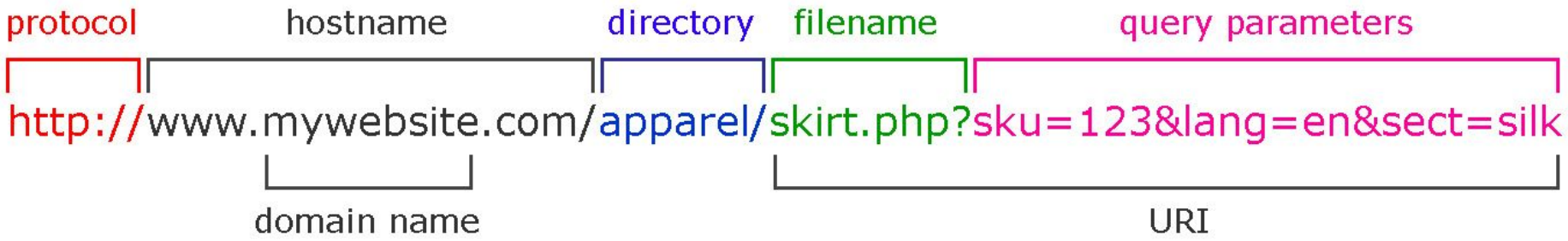
POST statuses/update

Paramètres des Endpoints

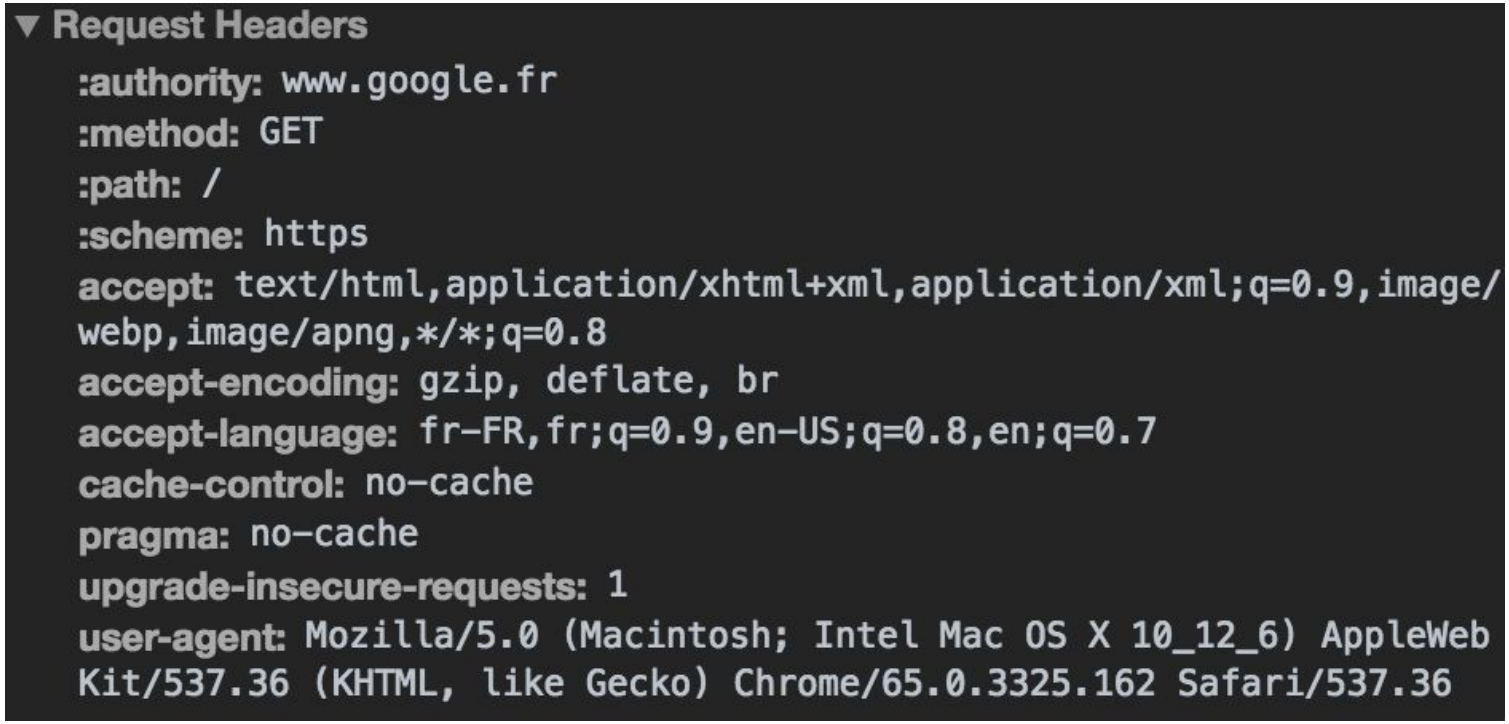
Un endpoint peut accepter un ensemble de paramètres pour effectuer sa tâche. Ces paramètres peuvent être passés de **trois manières** différentes:

- Paramètres dans l'URL.
- Headers dans la requête.
- Corps (body) de la requête.

Paramètres des Endpoints - paramètres dans l'URL



Paramètres des Endpoints - headers de la requête



Paramètres des Endpoints - corps de la requête

```
1    POST /login HTTP/1.1
2    Host: foo.com
3    Content-Type: application/x-www-form-urlencoded
4    Content-Length: 37
5
6    username=foo@foo.com&password=123_foo
```

Codes d’état HTTP (Status code)

Il existe plus de 40 codes HTTP, chacun ayant une siginification bien précise. Voici les grandes catégories:

- **1xx**: Information (e.g. 100 attente de la suite (continue))
- **2xx**: Succès (e.g. 200 tout s’est bien passé (OK))
- **3xx**: Redirection (e.g. 301 document déplacé (redirection))
- **4xx**: Erreur du client web (e.g. 404 document introuvable)
- **5xx**: Erreur du serveur (e.g. 500 Erreur interne)

API Restful et cache (E-tags)

Les E-tags permettent d'implémenter la contrainte de cache pour une API Restful. Il existe également d'autres méthodes qui sont l'utilisation des headers **Expires** et **cache-control**.

```
HTTP/1.1 200 OK
Date: Sat, 09 Feb 2013 16:09:50 GMT
Server: Apache/2.2.22 (Ubuntu)
Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT
ETag: "c0947-b1-4d0258df1f625"
Content-Type: application/json

{
    id: 4,
    item: "take out the trash",
    created: "Sat, 02 Feb 2013 08:29:53 GMT",
    updated: "Sat, 02 Feb 2013 12:02:47 GMT",
}
```

API Restful et cache (E-tags)

Une fois que je connais l'E-tag, je peux à l'avenir demander s'il a changé ou non en envoyant une requête avec comme header:

```
If-None-Match: c0947-b1-4d0258df1f625
```

```
HTTP/1.1 304 Not Modified
Date: Sat, 09 Feb 2013 16:09:50 GMT
Server: Apache/2.2.22 (Ubuntu)
Last-Modified: Sat, 02 Feb 2013 12:02:47 GMT
ETag: "c0947-b1-4d0258df1f625"
```

API Restful et cache (E-tags)

```
HTTP/1.1 200 OK
Date: Sat, 09 Feb 2013 16:29:24 GMT
Server: Apache/2.2.22 (Ubuntu)
Last-Modified: Sat, 02 Feb 2013 14:33:21 GMT
ETag: "c7493-d7-a6b64d37f6cc3"  # New ETag!
Content-Type: application/json

{
    id: 4,
    item: "Take out the trash, TODAY!",
    created: "Sat, 02 Feb 2013 08:29:53 GMT",
    updated: "Sat, 02 Feb 2013 14:33:21 GMT",
}
```

OpenAPI

C’est un ensemble de spécifications permettant de décrire une API REST.

Cette description peut être comprise aisément à la fois par les développeurs ainsi que par les ordinateurs.

L’API est décrite sous forme d’un fichier JSON ou YAML.

Un ensemble d’outils permet de facilement convertir ces fichiers de description en documentation web ou encore en bibliothèques pour directement utiliser l’API décrite sans avoir à l’implémenter.

OpenAPI

```yaml
swagger: '2.0'
schemes:
  - https
host: api.twitter.com
basePath: /1.1
info:
  contact:
    email: support@twitter.com
    name: Twitter support
    url: 'https://dev.twitter.com'
    x-twitter: twitter
  title: Twitter
  version: '1.1'
```

OpenAPI

```yaml
paths:
  /account/settings.json:
    get:
      description: |-
        Returns settings (including
        current trend, geo and sleep time information) for the authenticating user.
      externalDocs:
        url: 'https://dev.twitter.com/docs/api/1.1/get/account/settings'
      operationId: account.settings.get
      responses:
        '200':
          description: Successful Response
    parameters:
      - description: |-
          The Yahoo! Where On Earth ID to use as the user's default trend location. Global informati
          available by using 1 as the WOEID. The woeid must be one of the locations returned by GET
          trends/available.

          Example Values: 1
        in: query
        name: trend_location_woeid
        required: false
        type: string
      - description: |-
          When set to true, t or 1, will enable sleep time for the user. Sleep time is the time when
          SMS notifications should not be sent to the user.

          Example Values: true
        in: query
        name: sleep_time_enabled
        required: false
        type: string
```