



GrandPy'Bot

Une adresse ... une histoire

nClassrooms?

GrandPy'Bot



*Dis moi où tu désires aller ... Je te donnerai
l'adresse et te raconterai une histoire ...*

Gabrielle Azadian



SOMMAIRE

TRELLO	3
TEMPLATE index.html	4
SCHEMA	5
PARSER -> REGEX	6
GOOGLE API	7
TEST MOCK Google_api	8
MEDIAWIKI	9
TEST MOCK media_api	10
STYLE css	11
GRANDPY'BOT demande correcte	12
DEMANDE INCORRECTE demande mal formulée	13
article wikipédia non trouvé	14
adresse inconnue	14
inspecteur	15

Trello

https://trello.com/c/0wu4ajdH/2-ecrire-un-script-flask-accueil-search

[P7] GrandPy Bot

Tableaux Passer à ...

Users Stories

En tant qu'utilisateur, je souhaite taper un lieu afin d'obtenir une adresse.

En tant qu'utilisateur, je souhaite taper une adresse ou un lieu dans le formulaire afin d'obtenir une carte.

En tant qu'utilisateur, je souhaite taper un lieu ou adresse dans le formulaire afin d'obtenir l'historique du lieu donné.

+ Ajouter une autre carte

Ecrire un script Flask accueil + search

Dans la liste Terminé

A faire

+ Ajoutez une autre liste

Image de couverture

Description

Ajouter une description plus détaillée...

AJOUTER À LA CARTE

Membres

Étiquettes

Checklist

Dates

Pièce jointe

Initialiser Flask

Masquer les tâches cochées Supprimer

Création du template index.html

Création du script Flask

Parser

Ajouter un élément

POWER-UPS

+ Ajouter des Pow...

Passez à Business Class pour obtenir un nombre illimité de Power-ups par tableau.

Promouvoir l'offre de l'espace de travail

Activité

Masquer les détails

Écrivez un commentaire...

BUTLER

+ Ajouter un bouton

ACTIONS

→ Déplacer

Copier

Créer un modèle

Suivre

Archiver

Partager

Gabrielle Azadian a terminé Parser sur cette carte il y a 9 minutes

Gabrielle Azadian a terminé Création du script Flask sur cette carte il y a 9 minutes

Gabrielle Azadian a terminé Création du template index.html sur cette carte il y a 9 minutes

Gabrielle Azadian a ajouté Initialiser Flask à cette carte il y a 11 minutes

Gabrielle Azadian a déplacé cette carte de A faire à Terminé 15 mai à 15:03

Gabrielle Azadian a ajouté cette carte à A faire 1 avr. à 15:41

GrandPy'Bot ← 3

Template index.html

-> Création du script **index.html**

-> Initialisation npm (Bootstrap)
Fichiers --> package-lock.json
--> package.json
Dossier --> CSS, SCSS

-> Script **js**

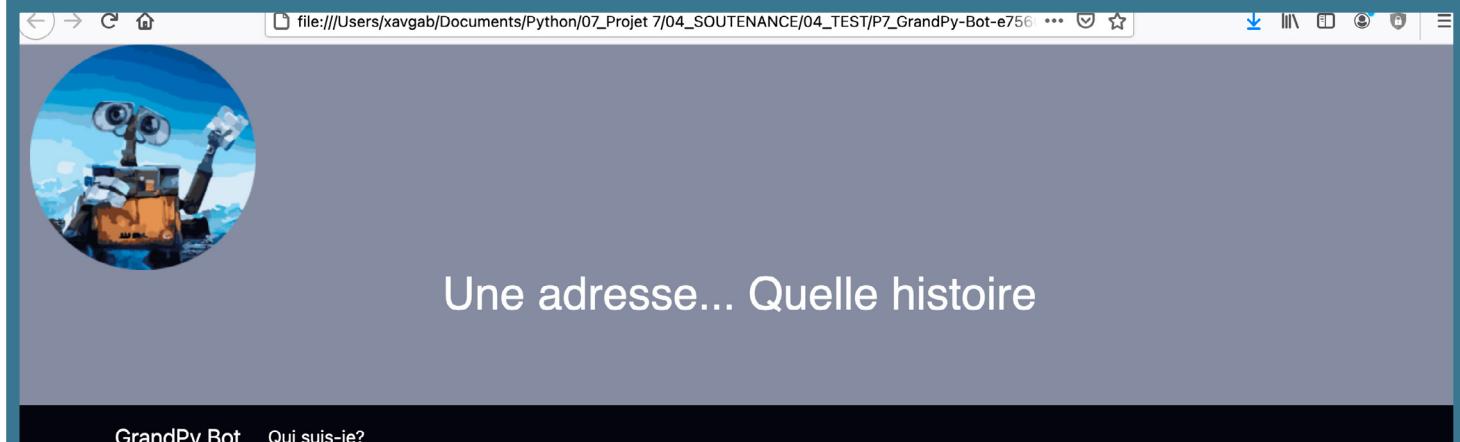
```
let adresse = document.getElementById('bouton');
adresse.addEventListener('click', getValue);
const regex = /[A-Z0-9][w]*/gm;
var request = new XMLHttpRequest();
let map;

function getValue() {
    // Sélectionner l'élément input et récupérer sa valeur
    var input = document.getElementById('adresse').value;
    const found = input.match(regex);
    if (found) {
        // Afficher la valeur
        alert(found);
    }
    else {
        alert('Invalide...Pas de lettres Majuscules')
    }
}
```

-> Création du script **views.py**

-> Création du premier test
--> **test_parser**

GrandPy'Bot



```
<div class=>col-1<>style=>background-color: #c1c7dc; min-height:30rem></div>
<div class=>col <>style = <background-color: #c1c7dc; min-height:30rem></div>
<form action=>{{ url_for('index') }}>, method=>POST></form>
```

→ Mauvaise utilisation des styles

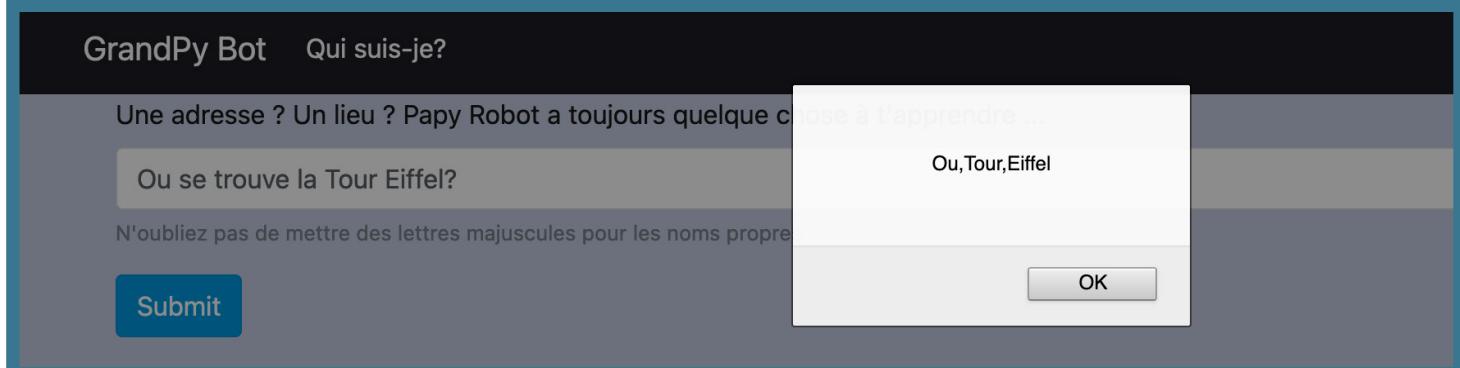
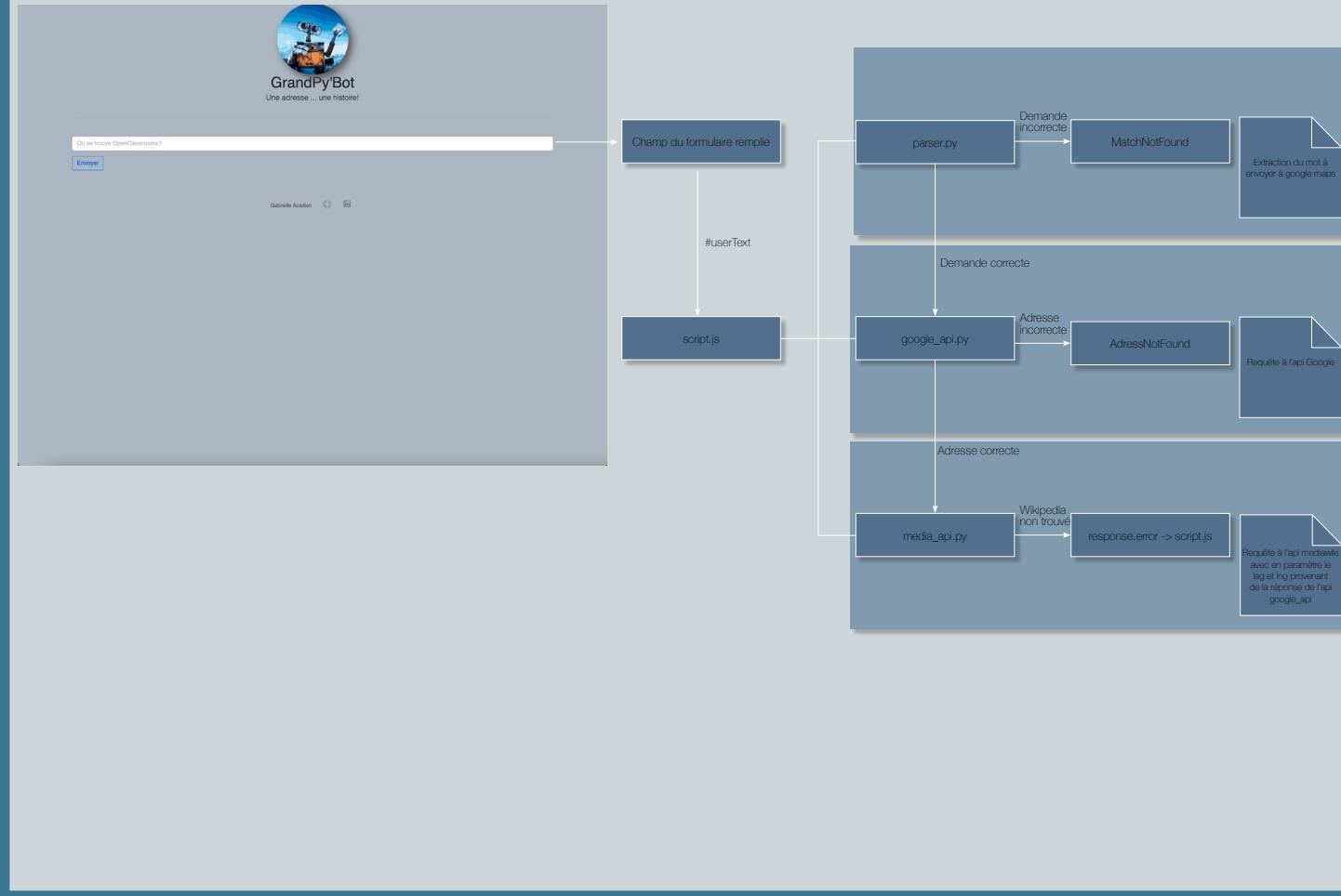


Schéma de l'application

Schéma de l'application



Parser -> Regex

- > Utilisation du site [regex101](https://regex101.com)
- > Création du fichier `constant.py`

```
PATTERN_PARSER = [
    r'.*se trouve [e]|a]\s?(.*?\?),
    r'.*se trouvent [es]\s?(.*?\?)]
```

-> Script js

```
let form = document.querySelector('#user-text-form');

function postFormData(url, data){
    return fetch(url, {
        method: 'POST',
        body: data,
    })
    .then(response => response.json())
    .catch(error => console.log(error));
}

form.addEventListener('submit', function(event){
    event.preventDefault();
    $('#btn-one').html('<span class="spinner-border spinner-border-sm mr-2" role="status" aria-hidden="true"></span>Loading...').attr('disabled', true);
    setTimeout(function(){
        document.getElementById('userText')
    }, 1000)
    postFormData('/post_address', new FormData(form))
})
```

-> views.py

```
@app.route('/post_address', methods=['POST'])
def post_address():
    regex_request = None
    user_request = request.form['userText']
    if user_request:
        try:
            regex_request = parse_search(user_request)
        except MatchNotFound:
            return jsonify({'error': 'erooooooooor'})
    response = geocode(regex_request)
    response_wiki = wiki_api(response['lat'], response['lng'])
    random_sentences = sentences_choice()
    response.update(response_wiki)
    response.update(random_sentences)
    return jsonify(response)
```

regular expressions 101

SAVE & SHARE

Save Regex 96+s

FLAVOR

</> PCRE2 (PHP >=7.3)
</> PCRE (PHP <7.3)
</> ECMAScript (JavaScript)
</> Python 2.7

REGULAR EXPRESSION

:r".+.*adresse=d[e|'|u]\s?(.*?)?\?"

TEST STRING

Donne moi l'adresse d'OpenClassrooms?

GrandPy'Bot 127.0.0.1:5000

Une adresse... Quelle histoire

GrandPy Bot Qui suis-je?

Ecrivez ... Envoyer

Récupérer les informations dans le champ -> requête pour envoyer au serveur.

```
let form = query.Selector('#user-text-form');
recherche sur la page
fetch -> url , method : 'POST', body : data -> script.js
@app.route('/post_address', methods=['POST']) -> views.py
(décorateur -> url indiqué)
.then (response=> response.json()) --> concept de promesse
.catch(error => console.log(error)) --> gestion des erreurs
```

Clic sur le bouton «Soumission» (écoute l'événement)
addEventListener('submit', function(event) -> script.js

Google API Maps

```

params = urllib.parse.urlencode({
    «address»: address,
    «key»: API_KEY,
    «region»: «fr»
})
response = requests.get(
    «https://maps.googleapis.com/maps/api/geocode/json?», params=params)
if response.status_code == 200:
    print(response.url)
    try:
        result = response.json()['results']
    return {
        «formatted_address»: result[0]['formatted_address'],
        «lat»: result[0]['geometry']['location'][«lat»],
        «lng»: result[0]['geometry']['location'][«lng»],
    }
}

```

The screenshot shows the Network tab of a browser developer tools interface. It displays a single request to "https://maps.googleapis.com/maps/api/geocode/json?". The response is a JSON object with the following structure:

```

{
  "status": "OK",
  "results": [
    {
      "geometry": {
        "location": {
          "lat": 48.8975156,
          "lng": 2.3833993
        }
      }
    }
  ]
}

```

```

short_name: "FR"
types:
  0: "country"
  1: "political"
  6:
    long_name: "75019"
    short_name: "75019"
    types:
      0: "postal_code"
    formatted_address: "10 Quai de la Charente, 75019 Paris, France"
  geometry:
    location:
      lat: 48.8975156
      lng: 2.3833993
      type: "Point"

```

The screenshot shows the "Simple Map" example from the Google Maps Platform documentation. On the left, there is a sidebar with navigation links like Overview, Basics, Simple Map, Geolocation, etc. The main content area shows a map centered on Sydney, Australia, with various locations and roads labeled. Below the map, there is a description: "This example creates a map that's centered on Sydney, New South Wales, Australia." To the right of the map, there is a code editor window displaying the HTML, CSS, and JavaScript code for the map.

```

<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <script src="https://polyfill.io/v3/polyfill.min.js?features=default"></script>
    <style type="text/css">
      /* Always set the map height explicitly to define the size of the div
       * element that contains the map. */
      #map {
        height: 100%;
      }

      /* Optional: Makes the sample page fill the window. */
      html, body {
        height: 100%;
        margin: 0;
        padding: 0;
      }
    </style>
    <script>
      let map;

      function initMap() {
        map = new google.maps.Map(document.getElementById("map"), {
          center: { lat: -34.39, lng: 150.64 },
          zoom: 8
        });
      }
    </script>
  </head>
  <body>
    <div id="map"></div>
    <!-- Async script executes immediately and must be after any DOM elements used in callback. -->
    <script>
      src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callback=initMap&libraries=&v=weekly
      async
    </script>
  </body>
</html>

```

The screenshot shows a terminal window titled "[P7]GrandPy_Bot — python - python run.py — 168x24". The output shows the bot starting up and interacting with the Google Maps API to find addresses. The terminal window also includes a Mac OS X desktop interface with various application icons visible in the background.

```

(base) iMac-de-XavGab:~ xavgab$ cd /Users/xavgab/Documents/Python/07_Projet\ 7/[P7]GrandPy_Bot
(base) iMac-de-XavGab:[P7]GrandPy_Bot xavgab$ source env/bin/activate
(base) iMac-de-XavGab:[P7]GrandPy_Bot xavgab$ python run.py
* Serving Flask app "grandpy_bot.views" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 231-388-542
* Detected change in '/Users/xavgab/Documents/Python/07_Projet 7/[P7]GrandPy_Bot/grandpy_bot/google_api.py', reloading
* Restarting with stat
* Debugger is active!
* Debugger PIN: 231-388-542
127.0.0.1 - [27/May/2021 11:20:23] "GET / HTTP/1.1" 200 -
127.0.0.1 - [27/May/2021 11:20:23] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - [27/May/2021 11:20:23] "GET /static/image/icon.png HTTP/1.1" 200 -
127.0.0.1 - [27/May/2021 11:20:23] "GET /static/js/script.js HTTP/1.1" 304 -
https://maps.googleapis.com/maps/api/geocode/json?address=Openclassrooms&key=AIzaSyBraI6bDnr4KJySInct4_vPl0KHGdTdzy4&region=fr
Sous nul doute ... :
127.0.0.1 - [27/May/2021 11:20:40] "POST /post_address HTTP/1.1" 200 -

```

Test Mock -> Google API

```
test_get_returns_correct_coordinates(monkeypatch):
    coordinates = {
        'formatted_address': '10 Quai de la Charente, 75019 Paris, France',
        'lat': 48.8975156,
        'lng': 2.3833993
    }
    class MockResponse:
        status_code = 200
        def json(self):
            return {
                "results": [
                    {
                        "formatted_address": "10 Quai de la Charente, 75019 Paris, France",
                        "geometry": {
                            "location": {
                                "lat": 48.8975156,
                                "lng": 2.3833993
                            }
                        }
                    }
                ]
            }
        def MockRequestsGet(url, params):
            return MockResponse()
    monkeypatch.setattr(requests, "get", MockRequestsGet)
    result = google_api.geocode('OpenClassrooms')
    assert result == coordinates
```

```
cachedir: .pytest_cache
rootdir: /Users/xavgab/Documents/Python/07_Projet 7/[P7]GrandPy_Bot
plugins: arraydiff-0.3, remotedata-0.3.2, doctestplus-0.4.0, openfiles-0.4.0, cov-2.11.1
collected 6 items

grandpy_bot/test/test_geocode.py::test_get_returns_correct_coordinates PASSED
grandpy_bot/test/test_media.py::test_get_returns_correct_extract PASSED
grandpy_bot/test/test_parser.py::test_parser PASSED
grandpy_bot/test/test_parser.py::test_parser_not_match PASSED
grandpy_bot/test/test_routes.py::test_index PASSED
grandpy_bot/test/test_routes.py::test_get_correct_post PASSED
===== 6 passed in 0.32s =====
```

```
[P7]GrandPy_Bot --bash-- 168x91
----- test session starts -----
platform darwin -- Python 3.7.4, pytest-6.2.2, py-1.10.0, pluggy-0.13.0 -- /Users/xavgab/opt/anaconda3/bin/python
cachedir: .pytest_cache
rootdir: /Users/xavgab/Documents/Python/07_Projet 7/[P7]GrandPy_Bot
plugins: arraydiff-0.3, remotedata-0.3.2, doctestplus-0.4.0, openfiles-0.4.0, cov-2.11.1
collected 6 items

grandpy_bot/test/test_geocode.py::test_get_returns_correct_coordinates FAILED
grandpy_bot/test/test_media.py::test_get_returns_correct_extract PASSED
grandpy_bot/test/test_parser.py::test_parser PASSED
grandpy_bot/test/test_parser.py::test_parser_not_match PASSED
grandpy_bot/test/test_routes.py::test_index PASSED
grandpy_bot/test/test_routes.py::test_get_correct_post FAILED
=====
===== FAILURES =====
===== test_get_returns_correct_coordinates =====

monkeypatch = <_pytest.monkeypatch.MonkeyPatch object at 0x7fce8385110>

def test_get_returns_correct_coordinates(monkeypatch):
    coordinates = {
        'formatted_address': 'test',
        'lat': 48.8975156,
        'lng': 2.3833993
    }
    class MockResponse:
        status_code = 200
        def json(self):
            return {
                "results": [
                    {
                        "formatted_address": "test",
                        "geometry": {
                            "location": {
                                "lat": 48.8975156,
                                "lng": 2.3833993
                            }
                        }
                    }
                ]
            }
        def MockRequestsGet(url, params):
            return MockResponse()
    monkeypatch.setattr(requests, "get", MockRequestsGet)
    result = google_api.geocode('OpenClassrooms')
>     grandpy_bot/test/test_geocode.py:37:
address = 'OpenClassrooms'

def geocode(address):
    """Call geocoding google map api
    Args:
        address (STRING): user input
    Returns:
        dictionary: returns values (address, coordinates)
    """
    API_KEY = os.getenv('GOOGLE_API_KEY')
    # urllib.parse.urlencode : transforms url string into its component
    params = urllib.parse.urlencode({
        'address': address,
        'key': API_KEY,
        'region': "fr"
    })
    response = requests.get(
        "https://maps.googleapis.com/maps/api/geocode/json?", params=params)
    if response.status_code == 200:
        print(response.url)
```

monkeypatch = test mock

Test la requête d'une API distante en fonction de sa réponse vérifie si le code fonctionne bien.

Simule une requête --> retourne une réponse

Simuler l'appel réseau de `requests.get('https:')` pour imiter mocker une réponse de l'appel à l'api.

Simulation Status_code et de la méthode json.

`monkeypatch setattr(requests, 'get', MockRequestGet)` --> remplace `requests.get` par la fonction `MockRequestGet`.

MediaWiki

```
def wiki_api(lat, lng):
    coord = str(lat) + «|» + str(lng)

    params = {
        «action»: «query»,
        «prop»: «extracts»,
        «generator»: «geosearch»,
        «exsentences»: «3»,
        «exlimit»: «1»,
        «explaintext»: «True»,
        «ggsradius»: «200»,
        «ggscoord»: coord,
        «gqslimit»: «3»,
        «format»: «json»,
    }
    response_wiki = requests.get(«https://fr.wikipedia.org/w/api.php?»,
                                  params=params)
    print(response_wiki.url)
```

```
if response_wiki.status_code == 200:
    try:
        wiki = response_wiki.json()[«query»][«pages»]
        page_id = list(wiki.keys())[0]
        extract = wiki[page_id][«extract»]
        return {«extract»: extract}
    except KeyError:
        return {«extract»: «mais non en fait je ne sais rien concernant ce lieu...»}
```

```
[27/May/2021 16:14:07] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [27/May/2021 16:14:07] "GET /static/image/icon.png HTTP/1.1" 200 -
127.0.0.1 - - [27/May/2021 16:14:07] "GET /static/js/script.js HTTP/1.1" 304 -
https://fr.wikipedia.org/w/api.php?action=query&prop=extracts&generator=geosearch&exsentences=3&exlimit=1&explaintext=True&ggsradius=3&gqslimit=3&format=json
Sans nul doute ... :
[27/May/2021 16:14:20] "POST /post_address HTTP/1.1" 200 -
```

WIKIPEDIA The Free Encyclopedia

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

Tools
Special pages
Printable version

Languages

API sandbox

Use this page to experiment with the MediaWiki web service API. Refer to the API documentation for further details of API usage: search for page titles matching a certain keyword. Select an action to see more examples. Note that, although this is a sandbox, actions you carry out on this page may modify the wiki.

[Make request](#) [Clear](#)

main

action=query
prop=extracts
generator=geosearch
format=json

Results

Show request data as:

JSON

Request JSON:

```
"prop": "extracts",
"generator": "geosearch",
"exsentences": "3",
"exlimit": "1",
"explaintext": 1,
"ggscoord": "37.7891838|122.4033522",
"ggsradius": "200",
"qqlimit": "3"
```

[Copy](#)

```
{
  "continue": {
    "excontinue": 1,
    "continue": "|"
  },
  "query": {
    "pages": {
      "9293658": {
        "pageid": 9293658,
        "ns": 0,
        "title": "One Montgomery Tower",
        "index": -1,
        "extract": "One Montgomery Tower (also known as Montgomery Tower and formerly Pacific Telesis Tower), part of the Post Montgomery Center complex, is an office skyscraper located at the northeast corner of Post and Kearny Streets in the financial district of San Francisco, California. The 500-foot (150-meter), 38-story tower was completed in 1982, and is connected to the Crocker Galleria mall. It houses around 2,500 office workers (as of 2019). Despite the \"One Montgomery\" branding, the building's main entrance is on 120 Kearny Street, rather than on Montgomery Street."
      },
      "18618509": {
        "pageid": 18618509,
        "ns": 0,
        "title": "Wikimedia Foundation",
        "index": 0
      },
      "42936625": {
        "pageid": 42936625,
        "ns": 0,
        "title": "Foxcroft Building",
        "index": 1
      }
    }
  }
}
```

JSON Données brutes En-têtes

Enregistrer Copier Tout réduire Tout développer Filtrer le JSON

```
{
  continue: {
    excontinue: 1
    continue: "|"
  },
  query: {
    pages: {
      3120649: {
        pageid: 3120649
        ns: 0
        title: "Quai de la Gironde"
        index: -1
        extract: "Le quai de la Gironde est un quai situé le long du canal Saint-Denis, à Paris, dans le 19e arrondissement.\n\n== Situation et accès ==\nIl fait face au quai de Charente, commence au quai de l'Oise et se termine avenue Corentin-Cariou.\nLa ligne du tramway passe sur ce quai."
      },
      3124793: {
        pageid: 3124793
        ns: 0
        title: "Square du Quai-de-la-Gironde"
        index: 0
      },
      11988883: {
        pageid: 11988883
        ns: 0
        title: "Parc du Pont de Flandre"
        index: 1
      }
    }
  }
}
```

GrandPy'Bot

← 9

Test Mock -> Media API

```
def test_get_returns_correct_extract(monkeypatch):
    extract = {'extract':
        «Le quai de la Gironde est un quai situé le long du canal Saint-Denis, à Paris,»
        « dans le 19e arrondissement.\n\n== Situation et accès ==\nIl fait face au quai de la Charente,»
        «commence au quai de l'Oise et se termine avenue Corentin-Cariou.\nLa ligne \u2009 du tramway
    passe sur ce quai.»
    }

    class MockResponse:
        status_code = 200

        def json(self):
            return {«continue»:
                {«excontinue»: 1, «continue»: «||»},
                «query»:
                {«pages»:
                    {«3120649»:
                        {«pageid»: 3120649, «ns»: 0,
                        «title»: «Quai de la Gironde»,
                        «index»: -1,
                        «extract»: «Le quai de la Gironde est un quai situ\u00e9 le long du canal Saint-Denis,»
                        « \u00e0 Paris, dans le 19e arrondissement.\n\n== Situation et acc\u00e8s ==\nIl fait face
                    au quai de la Charente,»
                        «commence au quai de l'Oise et se termine avenue Corentin-Cariou.\nLa ligne \u2009 du
                    tramway passe sur ce quai.»},
                    «3124793»:
                        {«pageid»: 3124793, «ns»: 0,
                        «title»: «Square du Quai-de-la-Gironde», «index»: 0},
                    «11988883»: {«pageid»: 11988883, «ns»: 0, «title»: «Parc du Pont de Flandre», «index»: 1}
                }
            }
        }

    def MockRequestsGet(url, params):
        return MockResponse()

    monkeypatch.setattr(requests, 'get', MockRequestsGet)
    result = media_api.wiki_api(48.8975156, 2.3833993)
    assert result == extract
```

monkeypatch = méthode de test pour simuler et imiter une fonction
Dictionnaire --> exemple de réponse de l'appel MediaWiki API
Test la requête d'une API distante en fonction de sa réponse vérifie si le code fonctionne bien.
Simule une requête --> retourne une réponse

Classe pour simuler et imiter l'appel à l'API et la réponse en json
Attribut de l'objet requests --> requête est un succès (code 200)
Simule la réponse de la requête Api dans le format json (un dictionnaire)

Simuler l'appel réseau de requests.get('https:') pour imiter mocké une réponse de l'appel à l'api.
monkeypatch.setattr(requests, 'get', MockRequestGet) --> remplace requests.get par la fonction
MockRequestGet.

Style de l'application

-> Suppression du npm

```
--> node_modules  
--> package.json  
--> package-lock.json
```

-> script.css

```
body, html, footer {  
font-family: helvetica, sans-serif;  
height: 100%;  
background-color: #aeb6bf;  
}
```

```
.message-error{  
background: #CD5C5C;  
border-radius: 20px 20px 2px 20px;  
max-width: 300px;  
color: white;  
margin-left: auto;  
padding: 13px;  
margin-bottom: 15px;  
margin-top: 15px;  
}
```

-> Bootstrap

```
<link href=>https://cdn.jsdelivr.net/npm/  
bootstrap@5.0.0/dist/css/bootstrap.min.css>
```

```
rel=>stylesheet
```

```
integrity=>sha384-wEmelV1mKuiNpC+IOBj7aAzPcEZee  
di5yW5f2yOq55WWLwNGmvx4Um1vskeMj0  
crossorigin=>anonymous>
```

-> style des icônes

```
<link rel=>stylesheet href=>https://cdnjs.cloudflare.  
com/ajax/libs/ionicons/2.0.1/css/ionicons.min.css>
```

GrandPy'Bot

Où se trouve OpenClassrooms?

Envoyer

Gabrielle Azadian

GitHub LinkedIn

Veuillez remplir le champ du formulaire...

Où se trouve OpenClassrooms?

Envoyer

GrandPy'Bot

-> Demande correcte

Où se trouve Openclassrooms

gabrielle2801/P7_GrandPy-Bot B Introduction - Bootstrap v.5.0 GrandPy'Bot 127.0.0.1:5000

Où se trouve Openclassrooms?

Veuillez remplir le champ du formulaire...

Si je me rappelle bien, c'est : 10 Quai de la Charente, 75019 Paris, France

Où se trouve OpenClassrooms?

Envoyer

Gabrielle Azadian

←12

GrandPy'Bot

-> Demande incorrecte

Quelle heure est-il?

Où se trouve le musée du Quai Branly

--> il manque ?

Où se trouve Openclassrooms?

Si je me rappelle bien, c'est : 10 Quai de la Charente, 75019 Paris, France



Il fut un temps où je venais flâner dans ce lieu ... : Le quai de la Gironde est un quai situé le long du canal Saint-Denis, à Paris, dans le 19^e arrondissement. == Situation et accès == Il fait face au quai de la Charente, commence au quai de l'Oise et se termine avenue Corentin-Cariou. La ligne du tramway passe sur ce quai.

La demande est incorrecte ... Veuillez reformuler

La demande est incorrecte ... Veuillez reformuler

Quelle heure est-il?

Envoyer

La demande est incorrecte ... Veuillez reformuler

Où se trouve la musée du Quai Branly?

Envoyer

reformuler

La demande est incorrecte ... Veuillez reformuler

Tiens cette adresse m'est inconnue

GrandPy'Bot

-> Demande incorrecte

Donne moi l'adresse de annie?

--> pas d'article wikipédia trouvé

Où se trouve le rrrrrrrrrrrrrrrrr?

--> adresse inconnue

Donne moi l'adresse de annie?

Sans nul doute ... : 39 Place du Champ de Foire, 36140 Aigurande, France



Il fut un temps où je venais flâner dans ce lieu ... : mais non en fait je ne sais rien concernant ce lieu...

Où se trouve OpenClassrooms?

Envoyer

Gabrielle Azadian



Tiens cette adresse m'est inconnue

Où se trouve le rrrrrrrrrrrrrrrrr?

Envoyer

GrandPy'Bot

-> inspecteur

```
14
15  .then(response => response.json())
16  )
17  .catch(error => console.log(error))
18 }
19
20 formData("post_address", new FormData(form))
21 event.preventDefault();
22
23 spinner();
24 if(response.error) {
25   .then(response => {
26     console.log(response);
27     const error = response.error;
28     if(response.error){
29       error.textContent = "Une erreur s'est produite lors de l'extraction des informations. Veuillez réessayer plus tard.";
30       error.setAttribute('class', 'message-error');
31       document.querySelector('.question-address').append(error);
32       reset();
33     } else{
34       setImmutut(()=> (extract(response)),1800);
35     }
36   })
37 }
```

Erreurs Avertissements Journaux Informations Débogage CSS XHR Requêtes

> Objet d'erreur : "Le quai de la Gironde est un quai situé le long du canal Saint-Denis, à Paris, dans le 19e arrondissement." Situation et accès : null
Tout l'ancien nom de la Charente, commence au quai de la Gironde jusqu'à l'intersection avec le canal Saint-Denis.
"10 Quai de la Charente, 75019 Paris, France", lat: 48.8975156, lon: 2.3833993, sentences: "Sous nul doute ... ", sentence_wiki: "L'histoire est la mémoire du monde..."
De se trouve Openclassrooms!

L'histoire est la mémoire du monde, et voici celle de ce lieu magique : Le quai de la Gironde est un quai situé le long du canal Saint-Denis, à Paris, dans le 19e arrondissement. == Situation et accès == Il fait face au quai de la Charente, commence au quai de l'Oise et se termine avenue Corréting-Carrou. La ligne du tramway passe sur ce quai.

SyntaxError: JSON.parse: unexpected character at Line 1 column 1 of the JSON data
undefined
① Unclosed tag promise: TypeDerror: response is undefined
 ↗
 En savoir plus

⚠ Propriété « speak » inconnue. Déclaration abandonnée.
⚠ Erreur d'analyse de la valeur pour « webkit-transform-size-adjust ». Déclaration abandonnée.
⚠ Propriété « -ms-grid-column-gap » inconnue. Déclaration abandonnée.
⚠ Jeu de règles ignoré suite à un mauvais sélecteur.
⚠ Pseudo-classe ou pseudo-élément « -moz-placeholder-shown » inconnue. Jeu de règles ignoré suite à un mauvais sélecteur.
⚠ Pseudo-classe ou pseudo-élément « -ms-placeholder-shown » inconnue. Jeu de règles ignoré suite à un mauvais sélecteur.
⚠ Erreur d'analyse de la valeur pour « transform ». Déclaration abandonnée.

ionicons-min.css:1111
reboot.scss
_forms.scss
_form-repo.scss
_floating-labels.scss
_floating-labels.scss
_floating-labels.scss
_floating-labels.scss
127.0.0.1:5000

GrandPy'Bot