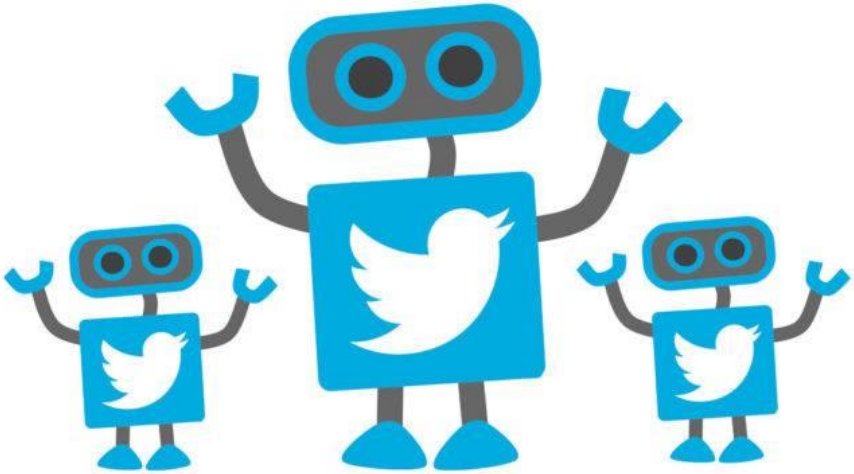


Développer un bot twitter

Par Alexandra et Lou

Les bots sur Twitter sont un bon moyen de récupérer de l'info facilement, de suivre ce que disent des groupes de personnes ou de publier régulièrement des tweets, intéressants ou non

Avec Python, tout peut se faire rapidement et gratuitement, il suffit d'installer le module tweepy

Pour tout installer:
Python: https://www.python.org/downloads/
Tweepy: http://www.tweepy.org/

Toute la doc sur ce qui est présenté ici est disponible sur:
https://developer.twitter.com/

Créer un accès développeur
..... pge 1 et 2

Créer une API
..... pge 3 et 4

Communiquer avec Twitter
..... pge 5 et 6

Traiter du texte en python
..... pge 7 et 8

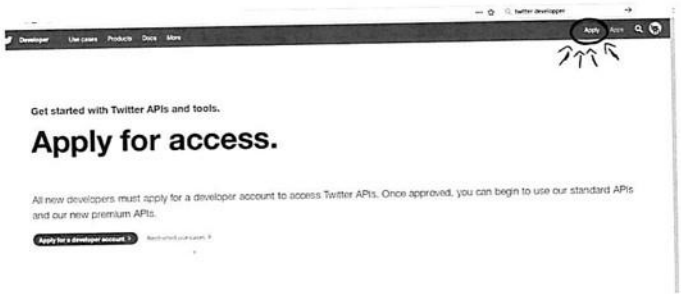
Des idées de bots !
..... pge 9 et 10

Créer un accès développeur

Pour commencer, il faut un compte twitter.
On pourra créer plusieurs bots avec

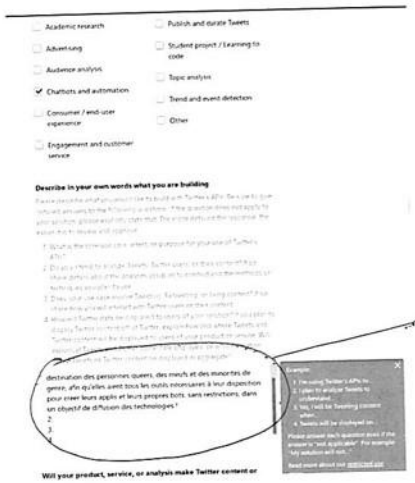
Pour créer un compte, il faut:
- Un mail valide (un mail poubelle ira très bien)
- Un numéro de téléphone

1ère étape: Se rendre sur https://developer.twitter.com



On choisit le compte à associer

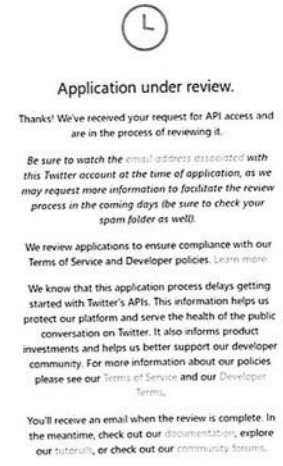
Twitter demande quelques infos :
-> mettre « usage personnel »
-> un nom quelconque

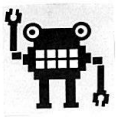


Twitter veut savoir pourquoi on demande l'accès, il faut 300 caractère (en anglais si possible), il faut baratiner un peu

La demande est analysée, ça peut prendre plusieurs heures

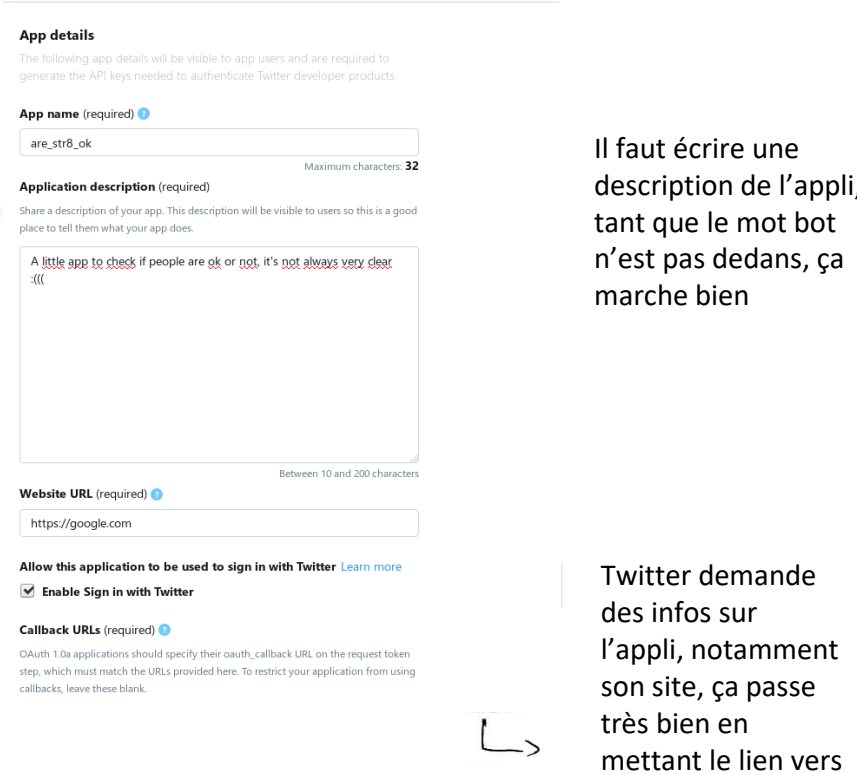
Avec une demande en anglais ça va plus vite, juste quelques secondes





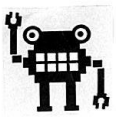
Créer des accès API

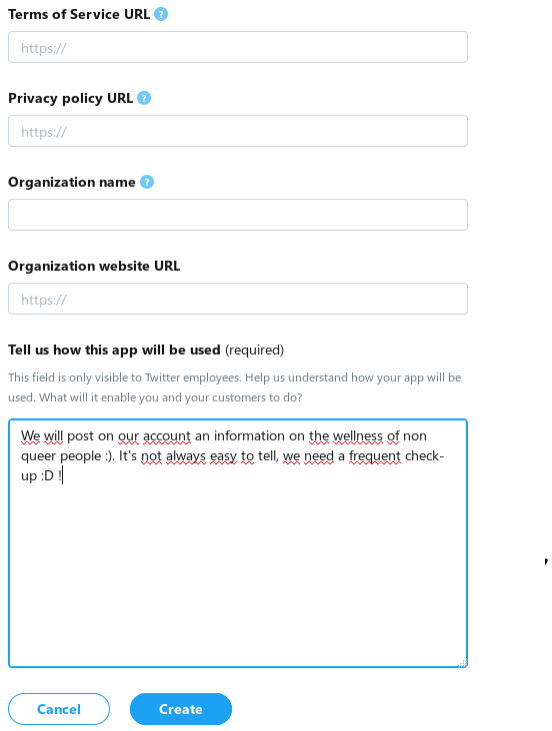
Tout se passe dans « Create an App »



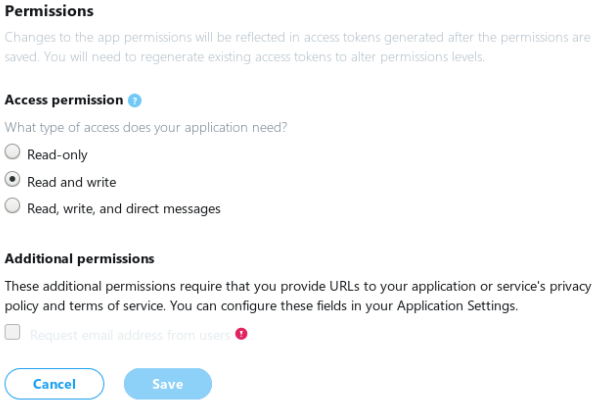
Il faut écrire une description de l'appli, tant que le mot bot n'est pas dedans, ça marche bien

Twitter demande des infos sur l'appli, notamment son site, ça passe très bien en mettant le lien vers Google





Encore une description mais plus longue: ne pas mettre le mot 'bot' pour éviter de se faire bloquer



Par défaut 'read and write' suffit, ajout 'direct messages' si on veut pouvoir parler en message privé

Comment se connecter à Twitter ?

Installation de la bibliothèque

On utilise la bibliothèque **tweepy** qui va permettre de se connecter à Twitter depuis Python.
Pour installer tweepy en ligne de commande :

- *pip3 install tweepy*

Toute la doc sur ce qui est expliqué ici est disponible à : https://developer.twitter.com/en/docs/

Envoyer des messages avec python

Authentification

Pour se connecter à twitter, on utilise « OauthHand » :

```
import tweepy

consumer_key = "xxxxxxxxxxxxxxxxxxxxxxxxx"
consumer_secret = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

access_token = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
access_token_secret = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token,access_token_secret)

api = tweepy.Api(auth)
```

Conseils : il faut faire attention avec les clés et les tokens secrets, qui ne doivent pas être publique (quelqu'un pourrait « usurper l'identité » du bot). Ces clés peuvent être stockées dans un fichier qui pourra être lu dans la fonction *authentification*.

Envoi d'un message

On va ensuite envoyer un message sur Twitter, à partir de Python, et appeler une méthode de l'API :

```
api.update_status(status = "Bonjour depuis python")
```

Steamer

Pour écouter et récolter des tweets, tweepy utilise un objet : le StreamListener.
Cet objet a 3 méthodes principales :

- filter, qui filtre les tweets en fonction de ce qu'on veut voir (des mots clés, des utilisateurs...)
- on_status, qui réagit aux tweets récupérs par filter.
- On_error qui explique quoi faire s'il y a un problème

On peut créer notre propre Stream, en choisissant ce qu'il va faire des tweets qu'il récupère

```
class StreamListener(tweepy.StreamListener): # Indique qu'on va partir de l'objet qui existe
    def on_status(self, status):
        statusText = status.text #récupère le texte du tweet

        if statusText == "Bonjour twitter !"
        else:
            api.update_status("Bonjour !", in_reply_to_status_id=status.id)


streamListener = StreamListener() # Crée un Stream
stream = tweepy.Stream(auth = api.auth, listener=streamListener) #Lance le Streaming
stream.filter(track=['Bonjour']) #Le Stream va chercher le mot 'Bonjour' dans les tweets
```

Ce bot récupère tous les tweets contenant le mot 'Bonjour' et lorsque le texte exact est 'Bonjour Twitter', il répond ;

On peut filtrer les tweets sur plusieurs paramètres.

Le nom du compte qui publie

```
stream.filter(follow = ["mut_tuto","le_RESET"])
```

-> on récupère tout ce que disent le reset et le compte tuto

Des mots clés

```
stream.filter(track = ["féminisme","queer"])
```

-> On récupère tous les tweets qui contiennent le mot « féminisme » et le mot « queer »

Une localisation

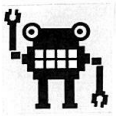
```
stream.filter(locations = [[-74,40,-73,41]] )
```

-> On récupère tous les tweets publiés à ces coordoonées géographiques

Récupérer une information d'un tweet

```
# L'identifiant du tweet
tweet.id_str

# Le compte qui poste
tweet.user.name # Son @
tweet.user.screen_name # Son nom
tweet.user.description # Sa bio

# Le texte du tweet
tweet.text

# La date de publication du tweet
tweet.created_at

# Les hashtags du tweet
tweet.entities.hastags
```

Pour plus d'éléments à récupérer dans un tweet :
https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/basic-stream-parameters



Créer et utiliser du texte en python

Créer du texte

En python, le texte se met entre guillemets

```
texte = "Le bot parle"
texte
```

```
'Le bot parle'
```

Diviser ou rassembler du texte

Pour créer un texte avec plusieurs bouts, on utilise « + »

```
la_phrase = "ce bot est "
la_phrase = la_phrase + "vraiment cool"

la_phrase
```

```
'ce bot est vraiment cool'
```

La fonction split() permet de diviser un texte

```
"Combien de mots dans cette phrase ?".split(" ")
```

```
['Combien', 'de', 'mots', 'dans', 'cette', 'phrase', '?']
```

On indique ici le caractère sur lequel diviser, par exemple « »

Chercher un mot ou une expression

```
import re
```

→ On utilise un package, c'est-à-dire un ensemble de fonctions, appelé « re »

re.findall(« mot », « texte ») recherche si « mot » apparait dans « texte »

↳ Si oui, elle donne la liste des mots

↳ Si non, elle renvoit une liste vide

```
trouve = re.findall("bot","mon bot twitter est le plus beau")
trouve
```

```
['bot']
```

```
trouve = re.findall("intelligent","mon bot twitter est le plus beau")
trouve
```

```
[]
```

Réagir à un texte

Pour réagir, on utilise des conditions

→ == pour voir si 2 textes sont exactement les mêmes

→ != pour voir si 2 textes sont différents

```
"Mon bot" == "Mon bot"
```

```
True
```

```
"Mon bot" == "Ton bot"
```

```
False
```

```
"Mon bot" != "Ton bot"
```

```
True
```

If permet de dire quoi faire quand la condition est vérifiée

```
la_chaine = "Le bot est très vénère"

if re.findall("vénère", la_chaine) != []:
    print("le bot n'a pas l'air content")
else:
    print("le bot est en pleine forme")
```

SI tu trouves le mot « Vénère », ALORS écris « le bot n'est pas content »
SINON écris, « le bot est en pleine forme »

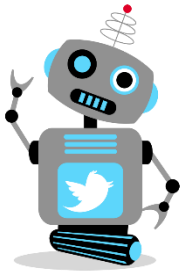
Des exemples de bots

Une page Wikipédia par jour

Tous les jours, le bot poste une page Wikipédia au hasard
On peut utiliser:
https://fr.wikipedia.org/wiki/Special:Page_au_hasard

Le bot Kamoulox

Utilise des listes de sujets, verbes et compléments pour créer une phrase aléatoire et la poster
Pour l'aléatoire en python: package numpy

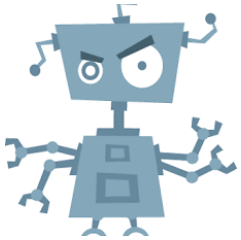


Poste des images de chat

- Si prises directement sur la machine: api.update_with_media()
- Sinon, on peut développer une API Google qui récupère des images sur Google Images

Le bot qui répond à certains tweets sexistes

Ecoute la phrase « Je ne suis pas sexiste mais … » et répond « si tu l'es »

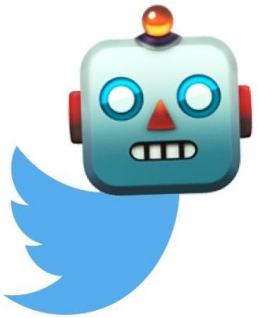


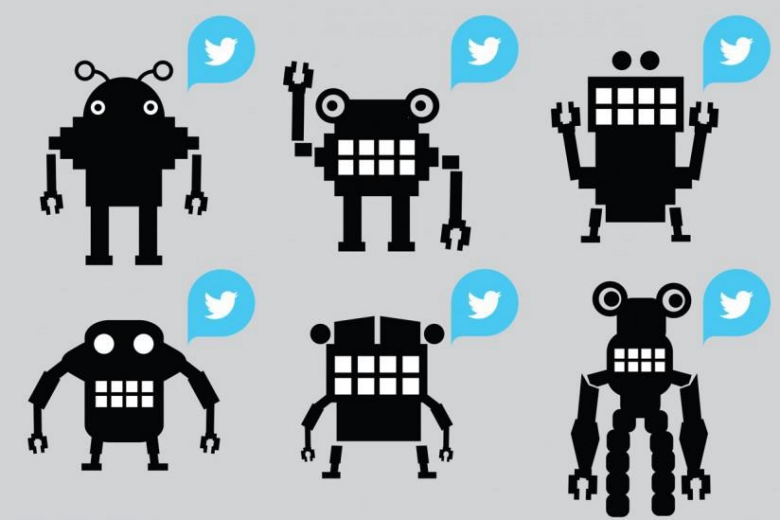
Le bot qui spoile

Ecoute quand quelqu'un parle d'une série et lui spoile la fin

Le bot qui s'inquiète pour les hétéros

Ecoute si quelqu'un demande « Est-ce que les hétéros vont bien » et répond une phrase aléatoire pour dire que non, toujours pas ☹





Have fun !