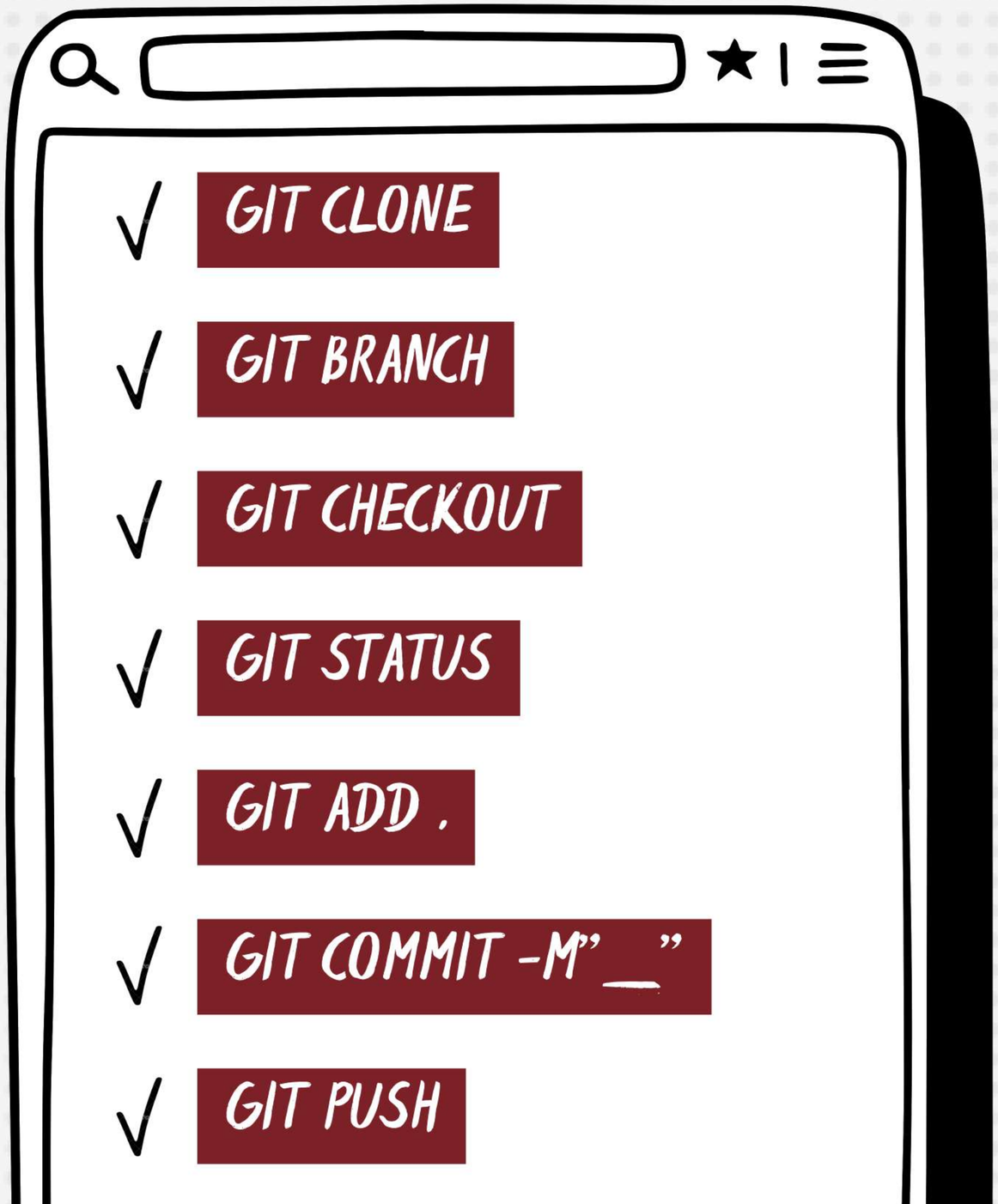
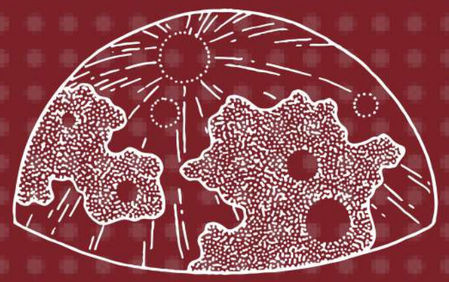
& ça va être du lourd ;)

API STAR WARS

Lou



CMD UTILISÉS



RESSOURCES

par Lou

FONCTION

permet d'avoir une seule fonction pour récupérer l'ensemble des données d'une ressource spécifique

RESSOURCES

DEMO JUPYTER

```python
import requests
import pandas as pd

def data(ressources):
    next_page = f"https://swapi.dev/api/{ressources}/"
    all_results = []

    while next_page:
        response = requests.get(next_page)
        if response.status_code == 200:
            data = response.json()

            all_results.extend(data['results'])

            next_page = data['next']
        else:
            print("Erreur lors de la récupération des données.")
            return None

    df = pd.DataFrame(all_results)
    return df

ressources_df = data("vehicles")
if ressources_df is not None:
    display(ressources_df)

```

PEOPLES

par Lou

FONCTION 1

récupération des nom de personnages en listing

FONCTION 2

récupération des nom de personnages et leur apparition dans les films

```python
def get_characters_names():
    next_page = "https://swapi.dev/api/people/"
    all_results = []

    while next_page:
        response = requests.get(next_page)
        if response.status_code == 200:
            data = response.json()

            all_results.extend(data['results'])

            next_page = data['next']
        else:
            print("Fail, essaye encore !")
            return None

    names = [result['name'] for result in all_results]
    return names

character_names = get_characters_names()

if character_names is not None:
    result_string = ", ".join(character_names)
    print("Personnages :", result_string)
```

LISTING

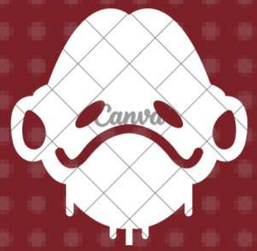
Récupération de data que je met en listing, séparé par une virgule.

```
'Personnages :'

'Luke Skywalker, C-3PO, R2-D2, Darth Vader, Leia Organa, Owen Lars, Beru Whitesun lars, R5-D4, Biggs Darklighter, Obi-Wan Kenobi, Anakin Skywalker, Wilhuff Tarkin, Chewbacca, Han Solo, Greedo, Jabba Desilijic Tiure, Wedge Antilles, Jek Tono Porkins, Yoda, Palpatine, Boba Fett, IG-88, Bossk, Lando Calrissian, Lobot, Ackbar, Mon Mothma, Arvel Crynyd, Wicket Systri Warrick, Nien Nunb, Qui-Gon Jinn, Nute Gunray, Finis Valorum, Padmé Amidala, Jar Jar Binks, Roos Tarpals, Rugor Nass, Ric Olié, Watto, Sebulba, Quarsh Panaka, Shmi Skywalker, Darth Maul, Bib Fortuna, Ayla Secura, Ratts Tyerel, Dud Bolt, Gasgano, Ben Quadinaros, Mace Windu, Ki-Adi-Mundi, Kit Fisto, Eeth Koth, Adi Gallia, Saesee Tiin, Yarael Poof, Plo Koon, Mas Amedda, Gregar Typho, Cordé, Cliegg Lars, Poggle the Lesser, Luminara Unduli, Barriss Offee, Dormé, Dooku, Bail Prestor Organa, Jango Fett, Zam Wesell, Dexter Jettster, Lama Su, Taun We, Jocasta Nu, R4-P17, Wat Tambor, San Hill, Shaak Ti, Grievous, Tarfful, Raymus Antilles, Sly Moore, Tion Medon'
```

CHARACTERS & FILMS

```
Nom du personnage: Luke Skywalker
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'Revenge of the Sith']

Nom du personnage: C-3PO
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'The Phantom Menace', 'Attack of the Clon
es', 'Revenge of the Sith']

Nom du personnage: R2-D2
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'The Phantom Menace', 'Attack of the Clon
es', 'Revenge of the Sith']

Nom du personnage: Darth Vader
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'Revenge of the Sith']

Nom du personnage: Leia Organa
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'Revenge of the Sith']

Nom du personnage: Owen Lars
Films où il apparaît: ['A New Hope', 'Attack of the Clones', 'Revenge of the Sith']

Nom du personnage: Beru Whitesun lars
Films où il apparaît: ['A New Hope', 'Attack of the Clones', 'Revenge of the Sith']

Nom du personnage: R5-D4
Films où il apparaît: ['A New Hope']

Nom du personnage: Biggs Darklighter
Films où il apparaît: ['A New Hope']

Nom du personnage: Obi-Wan Kenobi
Films où il apparaît: ['A New Hope', 'The Empire Strikes Back', 'Return of the Jedi', 'The Phantom Menace', 'Attack of the Clon
es', 'Revenge of the Sith']
```

```python
import requests

def characters_and_films():
    characters_url = "https://swapi.dev/api/people/"
    response = requests.get(characters_url)

    if response.status_code == 200:
        characters_data = response.json()['results']
        for character in characters_data:
            films = [requests.get(film).json()['title'] for film in character['films']]
            print(f"Nom du personnage: {character['name']}")
            print("Films où il apparaît:", films)
            print()
    else:
        print("Erreur lors de la récupération des données des personnages.")

characters_and_films()
```



SPECIES

par Lou

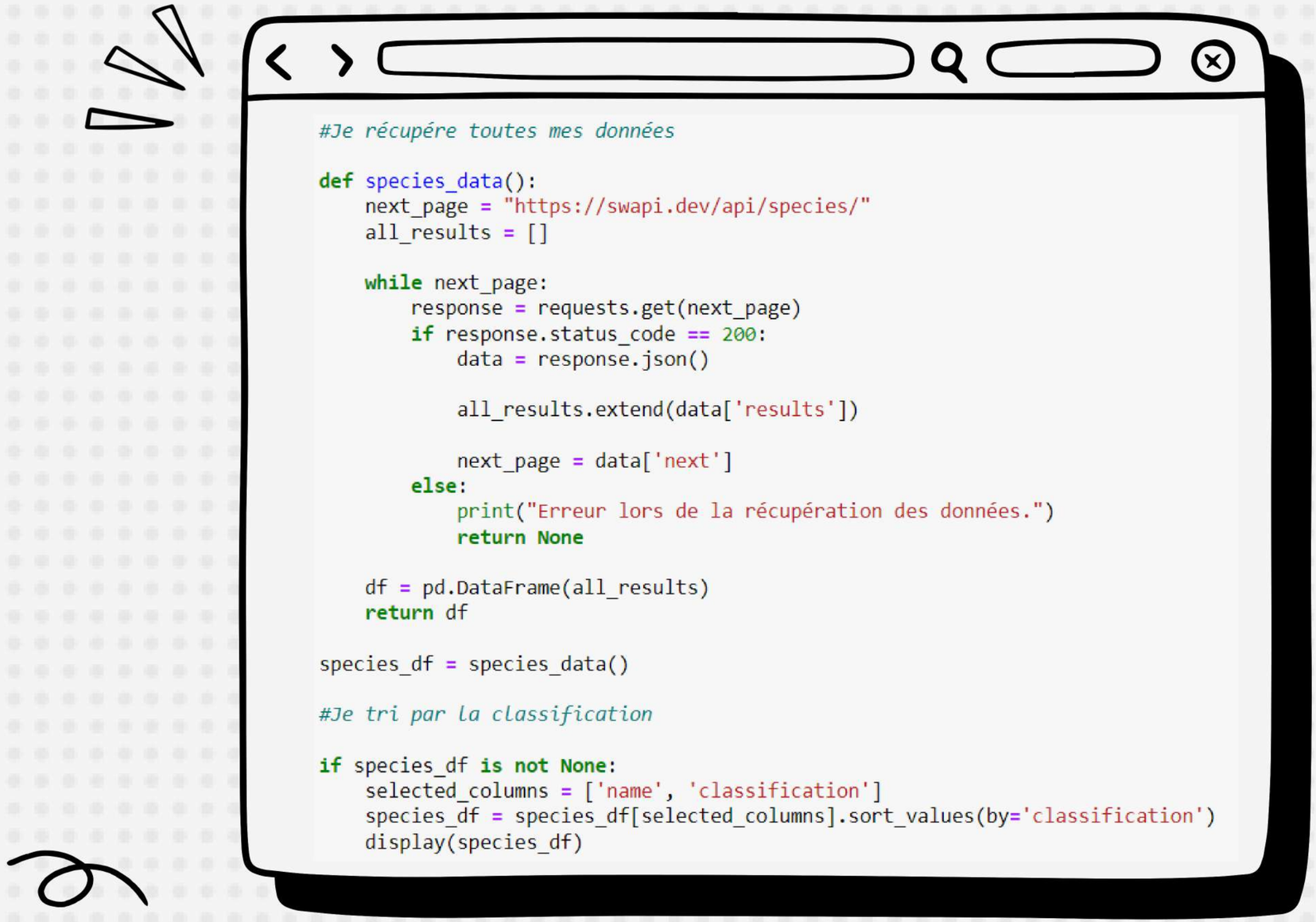
FONCTION 1

récupération de données et tri par “classification”

FONCTION 2

tri et classification des données dans des dataframes dédiés

	name	classification
30	Besalisk	amphibian
31	Kaminoan	amphibian
7	Mon Calamari	amphibian
20	Nautolan	amphibian
26	Chagrian	amphibian
11	Gungan	amphibian
1	Droid	artificial
4	Hutt	gastropod
27	Geonosian	insectoid
21	Zabrak	mammal
22	Tholothian	mammal
24	Quermian	mammal
0	Human	mammal
32	Skakoan	mammal
33	Muun	mammal
34	Togruta	mammal
28	Mirialan	mammal
19	Cerean	mammal
36	Pau'an	mammal
2	Wookie	mammal
13	Dug	mammal



```python
#Je récupére toutes mes données

def species_data():
    next_page = "https://swapi.dev/api/species/"
    all_results = []

    while next_page:
        response = requests.get(next_page)
        if response.status_code == 200:
            data = response.json()

            all_results.extend(data['results'])

            next_page = data['next']
        else:
            print("Erreur lors de la récupération des données.")
            return None

    df = pd.DataFrame(all_results)
    return df

species_df = species_data()

#Je tri par la classification

if species_df is not None:
    selected_columns = ['name', 'classification']
    species_df = species_df[selected_columns].sort_values(by='classification')
    display(species_df)
```

CLASSIFICATION

Classification: mammal

	name	classification
19	Cerean	mammal
13	Dug	mammal
8	Ewok	mammal
0	Human	mammal
28	Mirialan	mammal
33	Muun	mammal
36	Pau'an	mammal
24	Quermian	mammal
32	Skakoan	mammal
9	Sullustan	mammal
22	Tholothian	mammal
34	Togruta	mammal
12	Toydarian	mammal
2	Wookie	mammal
5	Yoda's species	mammal
21	Zabrak	mammal

Classification: mammals

	name	classification
14	Twi'lek	mammals

Classification: amphibian

	name	classification
30	Besalisk	amphibian
26	Chagrian	amphibian
11	Gungan	amphibian
31	Kaminoan	amphibian
7	Mon Calamari	amphibian
20	Nautolan	amphibian

Classification: artificial

	name	classification
1	Droid	artificial

Classification: gastropod

	name	classification
4	Hutt	gastropod

Classification: insectoid

	name	classification
27	Geonosian	insectoid

Classification: reptile

	name	classification
15	Aleena	reptile
35	Kaleesh	reptile
6	Trandoshan	reptile

Classification: reptilian

	name	classification
29	Clawdite	reptilian

Classification: sentient

	name	classification
3	Rodian	sentient

Classification: unknown

	name	classification
23	Iktotchi	unknown
25	Kel Dor	unknown
10	Neimodian	unknown
18	Toong	unknown
16	Vulptereen	unknown
17	Xexto	unknown

```python
#Je recupere les données de mon api species

def species_data():
    url = "https://swapi.dev/api/species/"
    all_results = []

    while url:
        response = requests.get(url)
        if response.status_code == 200:
            data = response.json()
            all_results.extend(data['results'])
            url = data['next']
        else:
            print("Erreur lors de la récupération des données.")
            return None

    return pd.DataFrame(all_results)

#Je classifie, tri et stock mes informations en fonction de leur classification
#tout en gardant les données de "name" & "classification" dans des dataframes dédiés

def species_by_classification(df): #
    if not df.empty:
        df = df[['name', 'classification']].sort_values(by='name')
        for classification, group in df.groupby('classification'):
            print(f"Classification: {classification}")
            display(group)
            print()
    else:
        print("Le DataFrame est vide.")

species_df = species_data()

if species_df is not None:
    species_by_classification(species_df)
```