DATASCIENCE, LEARNING AND APPLICATIONS

DALAS - Data acquisition via web scrapping

20 janvier 2024

Laure Soulier - Nicolas Baskiotis

SCIENCES SORBONNE UNIVERSITÉ

ISIR INSTITUT DES SYSTÈMES INTELLIGENTS ET DE ROBOTIQUE

MLIA Machine Learning & Deep Learning for Information Access

Inspired from Olivier Schwander's lecture and INSEE's "Formation web scrapping"

Web scrapping

- **Web Scrapping :** content extraction of websites
- **Web crawling :** search of new URLs on internet or on a specific website

But both can collaborate : crawler for gathering URLs and then, scrapping for extracting the content.

Examples

- Price follow-up, temporal analysis
- Sentiment analysis (comment on products, feedbacks, ...)
- Potential clients (Pages Jaunes, Google Maps, ...)
- Meta-engines (lastminute.com, kiwi.com), gathering info from different websites

Is it legal ?

- Not illegal, but the use of data might be regulated (GDRP, CNIL, intellectual property law, data gouvernance act, jurisprudence, licences, ...)
- Rules depend on the country you are and the data you scrap.
- Sell or forward scrapped data might be illegal

L'article 323-3 du code pénal

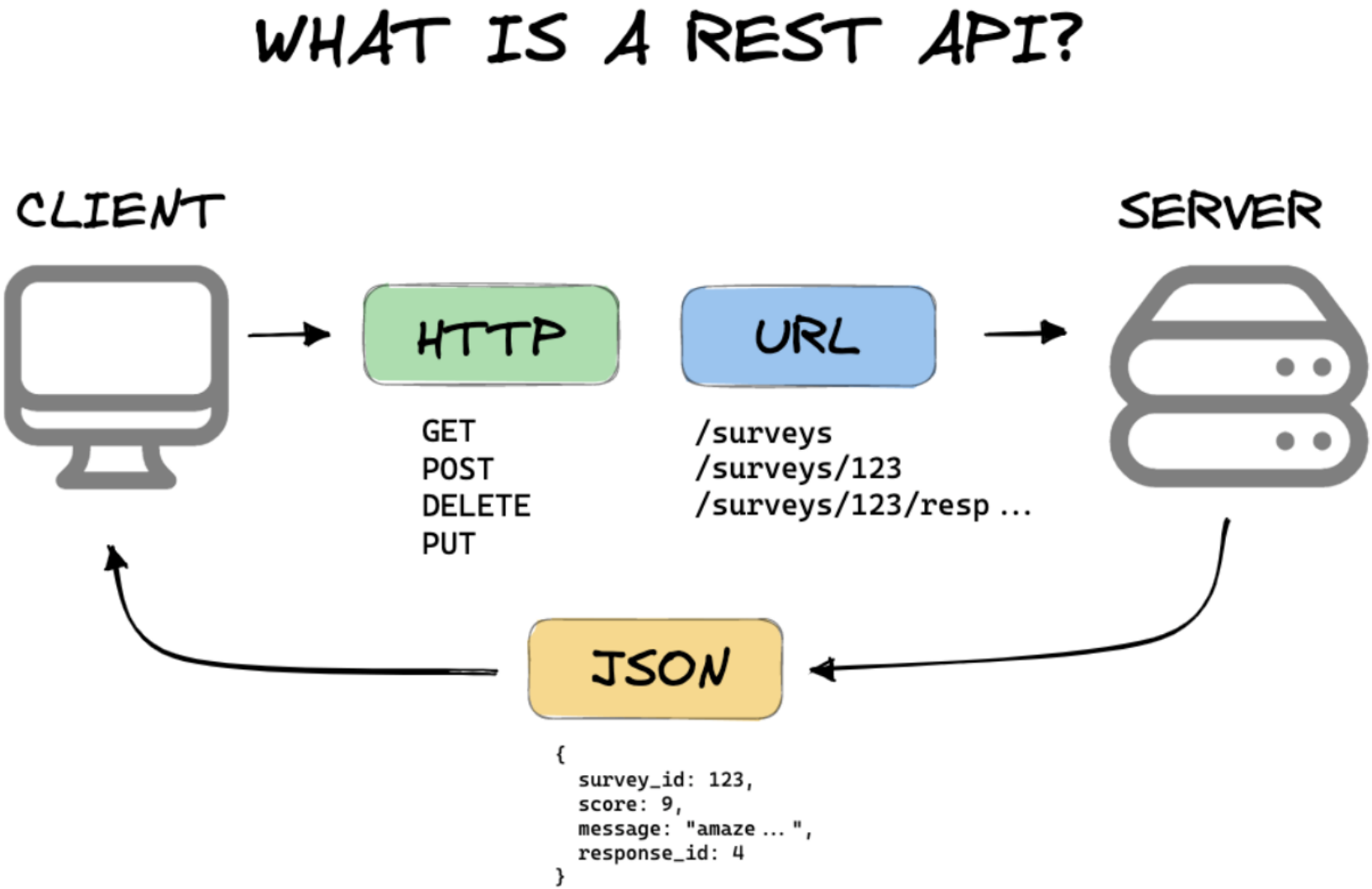
"Le fait [...] d'extraire, de détenir, de reproduire [...] frauduleusement les données qu'il [le site] contient est puni de cinq ans d"emprisonnement et de 150 000e d'amende"

Web service

Basic notions of the web

- Client-server architecture (request/answer)
- Web service : Application/software function allowing an inter-operable communication over the network between different platforms/tools. They use standard and open protocols.
- Most used architecture style : Representational State Transfer (REST). The URI is important (allows to identify resources and values, available operations : GET, POST, PUT, DELETE, ...)
- Others protocols/architecture styles SOAP and WSDL (more complexed and more structured)

Web services



Uniform Resource Locator (URL)

Format

http://example.com/chemin/vers/la/ressource?arg1=valeur1&arg2=valeur2

- http : protocol
- example.com : server
- path/to/the/resource : identifier of the resource
- ? : everything that follows is an arg
- arg1=valeur1&arg2=valeur2 : args

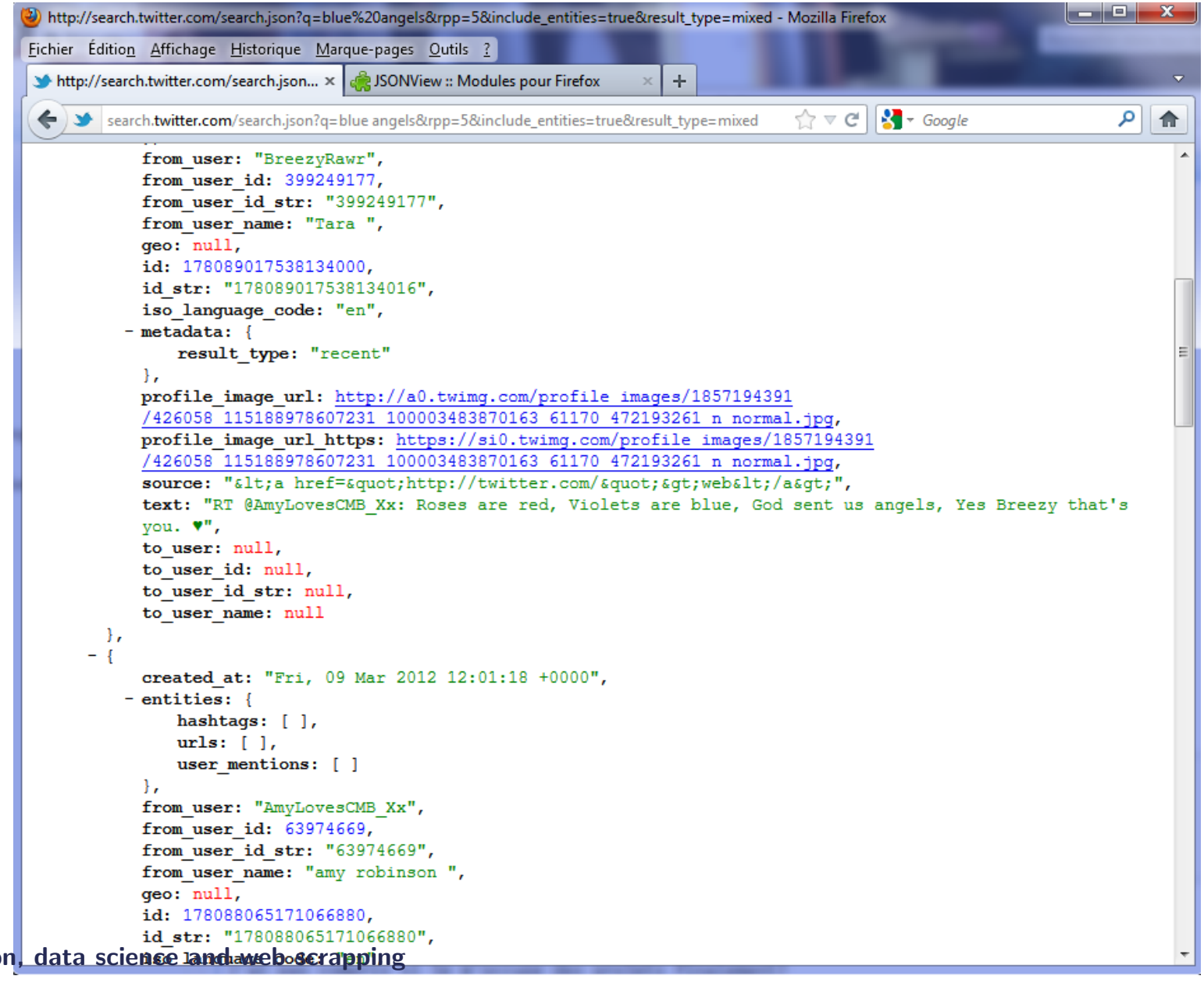
Requesting API Rest

- We use the HTTP protocol to call CRUD actions :

```
 //Get all devices: HTTP GET
http://api.example.com/device-management/managed-devices
//Create new Device: HTTP POST
http://api.example.com/device-management/managed-devices
//Get device for given Id: HTTP GET
http://api.example.com/device-management/managed-devices/{id}
//Update device for given Id: HTTP PUT
http://api.example.com/device-management/managed-devices/{id}
//Delete device for given Id: HTTP DELETE
http://api.example.com/device-management/managed-devices/{id}
```

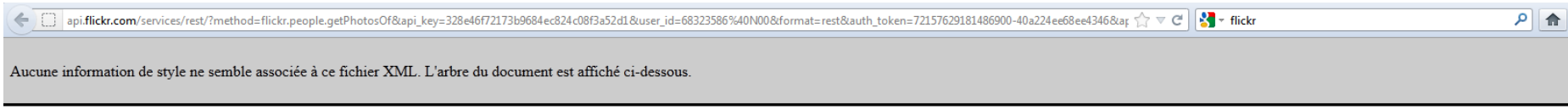
For more info : https://restfulapi.net/resource-naming/

API REST : Twitter



API REST : Flickr

SCIENCES SORBONNE UNIVERSITÉ




```
− <rsp stat="ok">
  − <photos page="1" pages="4" perpage="20" total="70" has_next_page="1">
      <photo id="6818874820" owner="29893924@N08" secret="75a1264454" server="7199" farm="8" title="::Feliz Dia de la Mujer::" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6857161799" owner="34829221@N00" secret="e42ceae339" server="7202" farm="8" title="What would you ask?" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6852635517" owner="8705967@N02" secret="725d819bcf" server="7152" farm="8" title="V de Venezuela, V de VOTA!!!" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6689735273" owner="48664097@N02" secret="b1735cdd60" server="7021" farm="8" title="Llamarada" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6642489581" owner="25257508@N03" secret="a4699ec6d4" server="7025" farm="8" title="Things I Love Thursday" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6641187815" owner="48739681@N05" secret="9c5144ca5a" server="7016" farm="8" title="~ Feliç Nit de Reis ~ Feliz Noche de Reyes ~" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6596007895" owner="34829221@N00" secret="961ed2d366" server="7165" farm="8" title="2011 in Review" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6593866295" owner="38838924@N03" secret="f867c83f24" server="7153" farm="8" title="Mi 2011 en flickr" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6590116673" owner="8705967@N02" secret="23bd13d66b" server="7018" farm="8" title="Navidad" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6587696631" owner="8332135@N03" secret="63cc7bd96a" server="7142" farm="8" title="The sky is the limit..." ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6586167143" owner="63307805@N00" secret="cfaebfd377" server="7007" farm="8" title="my year in pictures" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6581955027" owner="39307146@N08" secret="7b56b63043" server="7171" farm="8" title="My year in pictures" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6566465059" owner="23523125@N08" secret="4c73a0eb85" server="7147" farm="8" title="Feliz Navidad...." ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6564959719" owner="29219049@N06" secret="420005be9d" server="7153" farm="8" title="51/52- Feliz Navidad!!!" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6560620287" owner="40654531@N05" secret="8f0dc0970a" server="7001" farm="8" title="mi navidad, tu navidad? Explored Dec 23, 2011 #6 Muchas Gracias!" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6553379789" owner="49762065@N08" secret="20b62858fe" server="7002" farm="8" title="-" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6538048271" owner="37763325@N08" secret="66c295ff74" server="7171" farm="8" title="Felices Fiestas! ~ Happy Holidays!" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6526003925" owner="66881369@N06" secret="ebb8093294" server="7020" farm="8" title="Family lunch" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6475299671" owner="43616712@N03" secret="aa8110b0f7" server="7146" farm="8" title="lovely support" ispublic="1" isfriend="0" isfamily="0"/>
      <photo id="6358869783" owner="25257508@N03" secret="2051f88c38" server="6019" farm="7" title="Friday Favorites" ispublic="1" isfriend="0" isfamily="0"/>
    </photos>
  </rsp>
```

Request and Answer format

→ **HTTP Request**

- Other element in the request : meta-data, encoding, authentication, cookies, session tracking, ...
- Answer format : CML, JSON, Binary, proprietary, ...

→ **Authentication mechanisms**

- Login + password sent every time : rare
- Login + password sent once then secret : token mechanism
- Secret in URL : https://example.com/e1WnQ10c (Doodle for instance)
- Two factor login : sms, email, ...

Requesting REST API in Python

- **Requests - HTTP for Humans**
  Warning : Recreational use of other HTTP libraries may result in dangerous side-effects, including : security vulnerabilities, verbose code, reinventing the wheel, constantly reading documentation, depression, headaches, or even death.
  (source : http ://docs.python-requests.org/en/master/)
- **HTTPX - A next-generation HTTP client for Python**
  Same but looks better

Requesting REST API in Python

SCIENCES SORBONNE UNIVERSITÉ

→ Ideal case with web service, documentation provided, open formats, authorized access, ...

Base

```
>>> import requests
>>> result = requests.get("http://www.bing.com/search?q=http")
>>> print(result.text)
```

Json : Request does data loading

```
>>> import requests
>>> r = requests.get('https://api.github.com/events')
>>> r.json()
[{u'repository': {u'open_issues': 0, u'url': 'https://github.com/...
```

• If no documentation, we do reverse engineering : analysis of web page requests, study of mobile applications

But often...

→Less-than-ideal case : no web service, incomprehensible format, incompatible terms of use, not the information you're looking for.

What to do ?

- Data extraction from a web page
  - No guarantee of stability
  - Not necessarily authorized / general terms of use
  - Responsible behavior essential

How ?

- Make HTTP requests
- Extract data from responses

Examples in production

SCIENCES SORBONNE UNIVERSITÉ

woob : Web Out Of Browser https://woob.tech

- Interfaces for other sites
- Mainly for banks, but not only
- Professional support

Zotero https://www.zotero.org/

- Bibliographic database
- Collect metadata of scientific publications
- largely used by researchers

Examples in production

Scrapy https://scrapy.org

- Easy website browsing

Playwright https://playwright.dev/

- Automates a webbrowser

Web crawling

- websearch engine
- Archiving, tracking, ...

Crawling the web

SCIENCES SORBONNE UNIVERSITÉ

The "Classic" web

- GET request on the URL
- HTML code as answer : the page is a document
- The document is a HTML tree : easy to capture/browse

Libraries

- Requesting API : Requests / HTTPX
- Parsing HTML : BeautifulSoup

Example of HTML code

```
<!DOCTYPE html>
<html>
 <head>
  <title>
   Exemple de HTML
  </title>
 </head>
 <body>
  This is a sentence with an
    <a href="cible.html">hyperlink</a>.
  <p>
This is a paragraph without hyperlink  </p>
 </body>
</html>
```

Set up the path

SCIENCES SORBONNE UNIVERSITÉ

XPath

- Request language on XML tree
- Localization path : succession of steps to reach a target
- Localization step : information filtering and extraction

CSS

- Set up for page display
- But is an instance of tree browsing

Examples

SCIENCES SORBONNE UNIVERSITÉ

XPath

/	root node
/html/body	body node
//div	all div nodes
//h1[@class='title']	nodes of a given class
div	relative request
div[1]/p[2]	Second paragraph of first div
table[count(tbody/tr)>2]	Tables with more than two lines

XPath

- div.class#id
- and other filters

In Python - Beautiful Soup

SCIENCES SORBONNE UNIVERSITÉ

Python lib for syntax analysis (XML, HTML)

```
>>> import requests
>>> from bs4 import BeautifulSoup
>>> url = "https://fr.wikipedia.org/wiki/
    Uniform_Resource_Locator"
>>> page = requests.get(url)
>>> print(page.status_code)
  200
>>> soup = BeautifulSoup(page.text, "html.parser")
>>> soup.title
  <title>Uniform Resource Locator \u2014 Wikip\xe9dia
    </title>
>>> soup.title.string
  u'Uniform Resource Locator \u2014 Wikip\xe9dia'
>>> soup.find("h2")
  <h2>Sommaire</h2>
```

In Python - Beautiful Soup

SCIENCES SORBONNE UNIVERSITÉ

Find a node

```
>>> soup.find("div")
>>> soup.find("div", class="offer)
>>> soup.find("div", id="firstHeading")
```

Find all nodes

```
>>> soup.find_all("div")
```

Parents, enfants

```
>>> find_parents() , .find_parent()
>>> .content[0]
>>> .next_siblings , .previous_siblings
```

Crawling the web

SCIENCES SORBONNE UNIVERSITÉ

The "modern" web

- GET request on the URL
- A piece of information
- Javascript code : the page is a program
- The code executes other requests and build the page accordingly
- The content is on a virtual tree (DOM) : the tree is the result of a program execution
- More complex !

Libraries

- Selenium (when javascript : allow to simulate a browser and to interact with javascript elements)
- Scrapy (more complete and more complex tool. Recommended for big projects)

In Python - Selenium

SCIENCES SORBONNE UNIVERSITÉ

WebDriver / remote control interface

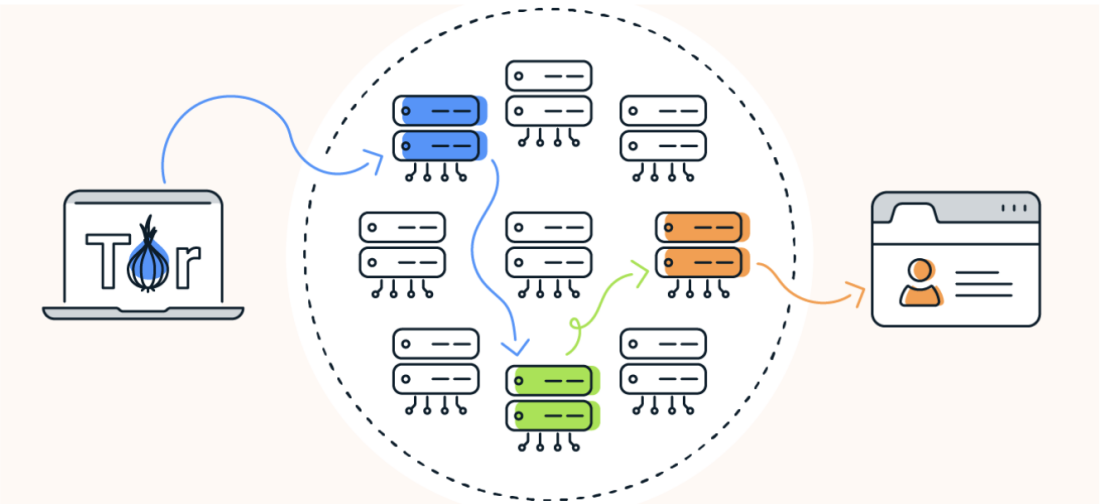
```
>>> driver = webdriver.Chrome()
>>> driver.get("https://www.selenium.dev/selenium/web/web-form.html")
>>> title = driver.title
>>> text_box = driver.find_element(by=By.NAME, value="my-text")
>>> submit_button = driver.find_element(by=By.CSS_SELECTOR, value="button")
```

In Python - Playwright

```
>>> from playwright.sync_api import sync_playwright
>>> with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page() page.goto("http://
    playwright.dev")
    print(page.title()) browser.close()
```

Scrapping on a private network

- TOR (https://www.torproject.org/fr/download/ and https://www.lobstr.io/fr/blog/comment-scraper-avec-des-tor-proxies-et-python)
  Tor (The Onion Router) is a network that anonymizes web traffic to ensure 100% private web browsing. The Tor browser masks your IP address and browsing activity by redirecting web traffic via a series of routers known as nodes.



→ **This does not prevent you from respecting the law**

Protections against scraping

- Check that the user is a human
  - Captcha
  - Two-factor authentication
  - Solution.... A human....
- Statistics
  - Rate limit : freeze when the speed of interactions is too high
  - Machine learning to detect non-human behavior
  - Solution : look like a human
- Other technics
  - Frequently updating the page structure
  - IP filtering

Conclusion

- Libraries
  - Complete navigators : Playwright, Selenium
  - Just request : Scrapy, Requests, HTTPS, BeautifulSoup
  - Private navigation : Tor
- Tools
  - Inspect the navigator
  - Tree browsing : XPath and CSS
- No formal method :
  - Understanding the website
  - Mimic human behavior
  - Responsible usage to limit legal/ethical issues

More info : https://inseefrlab.github.io/formation-webscraping/