

leboncoin.fr

lobstr.io

The purpose of this case is to test your abilities through 3 main area of competences :

- XPath
- Regular Expression (Regex)
- Requests/LXML with Python

You will find a doc named sample.html which is the source code of the following URL :

https://www.leboncoin.fr/recherche/?category=9&locations=Cassis_13260.



1. XPATH

Find the right XPath to find the very element in **sample.html**

https://www.w3schools.com/xml/xpath_intro.asp

Google Sheet (IMPORTXML)

1.a. (Example) Find the total number of “Annonces”

The screenshot shows a web browser interface with a real estate listing. The listing includes a bottle of wine for 20,90€ and an apartment for 850,000€. A DOM inspector is open, showing the following structure:

```
<div class="apn-lt">...</div>
<div class="_2Njaz _3GLp9">
  <div class="_358d0">
    <div class="_2r1q3">
      <div>
        <div class="_1uEY7">
          <p> == $0
          <span class="_2iING">138</span>
          <!-- react-text: 919 -->
          " Annonce"
          <!-- /react-text -->
          <!-- react-text: 920 -->
```

//span[@class="_2iING"]/text()

- 1.a Find XPath for title of annonce
- 1.b Find XPath for price of annonce
- 1.c Find XPath for kind of annonce (“Pro”, bold)
- 1.d Find XPath for type of annonce (“Ventes Immobilières”)
- 1.e Find XPath for city of annonce
- 1.f Find XPath for zip code of annonce
- 1.g Find XPath for date

NB : one XPath must work for at least 2 URLs, and return a list of values for every annonce



2. REGEX

Find the right Regex to find the element we are looking for in **sample.json**

<https://regex101.com/>

2.a. (Example) Find the total number of “Annonces”

```
33  
34     ],  
35     "data":{  
36         "total":138,  
37         "total_all":138,  
38         "total_pro":118,  
39         "total_private":20,  
40         "total_active":0,  
41         "total_inactive":0,  
42         "pivot":"35:1554795997000",  
43         "ads":[  
44             {
```

(?<=\\\"total\\\":)\\d+

2.b. Find the total number of private “Annonces” : “total_private”

/* ECRIRE LE REGEX ICI */

2.c. Find the total number of pro “Annonces” : “total_pro”

2.d. Find all the unique IDs of “Annonce” : “list_id”

```
306     "list_id":1598602783,
```

2.e. Find the prices : “price”

```
317     "price":[  
318         420000  
319     ],
```

2.f. Find the subject : “subject”

```
456     "subject":"Appartement 3 pièces 68 m²",
```

2.g. Find the number of rooms : “rooms”



```
501 ▼ {
502     "key": "rooms",
503     "value": "3",
504     "key_label": "Pièces",
505     "value_label": "3",
506     "generic": true
507 },
```

2.h.a Find lat

2.h.b Find lng

```
24 "lat": 43.2142,
25 "lng": 5.54296,
26 "name": "Hôtel de
```



3. REQUESTS/LXML

In this final case, you have to build a script, in Python, using **REQUESTS/LXML**, that will scrape annonces data from https://www.leboncoin.fr/recherche/?category=9&locations=Cassis_13260, from page 1 to page 4.

- write script.py able to scrape data
- save data in .csv file
- host your script on Github
- send to us :)

An already made extract is available : **output.xlsx**.



TECHNICAL ADVICE

NB: <https://www.leboncoin.fr/> is protected with *datadome*, French leading bot-mitigation services. Please, control your request speed and request attributes — headers, cookies...

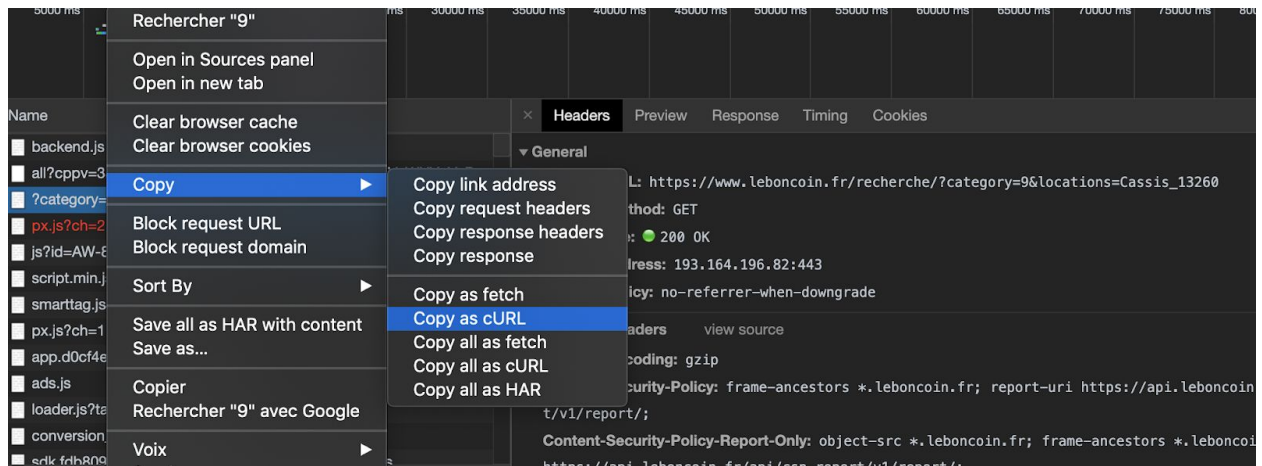
Use the **Google Chrome Search Tool**, to search the data you want to parse, and identify the very request that originated it.

The screenshot displays a web browser window with the leboncoin.fr website. The main content area shows a real estate listing for an apartment: "Appartement 3 pièces 90 m²" priced at "895 000 €". The listing includes a photo of the interior and text indicating it is for sale by a professional agent in Cassis 13260. To the right, there is an advertisement for a "Smartphone Y5 20...".

Below the browser window, the Chrome DevTools Network tab is open. The search bar at the top of the Network panel is set to "Appartement 3 pi". The list of network requests shows several JavaScript files and a JSON response. The selected request is a JSON response from the server, which contains configuration data for the application, including window properties like "window.ROC_CONFIG", "window.APP_CONFIG", "window.FLUX_STATE", and "window.REDIAL_PROPS". The response is highlighted in green in the original image.



Thereafter, use **Copy as CURL**, to get the nature of the request sent: URL, headers, cookies etc.



Endly, **convert it to `requests`** with <https://curl.trillworks.com/>:

curl command

```
VIWCGTV_o9mmBfgrGUfTA47uEVZsalW4w8Nn53zE5DqWzfl0FvO8KPYQN_w-  
9CNeGb2N82EFzRr7AMjZ1F5zIWWrlw.aBigbkmlVdTM96Zvfi-  
5hw%2C%2C0%2Ctrue%2C%2CeyJraWQiOiIyIiwiaWxnljoiSFMyNTYifQ..KkdZBUjvN  
5pD4QKkFgCqSZVCCL3ECJHkFzjxa3ioO4;  
datadome=AkalhU~mUX69SpsHaMz_aqgFP-  
_qLdQPHyixUMKnYotyloirUk_HiDEWnwxdpfiwIlaAtZKOcv5r7prX8eW7_O-qB694vkvQ  
dd3cef3caw;  
utag_main=v_id:016eb7a9cf18000330f12c342a970307900190710093c$ _sn:42$ _ss:0$  
st:1581874839663$ _pn:3%3Bexp-session$ses_id:1581872820948%3Bexp-session:  
cikneeto=date:1581873039678' --compressed
```

Examples: GET - POST - Basic Auth

Python requests

```
import requests  
  
cookies = {  
    'cikneeto_uuid': 'id:033c7e61-1fbb-40ad-ab27-eee445a3f19b',  
    'gcl_au': '1.1.1625974999.1575039388',  
    'cto_lwid': '4c84dbda-3735-454a-a5e6-e04f45801b25',  
    'xtvrn': '$562498$',  
    'xtan562498': '-undefined',  
    'xtant562498': '1',  
    'rv_rv-l3b0nco_realytics':  
}
```

Language Python

