

Tutorial #1: Web Scraping treaties from the IEA Database

Alice Soldà

December 1, 2020

The [IEA Database](#)¹ currently contains over 1,300 MEAs, over 2,200 BEAs, 250 other environmental agreements, and over 90,000 individual country “membership actions” that are constantly being revised and updated.

Retrieving these texts manually can quickly become a hassle.

Luckily for us, the database has been modified in November 2020 in a way that enables to easily collect any treaty text included in the database using a simple Python code.

To do so, you will need to download and install [Python](#) (this tutorial was made using Python 3.7.3).

¹(c) Ronald B. Mitchell and the IEA Database Project, 2002-2020

1 Retrieving the relevant IDs

The first thing we need to do is to identify the IDs of our treaties of interest. Each treaty is given an ID (called Mitch ID) that links it to various sets of information (metadata, membership data, etc.). What matters to us here, is that a particular treaty text is linked to a unique treaty ID.

Identifying treaties IDs can be easily done by downloading the list of agreements in CSV format from the [IEADB Agreement List](#) and using filters on excel. You can already use some filters directly on the IEADB website. The treaties IDs are stored in the first column of the CSV file.

IEA# (click for add'l info)																
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	IEA# (click for add'l info)	Inclusion	Signature Year	Signature Date	Date IEA entered into force	Treaty Name	Treaty Text**	Members	Agreement Title	Sequence Number	Data	Articles, etc.	Secretariat	UNEP Info		
2	4964	MEA	2010	01/01/2010	01/01/2010	Cooperation Treaty	Text**	Members	Agreement Te Vaka	100.000	Data	Lit	Te Vaka Moana			
3	4972	MEA	2010	02/02/2010	14/02/2017	Agreement o Treaty	Text**	Members	Agreement Prespa Park	100.000	Data	Lit	Prespa Park Secretariat			
4	4614	MEA	2010	25/03/2010	23/06/2010	Amendment: Treaty	Text**	Members	Amendment CITES	100.180	Data	Lit	CITES Secretariat			
5	4615	MEA	2010	26/03/2010	01/08/2011	Amendment: Treaty	Text**	Members	Amendment MARPOL	204.040	Data	Lit	International UNEP Info			
6	4695	MEA	2010	26/03/2010	01/08/2011	Amendment: Treaty	Text**	Members	Amendment MARPOL	203.540	Data	Lit	International Maritime Organization (IMO)			
7	4612	MEA	2010	31/03/2010	01/01/2599	Protocol for t Treaty	Text**	Members	Protocol Eastern Afric	103.000	Data	Lit	UNEP Region UNEP Info			
8	4613	MEA	2010	31/03/2010	01/01/2599	Amended Co Treaty	Text**	Members	Agreement Eastern Afric	200.000	Data	Lit	UNEP Regional Coordinating Unit for the Eastern Africa			
9	4611	MEA	2010	30/04/2010	01/01/2599	Protocol to t Treaty	Text**	Members	Protocol Liability Com	101.000	Data	Lit	International UNEP Info			
10	4957	MEA	2010	14/05/2010	01/01/2599	Agreement o Treaty	Text**	Members	Agreement Nile River Ba	1000.000	Data	Lit	Nile Basin Initiative			
11	8025	MEA	2010	21/05/2010	19/05/2011	Protocol on amendments t	Members		Amendment Belarus-Kaza	201.000	Data	Lit	Secretariat not yet identified			
12	8026	MEA	2010	21/05/2010	19/05/2011	Protocol on amendments t	Members		Amendment Belarus-Kaza	101.000	Data	Lit	Secretariat not yet identified			
13	4939	MEA	2010	01/06/2010	27/11/2015	Protocol On t Treaty	Text**	Members	Protocol Sava River Ba	103.000	Data	Lit	International Sava River Basin Commission			
14	4847	MEA	2010	04/06/2010	04/06/2010	Agreement b Treaty	Text**	Members	Agreement Lithuania-Est	200.000	Data	Lit	No secretariat established			
15	4899	MEA	2010	08/06/2010	01/01/2011	Amendment: Treaty	Text**	Members	Amendment Rhine Polluti	1100.020	Data	Lit	Secretariat for the Convention relative À la collecte, a			
16	4691	MEA	2010	25/06/2010	11/01/2011	Amendment: Treaty	Text**	Members	Amendment International	300.650	Data	Lit	International Whaling Commission			
17	4924	MEA	2010	11/09/2010	01/01/2899	Amendment: Treaty	Text**	Members	Amendment Micronesia F	200.020	Data	Lit	Parties to the Nauru Agreement			
18	5033	MEA	2010	11/09/2010	11/09/2010	2010 Amendi Treaty	Text**	Members	Amendment Micronesia F	103.010	Data	Lit	Parties to the Nauru Agreement			
19	4637	MEA	2010	01/10/2010	01/01/2014	Amendment: Treaty	Text**	Members	Amendment MARPOL	203.550	Data	Lit	International Maritime Organization (IMO)			
20	4705	MEA	2010	01/10/2010	01/02/2012	Amendment: Treaty	Text**	Members	Amendment MARPOL	204.050	Data	Lit	International UNEP Info			
21	4636	MEA	2010	15/10/2010	05/03/2018	Nagoya-Kual: Treaty	Text**	Members	Protocol Biological Div	102.000	Data	Lit	Secretariat: fc UNEP Info			
22	8024	MEA	2010	21/10/2010	23/05/2011	Agreement b Treaty	Text**	Members	Agreement Not Yet Assigned				Secretariat not yet identified			
23	4638	MEA	2010	29/10/2010	12/10/2014	Nagoya Protc Treaty	Text**	Members	Protocol Biological Div	103.000	Data	Lit	Secretariat: fc UNEP Info			
24	5081	MEA	2010	12/11/2010	01/01/2599	Amendment Treaty	Text**	Members	Amendment Conservator	600.020	Data	Lit	ACCOBAMS Secretariat			
25	8511	MEA	2010	12/11/2010	12/04/2011	Amendment Treaty	Text**	Members	Amendment Conservator	600.030	Data	Lit	ACCOBAMS Secretariat			
26	4898	MEA	2011	28/01/2011	01/01/2599	Amendment Treaty	Text**	Members	Amendment Association c	100.010	Data	Lit	Association of Caribbean States			
27	4966	MEA	2011	29/04/2011	27/10/2012	Amendment: Treaty	Text**	Members	Amendment Persistent Or	100.030	Data	Lit	UNEP Chemi UNEP Info			
28	8023	MEA	2011	19/05/2011	26/12/2011	Protocol on amendments t	Members		Amendment CIS Environmental Agree		Data	Lit	Secretariat not yet identified			
29	4872	MEA	2011	27/05/2011	21/10/2013	Protocol on S Treaty	Text**	Members	Protocol Carpathian Si	102.000	Data	Lit	Interim Secretariat of the Framework Convention on t			

2 Retrieving the relevant texts

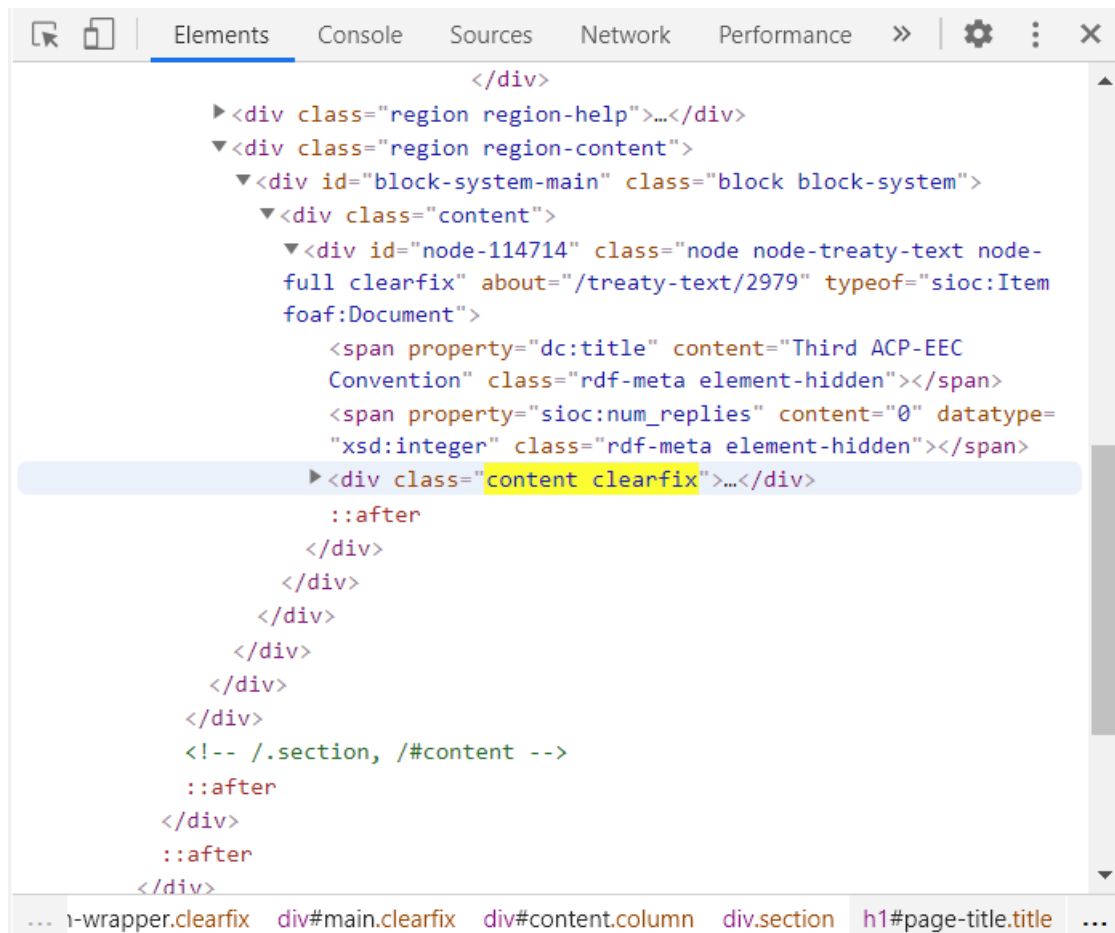
2.1 Inspecting the website

The second thing we need to do is to figure out where our treaties texts are stored in the IEADB website. The updated version of the database makes things easier for us. The URLs we are looking for are now all of the form [https://iea.uoregon.edu/treaty-text/\[treaty ID\]](https://iea.uoregon.edu/treaty-text/[treaty ID]). This will be helpful to code our Python program.

In addition, we need to tell our python program what we need from the page. More specifically, we need to figure out what are the HTML tags that contain the data we want to retrieve. Simply put, there is a lot of code on a website page and we want to find the relevant pieces of code that contains our data. You can learn more about HTML tags [here](#).

To save time, I have already identified the tag (`<div>`) and the class (`"content clearfix"`) that contains our text. Importantly, the class of this tag is unique within the page and is

the same for all treaties. If you want to know more about how to identify relevant HTML tags, you can check out Julia Kho's tutorial on web scraping with Python [here](#).



2.2 Python Code

Now that we have the CSV file with our IDs and we identified where to locate the treaties texts, the last thing we need before diving into the code is a folder in which we will store the treaties texts. For this tutorial, I have created a folder named "treaties" on my desktop.

All set? Now open your Python IDLE and let the fun begin!

We start by importing the relevant libraries.

```
import requests
import urllib.request
import re
from bs4 import BeautifulSoup
```

We use the [requests](#) library to tell our python program where to look for our data and the [re](#) library to clean the text what we have retrieved from its HTML tags. [Beautiful Soup](#) is a Python library for pulling information from web pages.

Next, we create a list that contains our treaties IDs. However, we cannot just copy-paste the first column from our CSV file into our python code because each element of a Python list needs to be separated by a comma. Luckily, we can easily do this in excel with the function CONCAT. If you need some help with this step, check [here](#). We can now copy-paste our concatenated IDs into our list of treaties.

```
list_treaty = [  
3059,  
8501,  
3100,  
5006]
```

Let's say we want to scrap the text of the first treaty in the list. We need to set the URL to the web page where the text is and access the page with our requests library.

```
url_string='https://iea.uoregon.edu/treaty-text/'+str(list_treaty[0])  
website_url=requests.get(url_string)
```

As mentioned earlier, the treaties texts URL are all of the form: `https://iea.uoregon.edu/treaty-text/[treaty Mitch ID]`. Because we want the first treaty in the list, we tell python to replace `[treaty Mitch ID]` by `list_text_id[0]` (i.e. the first element of our list).

To check that the URL is correct, we can print the URL in the Python console.

```
print(url_string)
```

If the request is successful, you should see the following output.

```
>>> https://iea.uoregon.edu/treaty-text/3059
```

Next, we parse the HTML with BeautifulSoup so that we can work with a nicer, nested BeautifulSoup data structure.

```
soup = BeautifulSoup(website_url.text, 'lxml')
```

We use the method `.find` to locate our text based on the HTML tag and class we identified earlier.

```
html_treaty = soup.find('div', {'class': 'content clearfix'})  
text_treaty = str(html_treaty)
```

Now if you print `text_treaty`, you will see that the current text is full of HTML tags, which does not look very nice.

```

<div class="content clearfix">|
<div class="field field-name-body field-type-text-with-summary
field-label-hidden"><div class="field-items"><div class="field-item
even" property="content:encoded"><p>FOURTH ACP-EEC CONVENTION</p>
<p>Source: <a href="http://eur-
lex.europa.eu/Notice.do?val=172163:cs&lang=en&list=172163:cs
,&pos=1&page=1&nbl=1&pgs=10&hwords=&checktex
te=checkbox&visu=#texte">http://eur-
lex.europa.eu/Notice.do?val=172163:cs&lang=en&list=172163:cs
,...</a></p>
<p>PART ONE</p>
<p>General provisions of ACP-EEC co-operation</p>
<p>CHAPTER 1</p>

```

The next step is to get rid of these tags using the re library.

```

cleanr = re.compile('<.*?>')
clean_text = re.sub(cleanr, '', text_treaty)

```

Finally, we create the word document in which we are going to save our treaty text. I name the file text_ + the treaty ID (which, as before, is given by the first element of our list of treaties) and I tell the program that I want the word document inside the folder “treaties” I have created at the beginning of this section. Of course, you need to replace “C:/Users/alice/Desktop/treaties/” by the path to your own folder. If you need help with this step, check [here](#).

```

filename = "text_"+str(list_text_id[0])
location = "C:/Users/alice/Desktop/treaties/" + filename + ".doc"

```

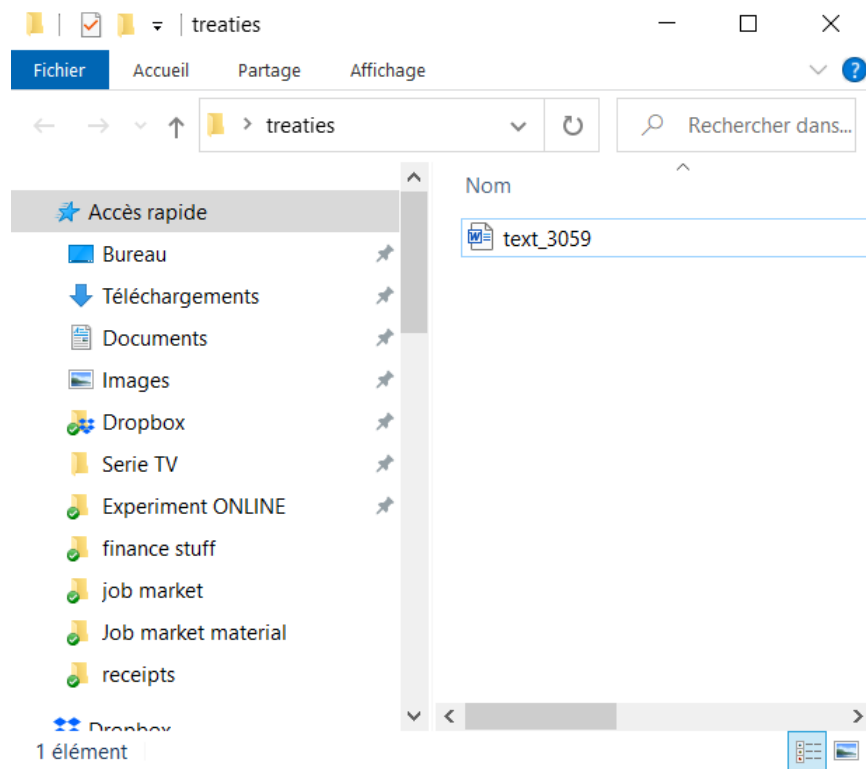
We then tell the program to open the word document and write our treaty text in it.

```

file = open(location, "w", encoding="utf-8")
file.write(clean_text)
file.close()

```

Go to your “treaties” folder. You now have a word document in there.



Open the word document. If word asks you to choose the encoding to use for this document choose "Unicode (UTF-8)". You just webscraped your first treaty text. Well done!

Note that if a text you are looking for is not available on the IAE Database, your word document will contain the following:

```
Apologies but that page does not exist.
We migrated the IEADB website to a new Drupal platform in September 2016
and a few links may have gotten broken in the process. You may be able
to find the old link by changing the URL from iea.uoregon.edu to iea-
archive.uoregon.edu
e.g., change
http://iea.uoregon.edu/page.php?file=home.htm&query=static to
http://iea-archive.uoregon.edu/page.php?file=home.htm&query=static
We would greatly appreciate it if you cut and pasted the URL that is not
working and sent it to the PI at rmitchel@uoregon.edu so I can repair it.
Thanks,
Ronald Mitchell
```

Now that we understand how to download a treaty text, let's try downloading the entire list of treaties texts. The code below contains the full code for web scraping your entire list of treaties texts. You can also download the .py file directly from [here](#).

```

import requests
import urllib.request
import re
from bs4 import BeautifulSoup

list_treaty = [

    #add all treaties IDs in there

]

for treaty in list_treaty:
    url_string='https://iea.uoregon.edu/treaty-text/'+str(treaty)

    website_url=requests.get(url_string)

    soup = BeautifulSoup(website_url.text,'lxml')
    html_treaty = soup.find('div', {'class':'content clearfix'})
    text_treaty = str(html_treaty)

    filename = "text_"+str(treaty)
    location = "C:/Users/alice/Desktop/treaties/" + filename + ".doc"
    file = open(location, "w", encoding="utf-8")

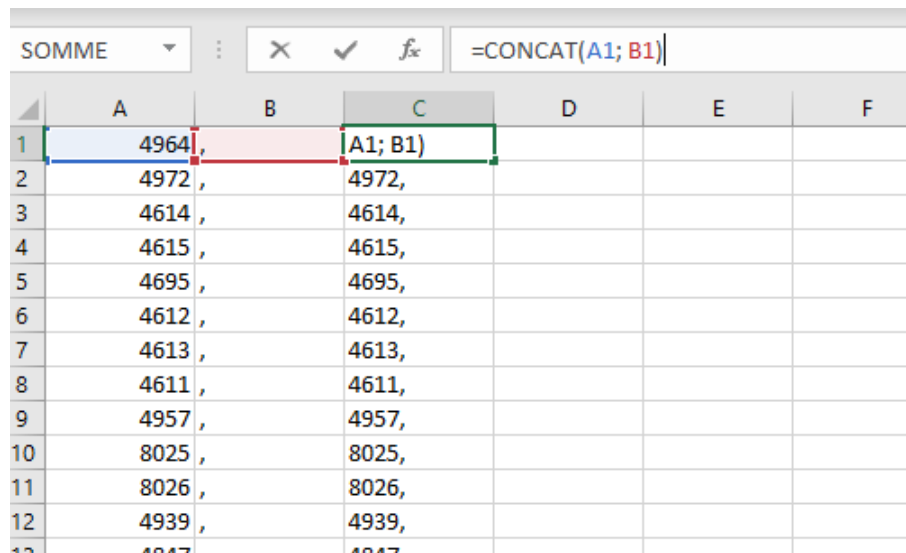
    cleanr = re.compile('<.*?>')
    clean_text = re.sub(cleanr, '', text_treaty)
    file.write(clean_text)
    file.close()

```

3 Extra Help

3.1 CONCAT in Excel

What we need is a column with the treaties IDs and one column filled with commas. Then we use the CONCAT function to add the comma to each ID. (Note that if you are using the ENGLISH version of excel, you need to use “,” instead of “;” in the CONCAT function).



	A	B	C	D	E	F
1	4964	,	A1; B1			
2	4972	,	4972,			
3	4614	,	4614,			
4	4615	,	4615,			
5	4695	,	4695,			
6	4612	,	4612,			
7	4613	,	4613,			
8	4611	,	4611,			
9	4957	,	4957,			
10	8025	,	8025,			
11	8026	,	8026,			
12	4939	,	4939,			

We can now copy-paste the content from the last column into our list of treaties in our python code!