# Web scraping for chemical compounds

Dr Krishna K. Govender

- Before we begin, we need to have the appropriate conda environment created along with the necessary tools needed.

- Launch the anaconda_prompt

- If you are in Linux or MacOS you can open a terminal and you should see (base)… on the screen. If so, you are already in the conda prompt and can follow the instructions below.

- In the anaconda prompt type the following:

```
1. conda create -n webd -c bioconda -y
2. conda activate webd
3. conda install pip spyder -y
4. pip install requests pubchempy pandas
```

- Before we begin writing our code, we need to have an idea of what we plan to scrap for. We will be trying to pull information from [Search the Databases | Dr. Duke's Phytochemical and Ethnobotanical Databases]

- Go back to the anaconda prompt:

```
1. spyder
2. import requests as req
3. import os
4. Click on the play button
5. File > Save as… > Choose where to save the file > pythochem_download.py
```

- Create a folder called **bioact** where the information we are going to download can be stored and if the folder already exists then don't recreate it:

```
1. # Create a folder for the bioact
2. try:
3.     os.mkdir("bioact")
4. except FileExistsError:
5.     pass
```

- Now define the base URL for the webpage:

```
1. base = 'https://phytochem.nal.usda.gov'
```

- Go to the website and click on the (-)-Bactericide entry, then under the Download these results as section right click on CSV and select copy link and then paste that information into spyder.
  Go back in your browser and do the same thing for the (-)-Chronotropic entry and paste the information after the previous one in spyder:

```
1. https://phytochem.nal.usda.gov/biological-activities-chemicals-csv-
export/4/all?page&_format=csv
2. https://phytochem.nal.usda.gov/biological-activities-chemicals-csv-
export/5/all?page&_format=csv
```

- From the two links you should notice that the only thing different between the two links is the entry number (4 and 5 in this case).

- We are going to make the second lines we copied into a comments by adding a # in front of each.

- Let's try to download the first file:

```
1. res = req.get('https://phytochem.nal.usda.gov/biological-activities-chemicals-csv-
export/4/all?page&_format=csv')
```

- The above has only sent a request to the webpage. Now we need to tell python that we want to have this downloaded:

```
1. csv = open("bioact/4.csv","wb")
2. csv.write(res.content)
3. csv.close()
4. print("File 4 downloaded")
```

- Now change up the code to download both entries by making use of a loop:

```
1. for bio in range(4, 6):
2.     compurl = '/biological-activities-chemicals-csv-export/'+str(bio)+'/all?page&_format=csv'
3.     # Send a GET request to the webpage
4.     res = req.get(base+compurl)
5.     csv = open("bioact/"+str(bio)+".csv","wb")
6.     csv.write(res.content)
7.     csv.close()
8.     print("Downloading "+str(bio))
```

- Running the code again will redownload the files that have already been downloaded so how can we ensure that it only downloads files that have not been downloaded:

```
1. # Send a GET request to the webpage
2. if os.path.isfile("bioact/"+str(bio)+".csv"):
3.     print(str(bio)+" already downloaded")
4. else:
5.     res = req.get(base+compurl)
6.     csv = open("bioact/"+str(bio)+".csv","wb")
7.     csv.write(res.content)
8.     csv.close()
9.     print("Downloading "+str(bio))
```

- Now change up the code to download entries 1 – 5 (including entry 5):

```
1. for bio in range(1, 6):
```

- Have a look at the files which should be in the same place as you pythochem.py file inside a folder called **bioact**. You should notice file 1 has nothing. As we saw when we first went to the webpage this entry did not contain any chemical information in the CSV file, so it is as we would expect.

- Let's try to download the compounds found in the CSV files from Pubchem.

- First, we need to import the appropriate libraries:

```
1. import pubchempy as pcp
2. import pandas as pd
3. import glob
4. from pandas.errors import EmptyDataError
```

- Create a folder to store the SDF entries:

```
try:
    os.mkdir("SDFS")
except FileExistsError:
    pass
```

- Find all CSV files of interest and store in an array:

```
csv_files = []
csv_files += (glob.glob('bioact\*.csv'))
print(csv_files)
```

- If we happy with what we see from the array that we created, we can remove the print statement.

- Read the data in the CSV files and store in temp dataframe:

```
for files in csv_files:
    temp_df = pd.read_csv(files, sep=',')
```

- The above will give you an error since the very first CSV file has no information, so we need to consider that there are going to be files with no data.

```
for files in csv_files:
    try:
        temp_df = pd.read_csv(files, sep=',')
    except EmptyDataError:
        print(files+" is an empty csv file")
```

- Append all the CSV information into an array

```
lig = []
for files in csv_files:
    try:
        temp_df = pd.read_csv(files, sep=',')
        lig.append(temp_df)
    except EmptyDataError:
        print(files+" is an empty csv file")
print(lig)
```

- Combine all the data into a single dataframe:

```
df = pd.concat(lig, axis=0, ignore_index=True)
```

- Iterate through the names stored in the dataframe and get their compound ID:

```
for name in df['Chemical Name']:
      compounds = pcp.get_compounds(name, 'name')
      for compound in compounds:
                cid = compound.cid
                print(cid)
```

- Download the SDF files for the compounds and store the number of SDFs that are being downloaded into an array:

```
pcp.download('SDF', 'SDFS/'+str(cid)+'.sdf', cid, 'cid')
print("Downloading "+str(cid))
tot.append(cid)
```

- For tot to be able to take in entries you must define if before the initial for loop that goes through chemical names:

```
tot = []
for name in df['Chemical Name']:
```

- Print a statement indicating how many SDF files were downloaded before letting the user know the program is DONE:

```
1. print("Total number of SDFs downloaded: "+str(len(tot)))
2. print('DONE')
```

- Running the code will say that 69 entries where downloaded, but checking the **SDFS** folder shows only 68 entries and this is probably because one of the entries was a duplicate and got overwritten, but the tot array still added this as an entry because off the way the code was written.

- To circumvent this one could have something written to check if an SDF exists and if it does do not redownload the entry. In addition, you can count the number of already downloaded SDF files in one array and those that are being downloaded in a second array then combine the two into a new array that ignores the duplicates.

- Something else that can be done is to check for those compounds that don't have hits on PubChem and make note of them so that one can look for them elsewhere.

- The SDF files downloaded are 2D structures so one would need to use an appropriate application (Openbabel, Schrödinger Maestro, etc.) to convert the structures to 3D before using for something like molecular docking.

- Git repo for the code:
  https://github.com/superbash-maker/CSS_2025.git