SI 206 Discussion 9

HTML and Beautiful Soup

HTTP Requests

- Library for making HTTP (Hypertext Transfer Protocol) requests to interact with web services, websites, and APIs

```python
import requests

url = "https://www.google.com/"

r = requests.get(url)

print(r)
print(r.text)
```

```
<Response [200]>
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lan
, including webpages, images, videos and more. Google has many special fea
ame="description"><meta content="noodp" name="robots"><meta content="text/
nt="/images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop
-w07ce2OA">(function(){var _g={kEI:'QNU2ZdzmNpjS2roP2M-y4As',kEXPI:'0,1816
7528,16112,28687,22430,1362,12314,17585,4998,17075,38444,2872,2891,3926,21
894,29703,1457,22610,6627,7596,1,42154,2,16395,342,3533,19491,5679,1021,31
14491,873,19633,7,1922,9779,42459,20199,20136,14,82,7651,8863,3692,109,241
,12089,1632,2173,6669,868,3785,949,3692,8565,7769,146,21746,5203198,12,692
5,2,40,7,16,6,9,8,9,23940932,4044106,14298,2374,39458,1446,1763,1216,3,210
3,4636,2945,5463,2093,787,1597,26,3525,5494,943,19,216,241,1725,3780,7351,
2490,419,1104,1484,149,1243,3,3363,2174,365,1718,2,3263,2811,9,217,3590,56
,2116,4,498,574,1594,85,614,4,872,438,534,894,1,14,762,9,217,1328,750,7,1,
,706,233,46,1095,35,707,828,42,28,3,2,537,406,114,127,795,206,128,2,1021,6
40,298,426,106,109,1150,78,11,1,3,4,2,2,2,545,41,539,2,497,131,773,206,429
7,477,227,29,337,86,714,1,437,5,243,1,3,27,136,300,7,108,192,4,193,238,47,
449};(function(){var a;(null==(a=window.google)?0:a.stvsc)?google.kEI=_g.k
sn='webhp';google.kHL='en';})();(function(){
var h=this||self;function l(){return void 0!==window.google&&void 0!==wind
null};var m,n=[];function p(a){for(var b;a&&(!a.getAttribute||!(b=a.getAtt
for(var b=null;a&&(!a.getAttribute||!(b=a.getAttribute("leid")));)a=a.pare
==window.location.protocol&&(google.ml&&google.ml(Error("a"),!1,{src:a,glm
function t(a,b,c,d,k){var e="";-1===b.search("&ei=")&&(e="&ei="+p(d),-1===
===b.search("&cshid=")&&"slh"!==a,f=[];f.push(["zx",Date.now().toString()]
.push(["opi",c.toString()]);for(c=0;c<f.length;c++){if(0===c||0<c)d+="&";d
="+String(a)+"&cad="+(b+e+d)};m=google.kEI;google.getEI=p;google.getLEI=q;
c,d,k,e){e=void 0===e?l:e;c||(c=t(a,b,e,d,k));if(c=r(c){a=new Image;var g
elete n[g]};a.src=c}};google.logUrl=function(a,b){b=void 0===b?l:b;return
=[];google.x=function(a,b){if(a)var c=a.id;else{do c=Math.random();while(g
(a){google.sy.push(a)};google.lm=[];google.plm=function(a){google.lm.push.
c){google.lq.push([[a],b,c])};google.loadAll=function(a,b){google.lq.push(
e.fce=function(a,b,c,e){d.push([a,b,c,e])};google.qce=d;}).call(this);goog
document.documentElement.addEventListener("submit",function(b){var a;if(a=
=c||"q"===c&&!a.elements.q.value?!0:!1}else a=!1;a&&(b.preventDefault(),b.
istener("click",function(b){var a;a:{for(a=b.target;a&&a!==document.docume
getAttribute("data-nohref");break a}a=!1}a&&b.preventDefault()},!0);}).cal
```

BeautifulSoup for scraping

```
import requests
from bs4 import BeautifulSoup

url = "https://www.google.com/"

response = requests.get(url)

if response.status_code == 200:
   html = response.text
else:
   print("Failed to retrieve the web
page.")

soup = BeautifulSoup(html, 'html.parser')
```

Scraping: extracting data from the structured data (e.g. HTML) of web pages.

- Import BeautifulSoup
- Load your raw data using requests or other methods
- Create a BeautifulSoup object to begin parsing your data!

Parsing Your Soup

```
soup =
BeautifulSoup(html,
'html.parser')


print(soup.find('a'))
```

Return first match

```
<a class="gb1"
href="https://www.google.com/imghp?hl=en&a
mp;tab=wi">Images</a>
```

```
soup =
BeautifulSoup(html,
'html.parser')


print(soup.find_all('a'))
```

Return list of all matches

```
[<a class="gb1"
href="https://www.google.com/imghp?hl=en&amp;ta
b=wi">Images</a>, <a class="gb1"
href="https://maps.google.com/maps?hl=en&amp;ta
b=wl">Maps</a>, <a class="gb1"
href="https://play.google.com/?hl=en&amp;tab=w8
">Play</a>, <a class="gb1"
href="https://www.youtube.com/?tab=w1">YouTube<
/a>, …]
```

Working with Tag Objects

```
soup = BeautifulSoup(html,
'html.parser')
first_tag = soup.find('a')


print(first_tag.attrs)
print(first_tag.get('class'))
```

```
<a class="gb1"
href="https://www.google.c
om/imghp?hl=en&amp;tab=wi"
>Images</a>
```

```
{'class': ['gb1'], 'href':
'https://www.google.com/imghp?h
l=en&tab=wi'}
```

```
['gb1']
```

N.B.: You can use `find()` and `find_all()` on a tag object to find children tags

More on `find()` and `find_all()`

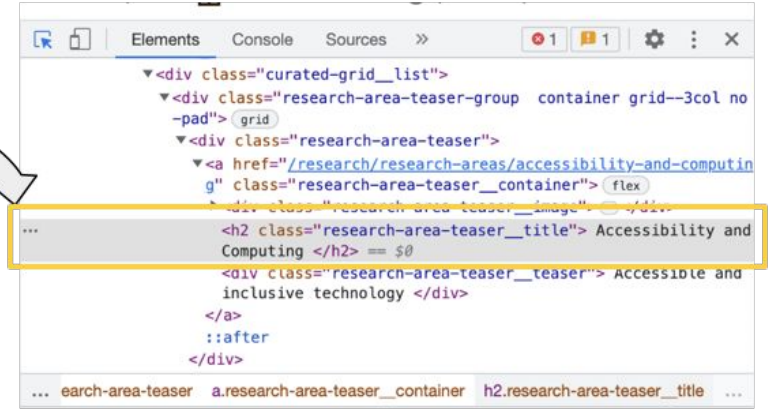
- You can pass attributes and their values as arguments to your function to run a more specific search

E.g.:

- `soup.find_all('a', href='https://www.google.com')`
  - All tags that link to Google
- `soup.find_all('div', class_='myClass')`
  - All divs with the class ‘myClass’
  - Because ‘class’ is a reserved keyword in Python, use ‘**class_**’
- `soup.find_all('div', class_='myClass myClass2 myClass3')`
  - Multiple classes are separated by a space; these are not one class
- N.B.: When trying to decide how to grab a certain tag, remember that a class can be assigned to multiple tags while ids are unique

Getting Info Using DevTools - `find()`

Getting Info Using DevTools - `find_all()`




```
# Get Get all tags of a certain type from the soup
research_areas = soup.find_all("h2", class_="research-area-teaser__title")
research_areas_list = []
for tag in research_areas:
    # Get info from tag
    research_areas_list.append(tag.text)
```

Assignment: Scraping Wikipedia

We will use BeautifulSoup to scrape data from:
https://en.wikipedia.org/wiki/University_of_Michigan

- **Task 1:** Create a BeautifulSoup object
- **Task 2:** get the link (source) for the University of Michigan emblem

Assignment: Scraping Wikipedia

- **Task 3:** Get the details of the table that has the founding year of all schools/colleges at the University of Michigan and organize the information into key-value pairs in a dictionary.Be sure to convert the founding year to an integer.

Output:

```
{'Literature, Science, and the Arts': 1841, 'Medicine': 1850, 'Engineering': 1854, 'Law': 1859, 'Dentistry': 1875, 'Pharmacy': 1876, 'Music, Theatre & Dance': 1880, 'Nursing': 1893, 'Architecture & Urban Planning': 1906, 'Graduate Studies': 1912, 'Government': 1914, 'Education': 1921, 'Business': 1924, 'Environment and Sustainability': 1927, 'Public Health': 1941, 'Social Work': 1951, 'Information': 1969, 'Art & Design': 1974, 'Kinesiology': 1984}
```

College/school	Year founded[126]
Literature, Science, and the Arts	1841
Medicine	1850
Engineering	1854
Law	1859
Dentistry	1875
Pharmacy	1876
Music, Theatre & Dance	1880
Nursing	1893
Architecture & Urban Planning	1906
Graduate Studies	1912
Government	1914
Education	1921
Business	1924
Environment and Sustainability	1927
Public Health	1941
Social Work	1951
Information	1969
Art & Design	1974
Kinesiology	1984

Assignment: Scraping Wikipedia

- **Task 4:** Sort the dictionary by founding year to see what the 3 newest programs are at the University of Michigan.

  Output:

```
Three newest programs at the University of Michigan:
Kinesiology: 1984
Art & Design: 1974
Information: 1969
```