SECTION 15: WEB SCRAPING WITH PYTHON - 1 hour 40 minutes, 9 parts

4/9 Python Web Scraping - Grabbing a Class

Syntax	Match Results
soup.select('div')	All elements with 'div' tag
soup.select('#some_id')	Elements containing id='some_id'
soup.select('.some_class')	Elements containing class = 'some_class'
soup.select('div span')	Any elements named span within a div element.
soup.select('div > span')	Any elements named span **directly** within a div element, with nothing in between.

- **grabbing all the elements associated with a class in a website**
  - -> beautiful soup for web scraping
  - -> we want to know what elements to grab associated with the parts of a website
  - -> everything before the soup.select() method is the same regardless of the website
  - -> he's going through a table for CSS syntax

- **the table**
  - -> the first row grabs everything with a particular tag
  - -> the second grabs an element with an ID
  - -> classes do the same thing
  - -> to look for elements in elements, it's div span for example
    - -> look for elements named span within a div
    - **-> to look directly within something, use a > symbol**
    - -> > <- this returns elements which are a span within a div in this example
  - -> the arguments for these follow CSS

- **in the code**
  - -> he's picked a Wikipedia article
  - -> this is for Grace Hoper
  - -> we are grabbing all the strings in the table of contents
  - -> inspecting all of the elements on the page and grabbing a class
  - **-> he inspects an element on a webpage which we want to 'grab'**
  - -> we have a series of list elements for this
  - -> there are classes used for some of the inspected elements in HTML
  - -> he's used requests.get in order to import the text which we want
  - -> set it equal to a variable
  - -> and then made a soup out of it
  - -> he has then used soup.select to pass in the name of the class for the element on the expected webpage which we are targeting
  - -> this is the Python -> we are scraping the element from the webpage into it
  - -> the class call on the webpage
  - -> the type of the item is a specialised beautiful

```
In [26]: soup.select('title')[0].getText()
Out[26]: 'Example Domain'

In [27]: site_paragraphs = soup.select("p")

In [31]: site_paragraphs[0].getText()
Out[31]: 'This domain is for use in illustrative examples in documents. You may use this
         \n    domain in literature without prior coordination or asking for permissio
         n.'

In [ ]: res = requests.get('https://en.wikipedia.org/wiki/Grace_Hopper')

In [33]: soup = bs4.BeautifulSoup(res.text,"lxml")

In [34]: soup
P identifiers","Wikipedia articles with GND identifiers","Wikipedia articles w
ith ISNI identifiers","Wikipedia articles with LCCN identifiers","Wikipedia ar
ticles with MGP identifiers","Wikipedia articles with NKC identifiers","Wikipe
dia articles with NLA identifiers","Wikipedia articles with NLI identifier
s","Wikipedia articles with NLP identifiers","Wikipedia articles with NTA iden

In [35]: # soup

In [38]: type(soup.select('.toctext')[0])
Out[38]: bs4.element.Tag

In [39]: first_item = soup.select('.toctext')[0]

In [42]: for item in soup.select('.toctext'):
             print(item.text)

Early life and education
Career
World War II
UNIVAC
COBOL
Standards
Retirement
```

soup item tag
- **-> if you tab in an ipynb file, you can see the different possible methods <- we also have documentation for this**
- -> next is grabbing an image and saving it to the computer