FoxMIS CONNECT | INNOVATE

Fox School of Business
TEMPLE UNIVERSITY

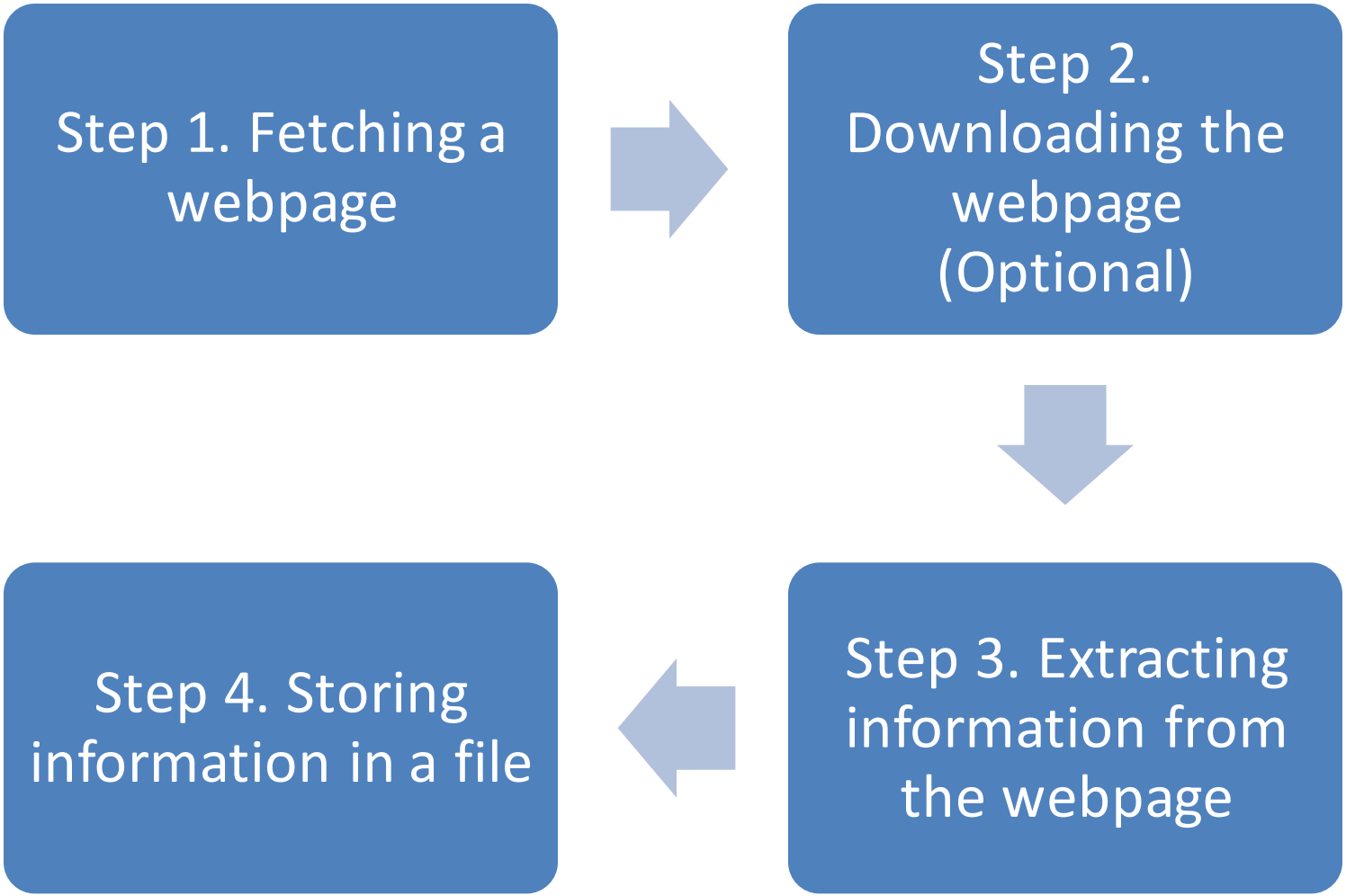
Python and Web Data Extraction: *Introduction*

Alvin Zuyin Zheng
zheng@temple.edu
http://community.mis.temple.edu/zuyinzheng/

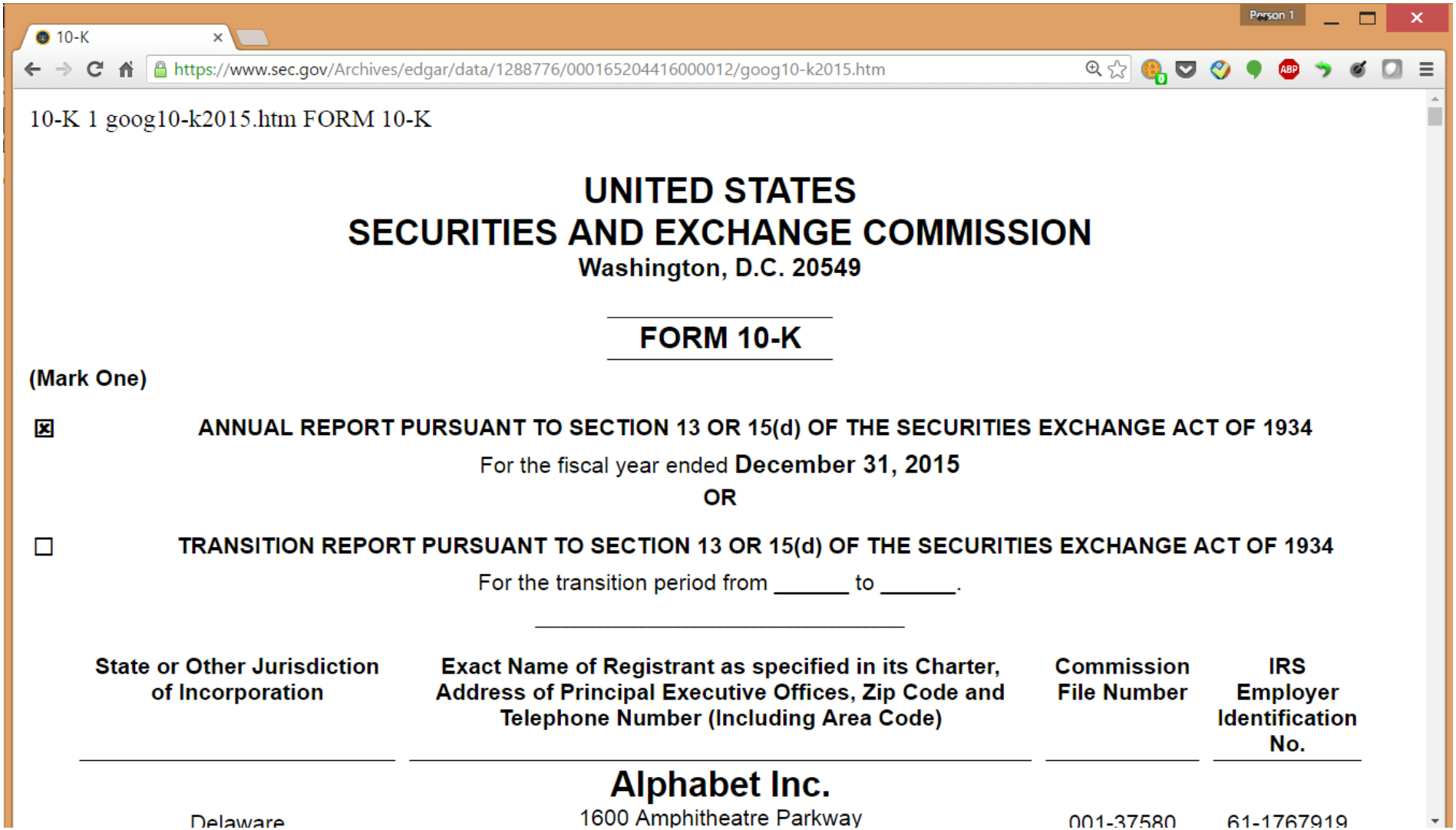
Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Download the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

Web scraping typically consist of



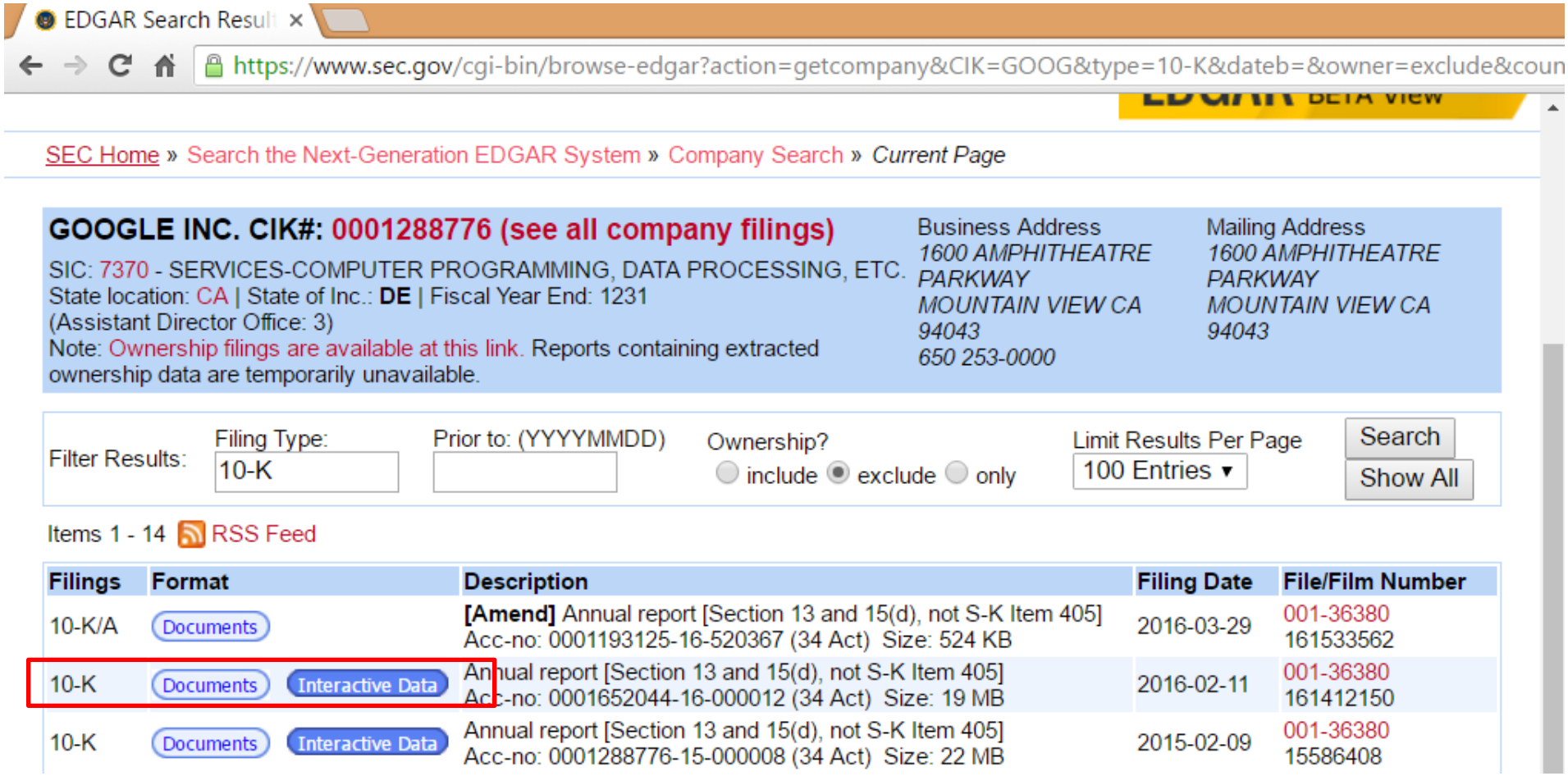
Example: 10-K



URL:

https://www.sec.gov/Archives/edgar/data/1288776/000165204416000012/goog10-k2015.htm

Example: Table with Links



URL:
https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=GOOG&type=10-K&dateb=&owner=exclude&count=100

https://www.sec.gov/edgar/searchedgar/companysearch.html

Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Downloading the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

Fetching a Webpage

- Use the `urllib2` package to open a webpage
  - Do not need to install manually

```
>>> import urllib2
>>> urlLink = "https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=GOOG&type=10-K&dateb=&owner=exclude&count=100"
>>> pageRequest = urllib2.Request(urlLink)
>>> pageOpen = urllib2.urlopen(pageRequest)
>>> pageRead = pageOpen.read()
>>>
```

Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Downloading the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

Downloading a Webpage

- We often want to download the webpages because
  - We want to limit web requests
  - Websites may change over time
  - We want to replicate research

```
>>> os.chdir
('/Users/alvinzuyinzheng/Dropbox/PythonWorkshop/scripts
/')  #Change your working directory

>>> htmlname = "goog10-k2015.htm"

>>> htmlfile = open(htmlname, "wb")

>>> htmlfile.write(pageRead)

>>> htmlfile.close()
```

Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Download the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

What is HTML

- When performing web data extraction, we deal with HTML files
  - Hyper Text Markup Language

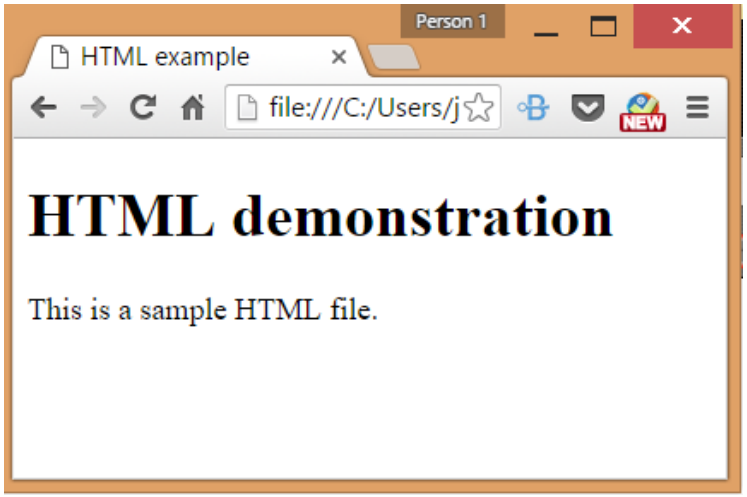
- HTML specifies a set of ***tags*** that identify structure and content type of webpages
  - Tags: surrounded by angle brackets < >
  - most tags come in pairs, marking a beginning and ending
    - `E.g., <title>` and `</title>` enclose the title of a page

HTML Layout

html_example.html

```
<html>

<head>
<title>HTML examples</title>
</head>

<body>
<h1>HTML demonstration</h1>
<p>This is a sample HTML page.</p>
</body>

</html>
```

Open in a browser:

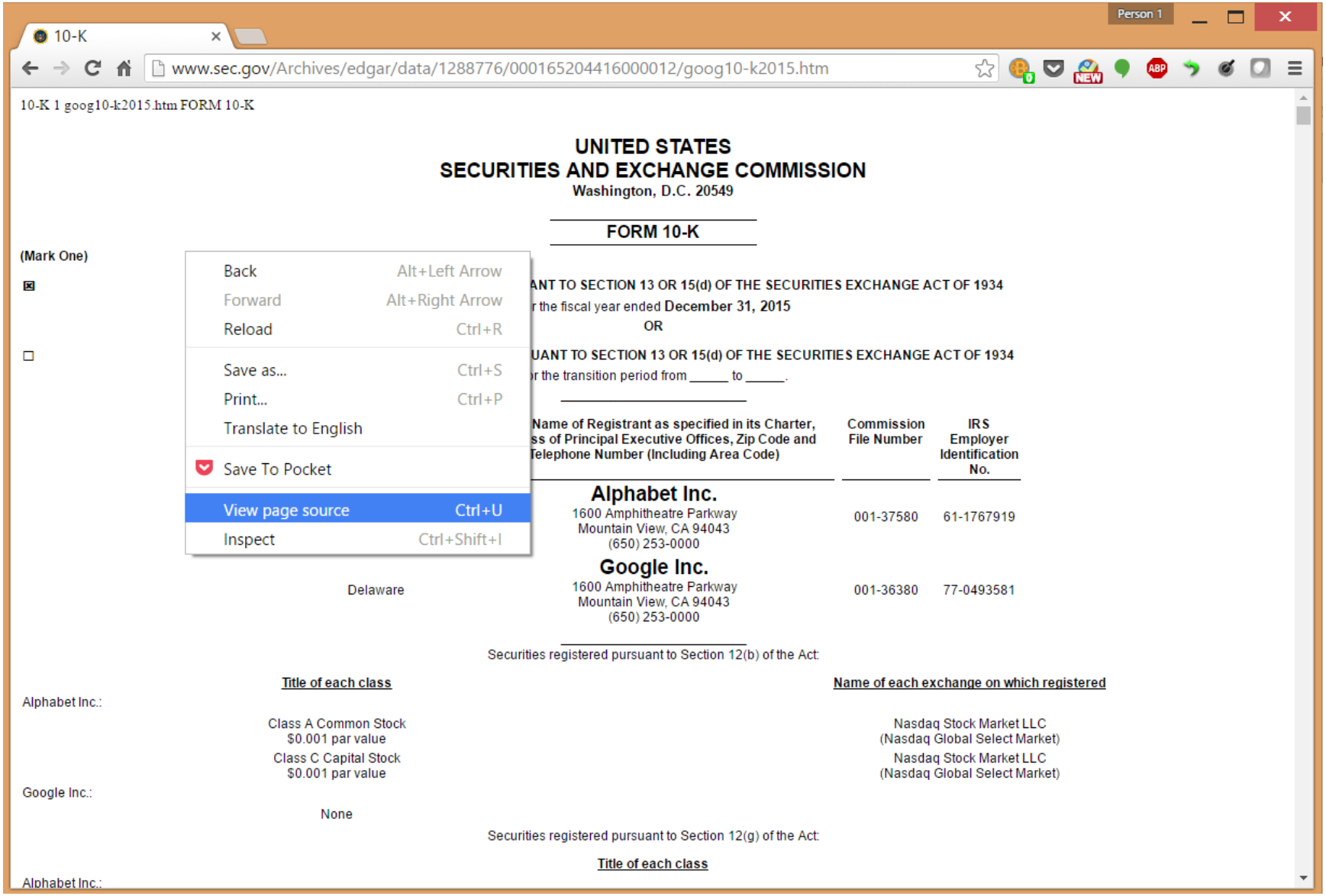


More on HTML tags: check HTML tutorial from W3schools.

View HTML Source Code

- To inspect the HTML page in details, you can do one of the following:
  - In Firefox/Chrome : Right click > View Page Source
  - Open the HTML file in a text editor (eg, Notepad++)

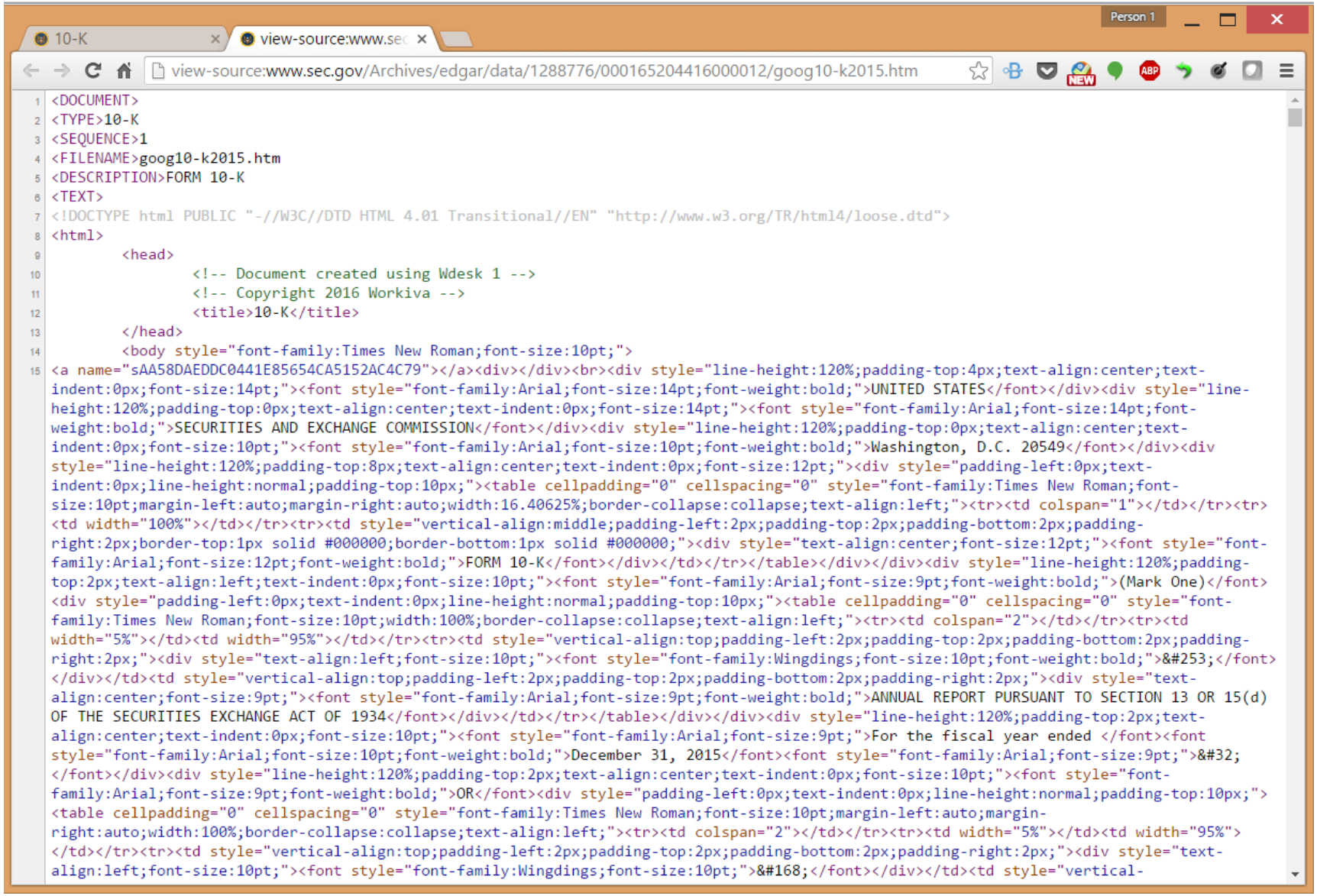
Example: 10-K



[View the webpage in browser]

Example: 10-K

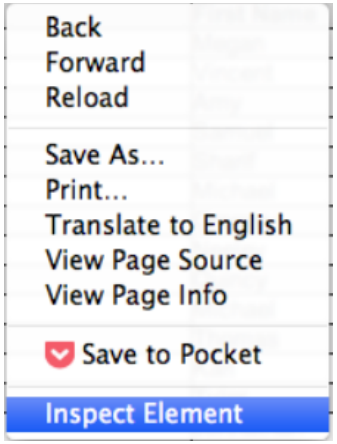
HTML source code:



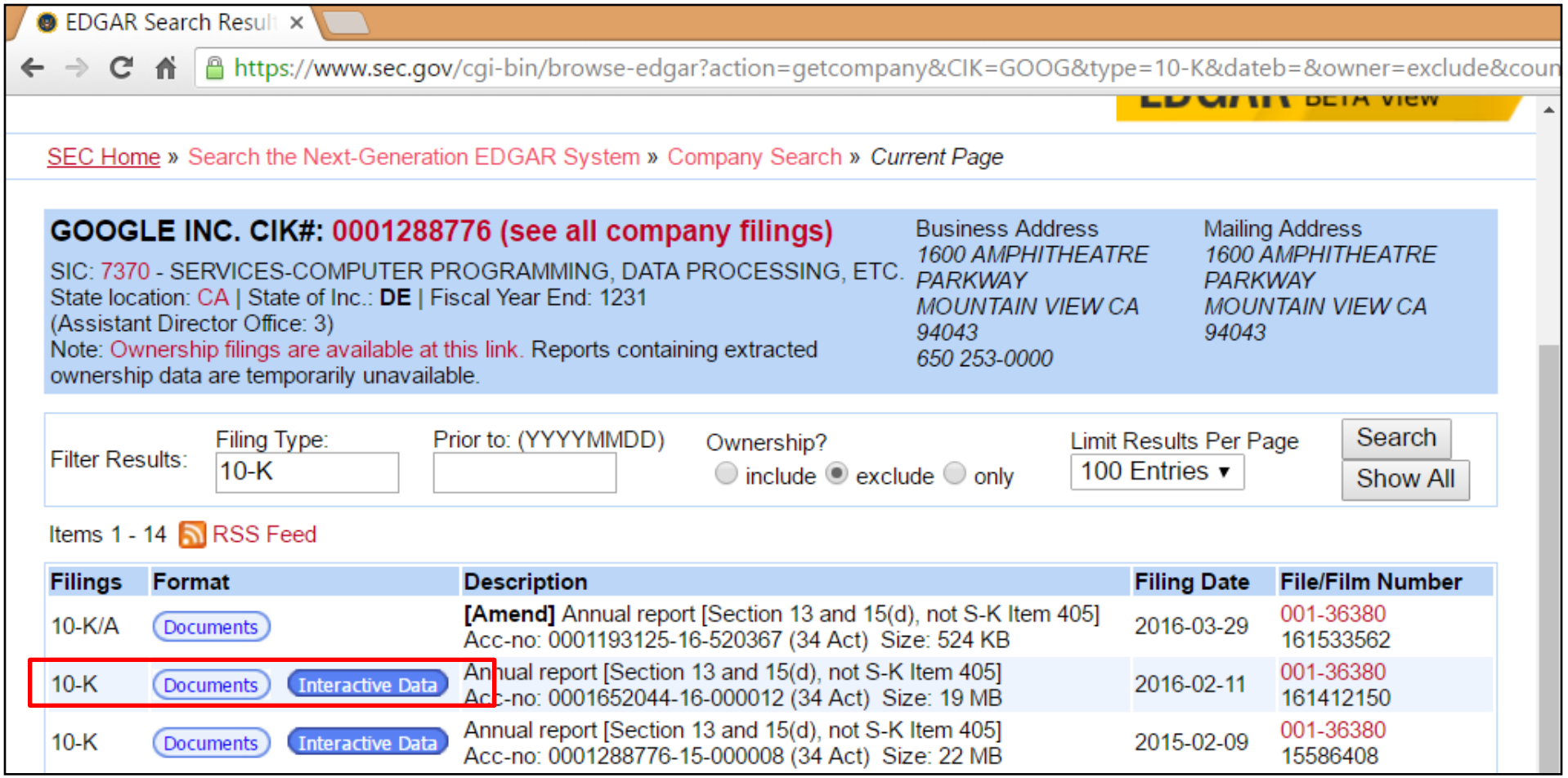
Inspect Elements

- To inspect a specific element on the HTML page, you can do one of the following:

  - In Chrome: Right click on the element> Inspect
  - In Firefox: Right click on the element> Inspect Element



Example: Table with Links



Example: Table with Links

HTML source code:

Ways to Extract Data from HTML

- The bs4 (BeautifulSoup) Package
  - Used for pulling data out of HTML and XML files
- The re (regular expression) Package
  - Can be used for both HTML and plain text files

The bs4 (Beautiful Soup) Package

- Installing the package in your command line interface:

```
pip install beautifulsoup4
```

- Import the package in Python

```
from bs4 import BeautifulSoup
```

Visit here to learn more about Beautiful Soup:
https://www.crummy.com/software/BeautifulSoup/

Example: Extracting Links from a Table

Filings	Format	Description	Filing Date	File/Film Number
10-K/A	Documents	**[Amend]** Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-16-520367 (34 Act) Size: 524 KB	2016-03-29	001-36380 161533562
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001652044-16-000012 (34 Act) Size: 19 MB	2016-02-11	001-36380 161412150
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001288776-15-000008 (34 Act) Size: 22 MB	2015-02-09	001-36380 15586408
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001288776-14-000020 (34 Act) Size: 21 MB	2014-02-12	000-50726 14595629
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-13-028362 (34 Act) Size: 13 MB	2013-01-29	000-50726 13556405
10-K/A	Documents	**[Amend]** Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-12-174477 (34 Act) Size: 772 KB	2012-04-23	000-50726 12771953
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-12-025336 (34 Act) Size: 11 MB	2012-01-26	000-50726 12548435
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-11-032930 (34 Act) Size: 14 MB	2011-02-11	000-50726 11600418
10-K	Documents Interactive Data	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-10-030774 (34 Act) Size: 3 MB	2010-02-12	000-50726 10601056
10-K	Documents	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-09-029448 (34 Act) Size: 1 MB	2009-02-13	000-50726 09605412
10-K	Documents	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-08-032690 (34 Act) Size: 1 MB	2008-02-15	000-50726 08623742
10-K	Documents	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-07-044494 (34 Act) Size: 2 MB	2007-03-01	000-50726 07664236
10-K	Documents	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-06-056598 (34 Act) Size: 1 MB	2006-03-16	000-50726 06691788
10-K	Documents	Annual report [Section 13 and 15(d), not S-K Item 405] Acc-no: 0001193125-05-065298 (34 Act) Size: 1 MB	2005-03-30	000-50726 05715414

Fetching the Webpage with `urllib2`

In Python:

```
>>> import urllib2
>>> urlLink = "https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=GOOG&type=10-K&dateb=&owner=exclude&count=100"
>>> pageRequest = urllib2.Request(urlLink)
>>> pageOpen = urllib2.urlopen(pageRequest)
>>> pageRead = pageOpen.read()
>>>
```

https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=GOOG&type=10-K&dateb=&owner=exclude&count=100

Extracting the Links with BeautifulSoup

In Python:

```
>>> from bs4 import BeautifulSoup
>>> soup = BeautifulSoup(pageRead,"html.parser")
>>> table = soup.find("table",{"class":"tableFile2"})
>>> links = []
>>> for row in table.findAll("tr"):
...     cells = row.findAll("td")
...     if len(cells)==5:
...         link=cells[1].find("a",{"id":"documentsbutton"})
...         docLink="https://www.sec.gov"+link['href']
...         links.append(docLink)
```

Extracting the Links with BeautifulSoup

What we will get:

```
https://www.sec.gov/Archives/edgar/data/1288776/000119
312516520367/0001193125-16-520367-index.htm

https://www.sec.gov/Archives/edgar/data/1288776/000165
204416000012/0001652044-16-000012-index.htm

https://www.sec.gov/Archives/edgar/data/1288776/000128
877615000008/0001288776-15-000008-index.htm

https://www.sec.gov/Archives/edgar/data/1288776/000128
877614000020/0001288776-14-000020-index.htm

……
```

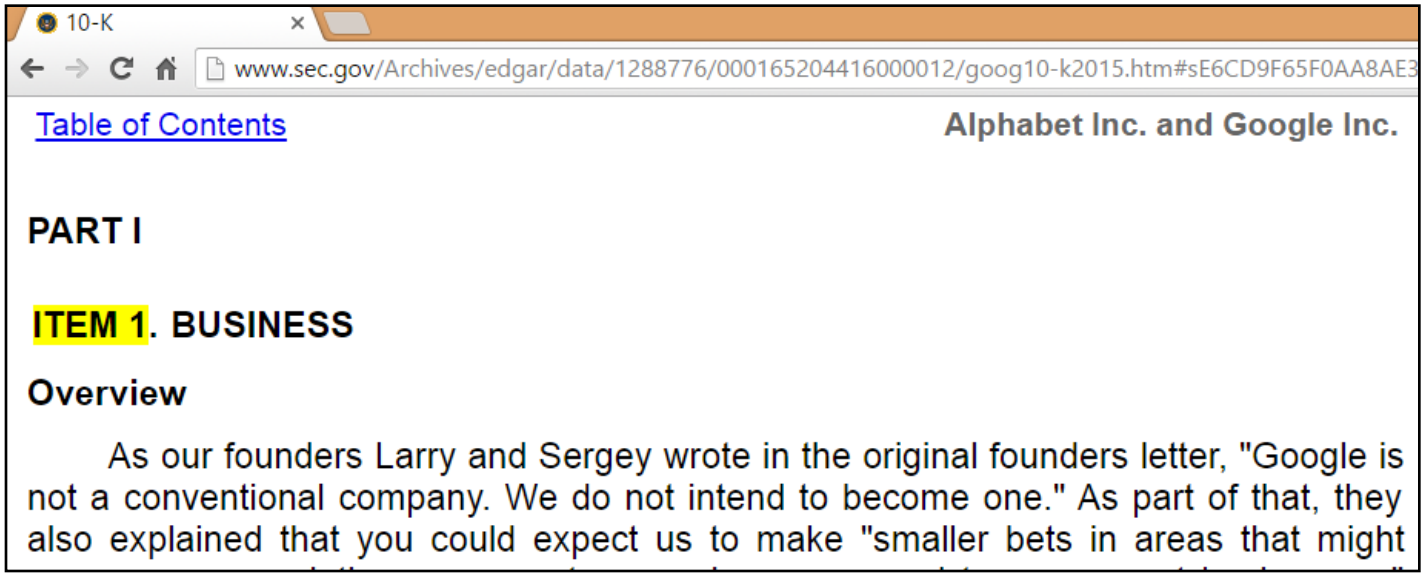
Extracting Textual Data Using `re`

- We talked about *Regular Expressions*
  - Powerful **text** manipulation tool for searching, replacing, and parsing text patterns

- In Python, you need to load the "**`re`**" package

```
>>> import re
```

Example: Item 1 of 10-K



Inspect element "ITEM 1."

HTML source code:

```
<div style="text-align:left;font-size:10pt;">
   <font style="font-family:Arial;font-size:10pt;font-
   weight:bold;">ITEM 1.</font>
</div>
```

Having the subtitle "ITEM 1." in bold makes sure that it is in the subtitle, not in main text

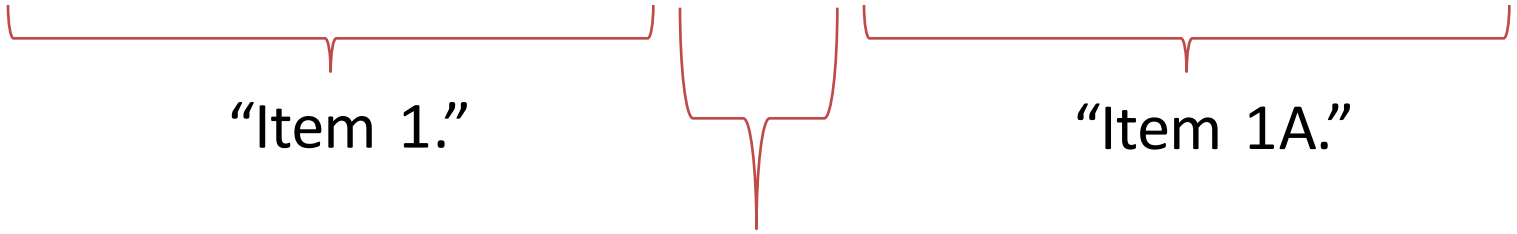
Extracting Textual Data Using `re`

```
# assume we have pre-processed the webpage
# and the page content is stored in a variable "page"

>>> import re
>>> regex="bold;\">\s*Item 1\.(.+?)bold;\">\s*Item 1A\."

>>> match = re.search(regex, page, flags=re.IGNORECASE)

#returns everything between "Item 1." and "Item 1A."
>>> match.group(1)
```

Anatomy of the RE Pattern

- We used the following pattern:

```
'bold;\">\s*Item 1\.(.+?)bold;\">\s*Item 1A\.'
```



`(.+?)` represents everything else in between that will be extracted

- What does each element mean?

Regular Expression	Corresponding Text
\"	"
\s	Whitespace (such as space, tab, new line)
*	Repeats the preceding character zero or more times
\.	. (dot)

More Basic Patterns

Symbols	Meaning
^	Matches the beginning of a line
$	Matches the end of the line
. (dot)	Matches any character but a whitespace
\s	Matches a single whitespace
*	Repeats a character zero or more times
+	Repeats a character one or more times
?	Repeats a character zero or 1 times
[xyz]	Matches any of x, y, z
\w	Matches a letter or digit or underbar
\d	Matches a digit [0-9]
()	Indicates where string extraction is to start and end

Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Download the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

Storing information to a csv file

```
# Previously we have a list of links extracted
# and stored in a variable “links”

>>> import csv
>>> csvOutput = open("IndexLinks.csv", "wb")
>>> csvWriter = csv.writer(csvOutput, quoting =
csv.QUOTE_NONNUMERIC)
>>> for link in links:
...     csvWriter.writerow([link])
>>> csvOutput.close()
```

Outline

- Overview
- Steps in Web Scraping
  - Fetching a Webpage
  - Download the webpage
  - Extracting information from the webpage
  - Storing information in a file
- Tutorial 2: Extracting Textual Data from 10-K

Tutorial 2: Extracting Textual Data from 10-K

- Install the Beautiful Soup package

```
pip install beautifulsoup4
```

- Download the following files from our website, and put them into the same folder
  - 1GetIndexLinks.py
  - 2Get10kLinks.py
  - 3DownLoadHTML.py
  - 4ReadHTML.py
  - CompanyList.csv

Tutorial 2: Extracting Textual Data from 10-K

- Changing Working Directory
  - For each of the four scripts, change the working directory to where you put the company list (CompanyList.csv) by changing the following line:

```
os.chdir(‘/Users/alvinzuyinzheng/Dropbox/PythonWorkshop/scripts/')
```

- Run each Python Script one-by-one

Other Resources

- Books:
  - Web Scraping with Python: Collecting Data from the Modern Web (by Ryan Mitchell)
  - Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More (by Matthew A. Russell)
- Beautiful Soup Documentation:
  - https://www.crummy.com/software/BeautifulSoup/bs4/doc/