Data Crawl with Python

Python + Selenium

IMDB Demo

July 28^{th}, 2019

Outline

Introduction

- 通过 **Selenium** 与浏览器进行互动，**Python** 可以模仿浏览器操作，执行鼠标点击、键盘操作、网页数据获取等操作。在实际编程中，可以调出浏览器界面，进行可视化调整，十分方便快捷。结合简单的数据获取命令，**Python** + **Selenium** 即可完成数据爬取操作，同时 **Python** 可以继续用作后续的数据文件生成和数据处理等操作。

- 首先需要安装 **Python**, 之后需要安装 **Selenium** 包。由于在调用浏览器操作的时候，需要使用浏览器驱动，还需下载浏览器驱动，并将下载好的浏览器驱动（*chromedriver.exe*）与编写的程序（*.py*）放在相同路径下。

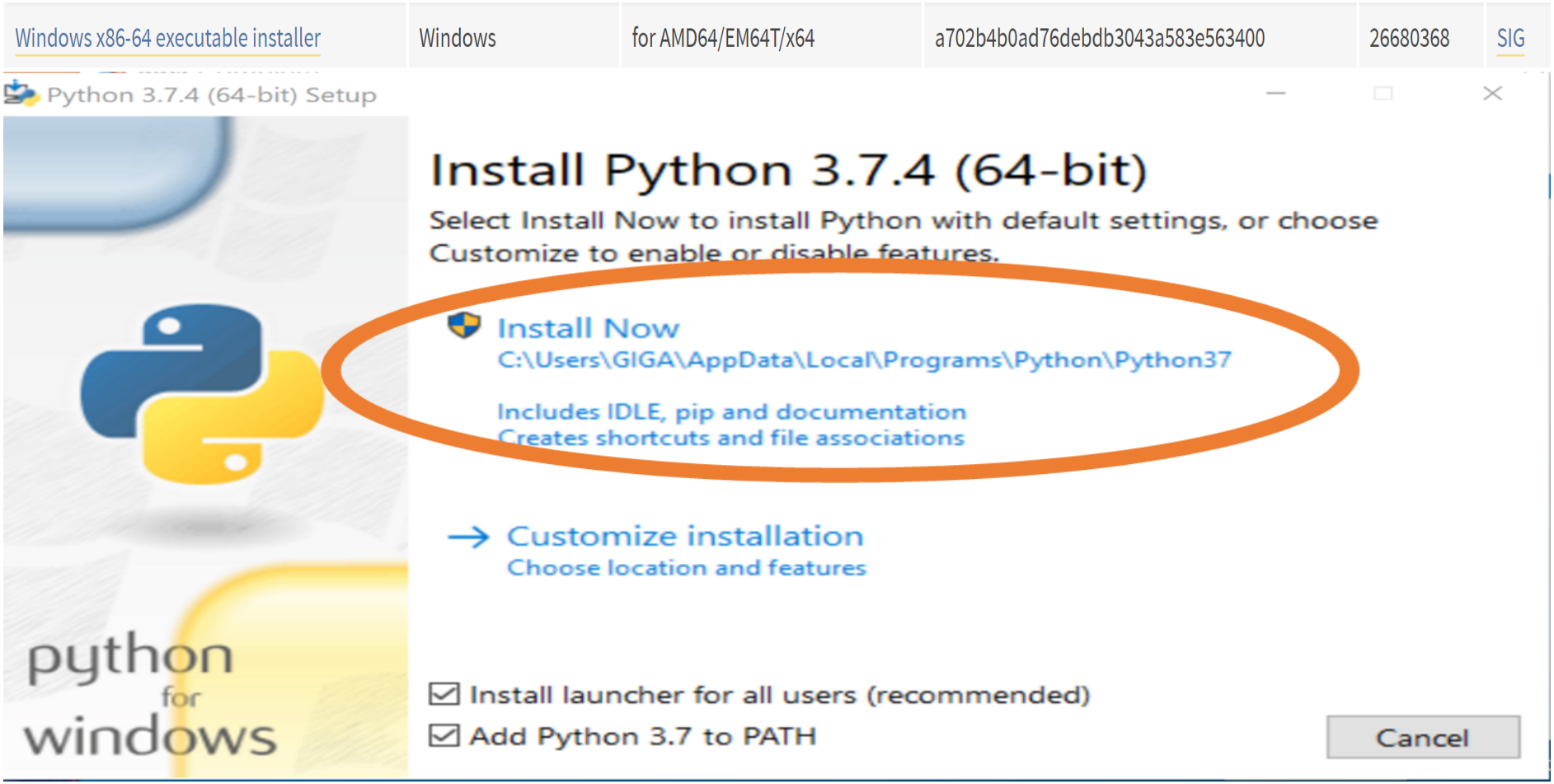
Examples

An example program for IMDB data crawling is given in *CrawlerDemo.py* and default chrome driver is given as *chromedriver.exe*

Python Installation

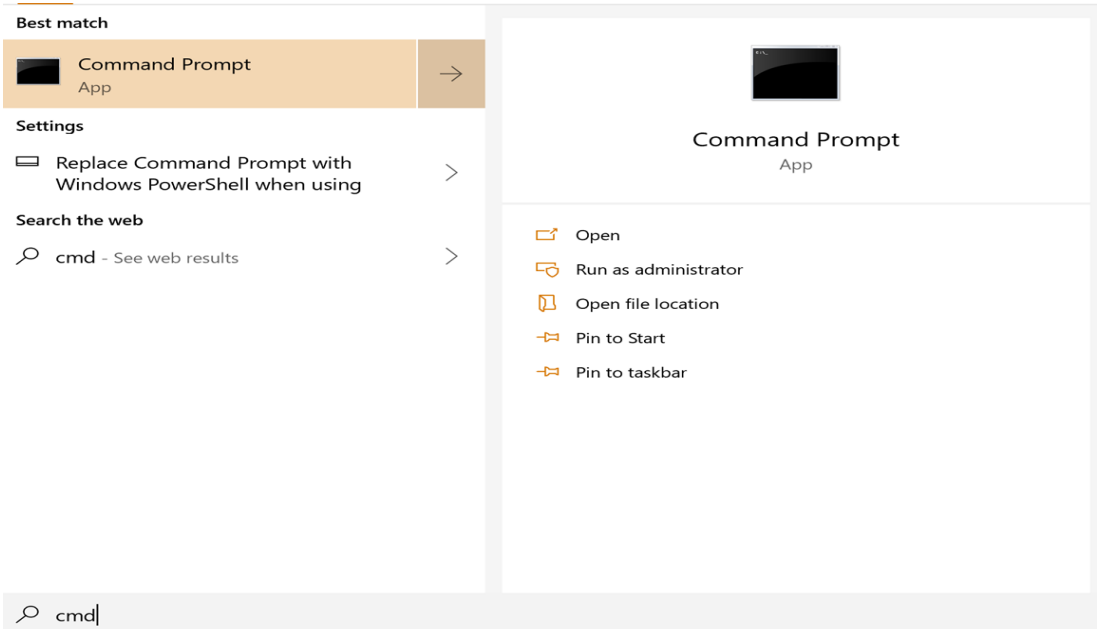
Python is available in following site...

https://www.python.org/downloads/release/python-374



Python Installation

After installing **Python**, use **cmd** to see whether the installation is successful. If console shows as following, installation is completed...



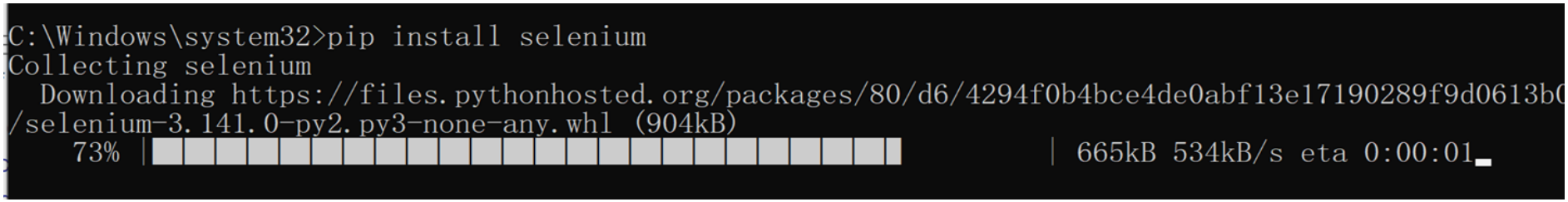

```
C:\Windows\system32>python
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

```
C:\Windows\system32>python
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()

C:\Windows\system32>
```

Selenium Installation

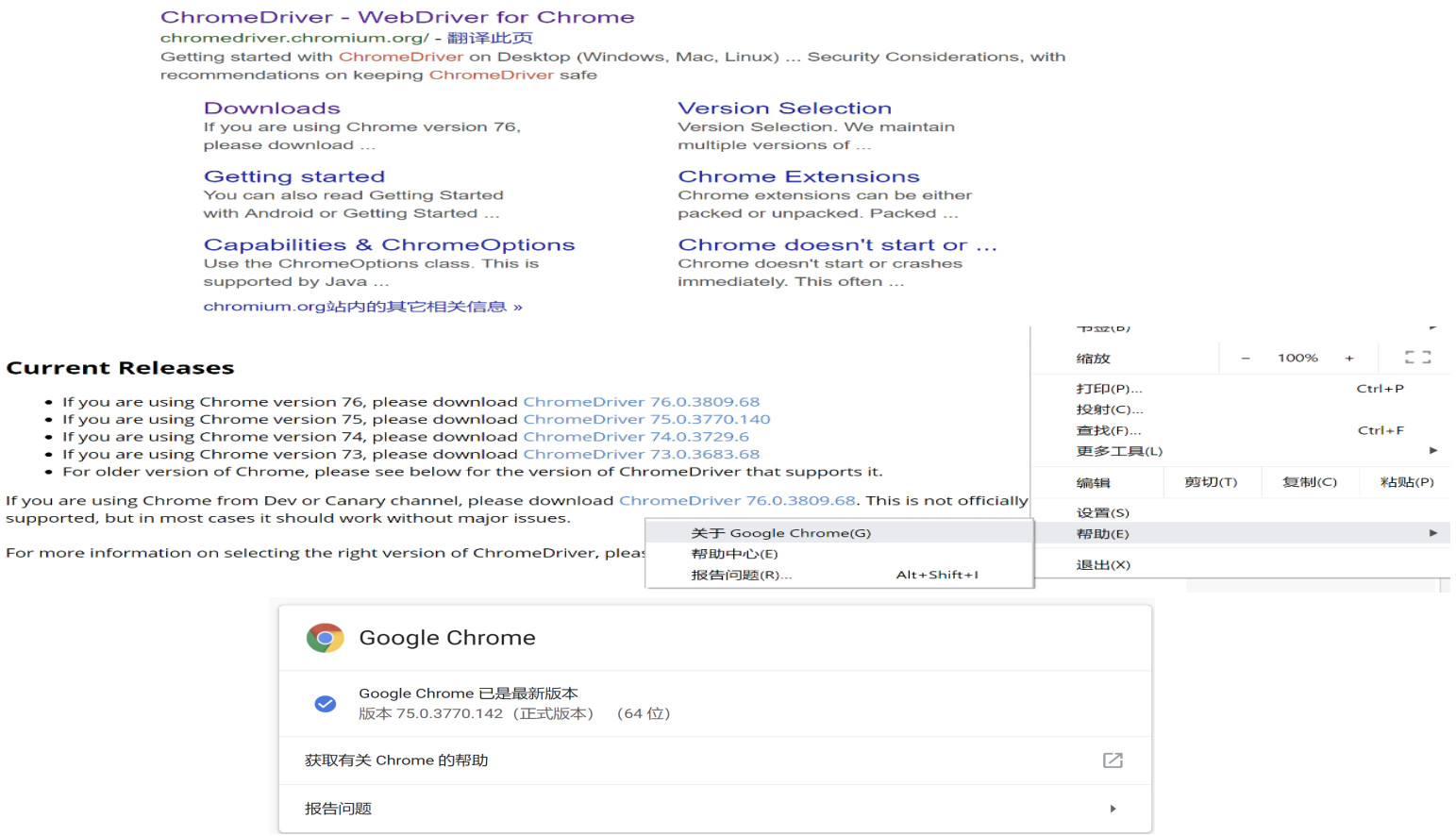
Selenium is required to perform visualized data crawl, thus this package should be downloaded and installed for our **Python**. **Pip** is a tool for **Python** to collect and install package from internet for local **Python**. "pip Install" command is used for this purpose...



Again, we use **cmd** to run "pip install" command. With above window showing "complete" information, the installation for **Selenium** is completed

Browser Driver Installation

Since **Selenium** runs by employing browser, browser driver is required. We use **Google Chrome** as our choice for browser and download its driver from website. A default version of *chromedriver.exe* is appended in our demo example...



Find Data in Website

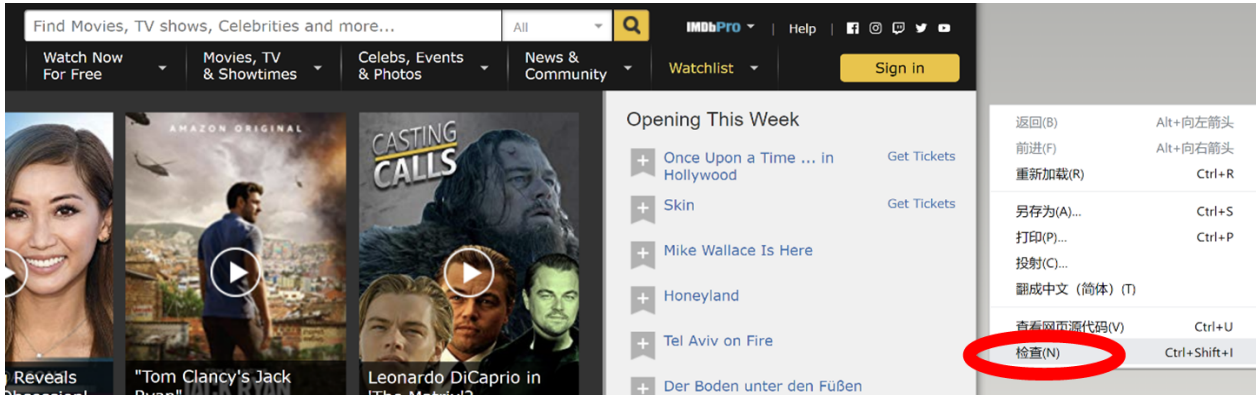
We use our demo program to illustrate the procedures for locating elements in website and retrieving desirable data. Reference file operation for data processing and storage is also available in code...
In our demo, we want to collect information, including directors and rating for some movies, from **IMDB**. The search conditions include *Movie Name* and *Release Year*...

- Following code starts the search in **IMDB**

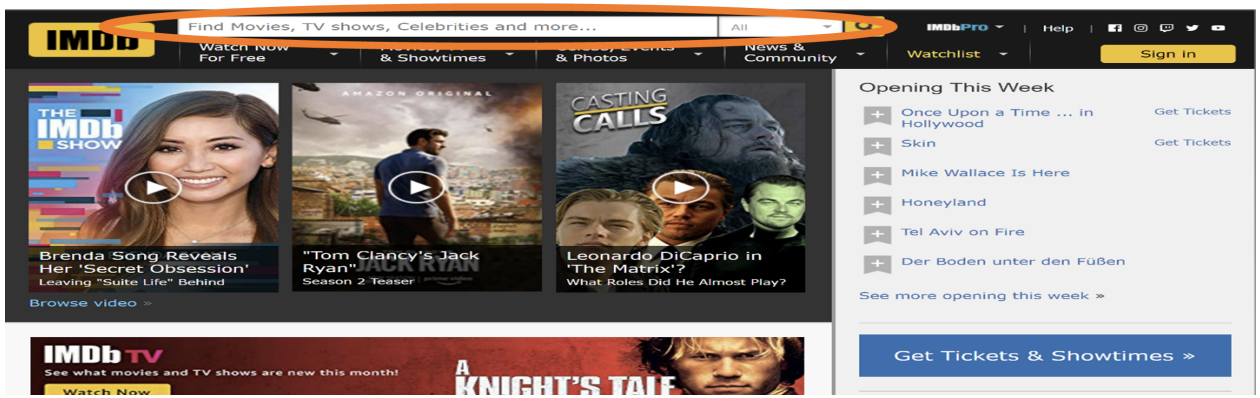
```
movieDirectorCrawled = list()
movieRatingCrawled = list()
for movie in movieInfoHolder:
    # visit the website
    chromeDriver.get(websiteCrawling)
    # locate and find search bar by xpath
    searchBar = chromeDriver.find_element_by_xpath(".//input[@type='text'][@name='q'][@id='navbar-query']")
    # input movie name, movie[0] + Keys.ENTER perform the same as following button click
    searchBar.send_keys(movie[0])
    # locate and find search button by xpath
    searchButton = chromeDriver.find_element_by_xpath(".//button[@id='navbar-submit-button'][@type='submit']")
    # click the search button
    searchButton.click()
    # sleep one seconds for loading the search results
    time.sleep(1)
```

Find Data in Website

- We need first find way to locate the desired data in website...

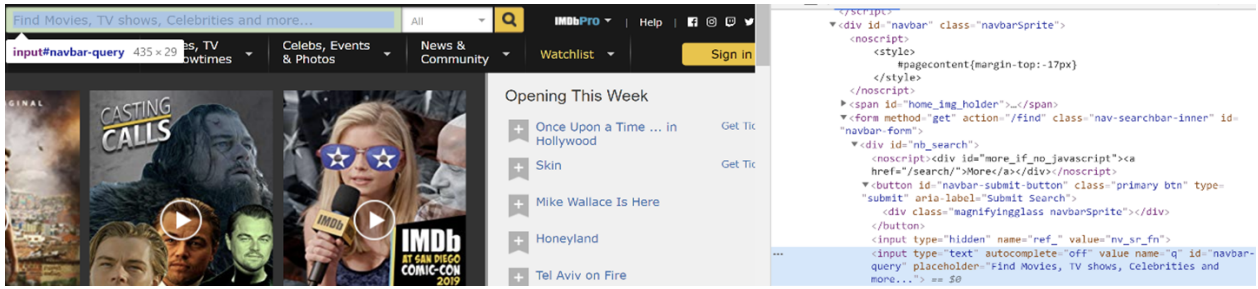


- Since we want to type movie name and then search in **IMDB**, the search bar and search button are wanted.
- We first locate the search bar for entering movie name...



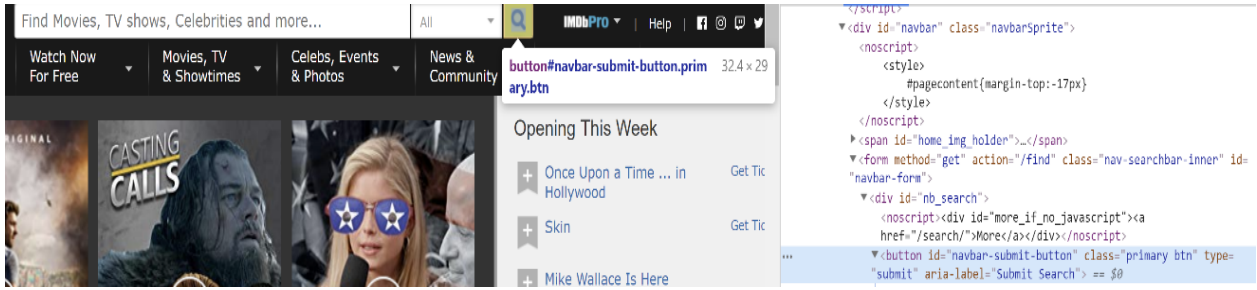
Find Data in Website

- We want to find its corresponding position in the web source file...



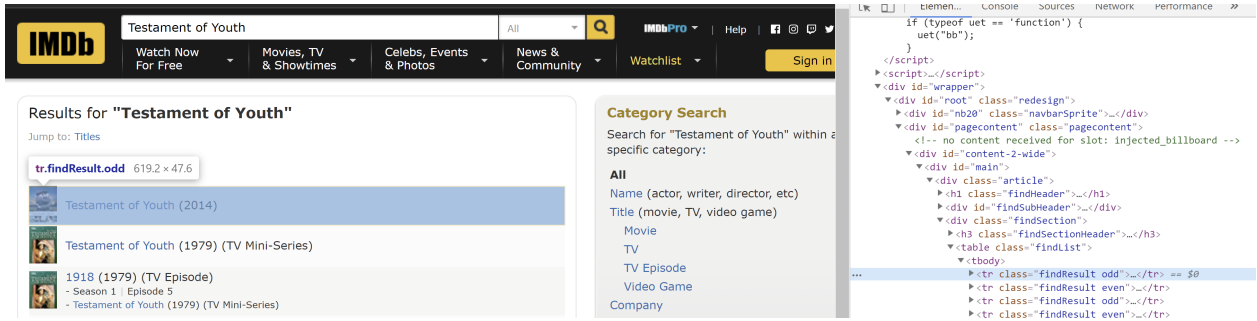
- We then locate the search button for submitting our search...

- We want to find its corresponding position in the web source file...



Find Data in Website

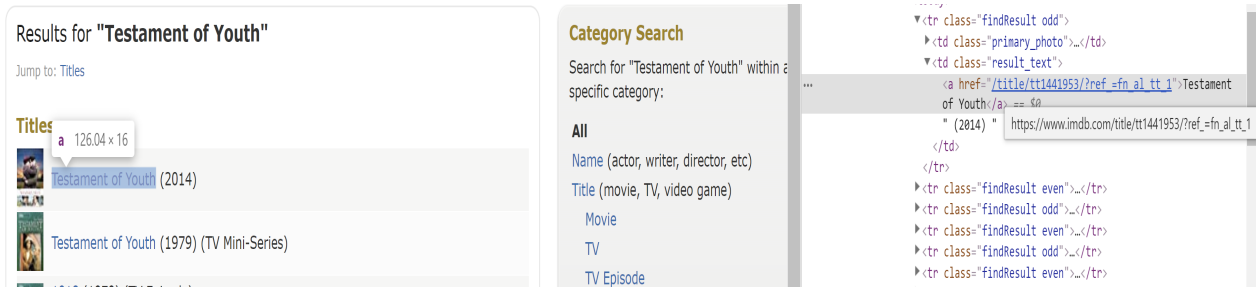
- After entering movie name and click search button to search in **IMDB**...



- Following code retrieves all valid options for our search, along with their link (use "Release Year" to identify the wanted movie)

```
# locate the section for movies, there are multiple "table" element with class as "findList", we want the first one
movieItemSection = chromeDriver.find_element_by_xpath(".//*[@id='main']/div/div[2]/table")
# find all movie items related with the searching movie name
movieItems = movieItemSection.find_elements_by_tag_name("tr")
```

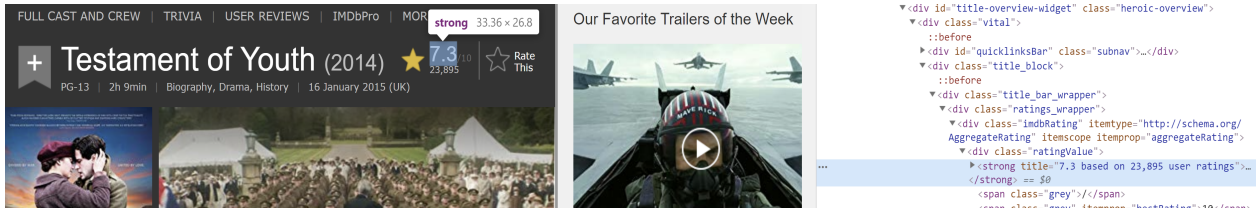
- Corresponding movie link information in the web source file...



Find Data in Website

- We go to detailed movie page to collect the information of movies...

- Corresponding rating information in the web source file...



- Corresponding director information in the web source file...

Find Data in Website

- Following code collect the director and rating information in each detailed movie page...

```
chromeDriver.get(movieLink)
time.sleep(1)
rating = chromeDriver.find_element_by_xpath(
    ".//*[@id='title-overview-widget']/div[1]/div[2]/div/div[1]/div[1]/div[1]/strong/span")
directorInfoElements = chromeDriver.find_element_by_xpath(
    ".//*[@id='title-overview-widget']/div[2]/div[1]").find_element_by_xpath(
    ".//h4[contains(text(), 'Director')]/..").find_elements_by_tag_name("a")
directorList = [info.text for info in directorInfoElements]
movieDirectorCrawled.append(directorList)
movieRatingCrawled.append(rating.text)
break
```

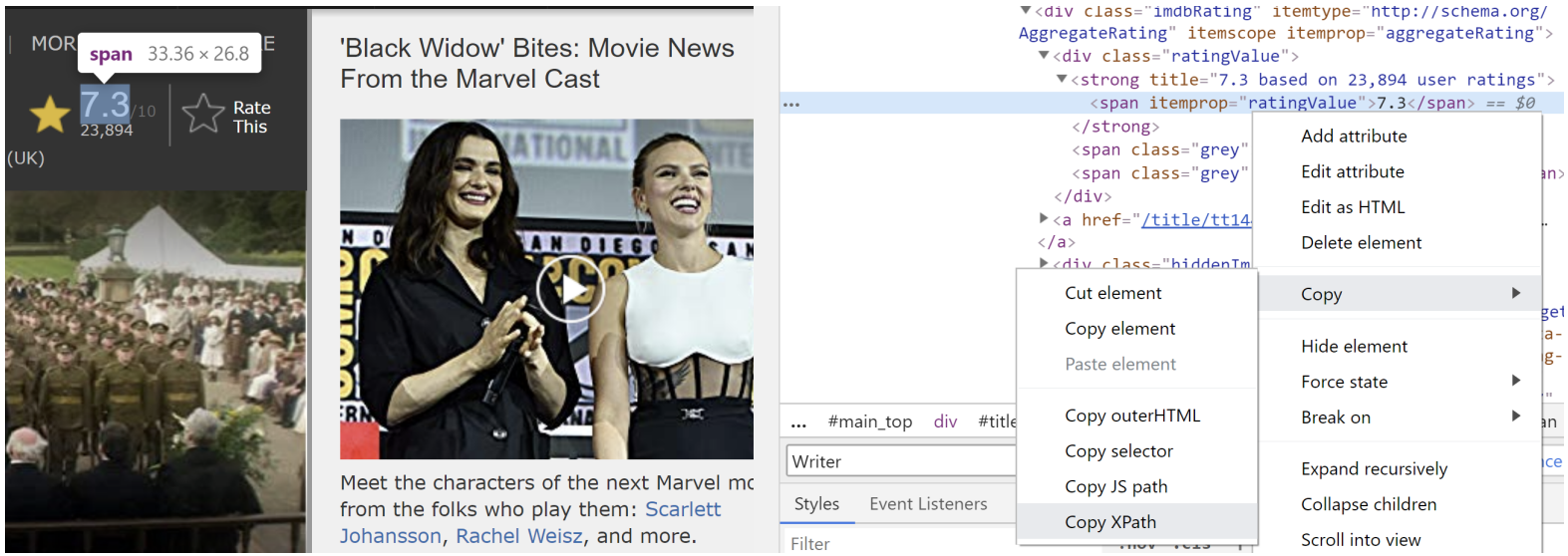
Appendix

- How to run Python program in WIN cmd console?

```
C:\Users\GIGA>cd C:\Users\GIGA\PycharmProjects\DataCrawler

C:\Users\GIGA\PycharmProjects\DataCrawler>python CrawlerDemo.py
```

- How to find and write the XPath of website element in easy way?



Reference

- About **XPath**
  Reading materials and extended guide for **XPath**

- About **Python**
  Reading materials and extended guide for **Python**