مبانی بازیابی اطلاعات و جست‌وجوی وب، تمرین سه

مهران سیفی - سجاد فرزانه - طهورا سعیدی نامقی

۹۹۱۲۷۶۲۲۲۳ - ۹۹۱۲۷۶۲۵۱۸ - ۹۹۲۲۷۶۲۳۷۷

هدف این پروژه استخراج مقالات مرتبط با یک حوزه خاص (استثنا در این کد حوزه Blockchain) از سایت IEEE Xplore می‌باشد. اطلاعات استخراج‌شده به دو ترتیب Relevance و Newest هستند.

برای راحتی کار یک تابع با امضای زیر تعریف شده است که سایت IEEE را با ترتیبی که به عنوان پارامتر sort به آن ارسال می‌شود Crawl می‌کند و نتایج را در قالب یک لیست برمی‌گرداند.

```
def crawl_ieee_xplore(sort):
```

مراحل انجام‌شده در این تابع و همچنین برنامه اصلی را در ادامه توضیح می‌دهیم.

۱) ابتدا سایت IEEE را توسط مرورگر Chrome باز می‌کنیم.

```
driver = webdriver.Chrome()
driver.maximize_window()
driver.get('https://ieeexplore.ieee.org/Xplore/home.jsp')
```

۲) سپس صبر می‌کنیم تا نوار جست‌وجوی موجود در سایت به طور کامل Load شود. یکی از چالش‌هایی که در مورد Wait در قسمت‌های مختلف با آن مواجه شدیم، کندی سرعت اینترنت و دیر Load شدن بخش‌های مختلف سایت و مواجه شدن با TimeoutException بود. بنابراین بعد از تست اعداد مختلف به عنوان پارامتر ارسالی به تابع Wait، با عدد ۱۲۰ ثانیه توانستیم به خروجی مورد نظر برسیم. در صورت بهتر بودن وضعیت اینترنت، می‌توان این مقدار این پارامتر را کاهش داد.

```
WebDriverWait(driver, 120).until(
    ec.presence_of_element_located((By.XPATH,

'//*[@id="LayoutWrapper"]/div/div/div[3]/div/xpl-root/header/xpl-
header/div/div[2]/div[2]/xpl-search-bar-migr/div/form/div[2]/div/div[1]/xpl-
typeahead-migr/div/input'))
)
```

۳) بعد از Load شدن نوار جست‌وجو، حال باید عبارت مورد نظر را در آن وارد کرده و جست‌وجو کنیم. در این تمرین استثنا مقالات در حوزه Blockchain را Crawl کردیم؛ اما به راحتی و با تغییر یک پارامتر، می‌توانیم همین کار را برای حوزه‌های دیگر نیز انجام دهیم. حتی بعد از نوشتن کد متوجه شدیم بهتر بود حوزه مورد جست‌وجو را نیز به عنوان یک پارامتر ورودی تابع crawl_ieee_xplore در نظر بگیریم! در ادامه گزارش موارد مشابه دیگری نیز ذکر کرده‌ایم و بهبودهایی که می‌توانستیم در کد اعمال کنیم تا به صورت پویاتر عمل کند را بیان کرده‌ایم. اما به دلیل کمبود وقت و زمان نسبتا طولانی که برای هر بار اجرای کد لازم است، این موارد را در کد اعمال نکرده‌ایم.

```
search_box = driver.find_element(By.XPATH,

'//*[@id="LayoutWrapper"]/div/div/div[3]/div/xpl-root/header/xpl-
header/div/div[2]/div[2]/xpl-search-bar-migr/div/form/div[2]/div/div[1]/xpl-
typeahead-migr/div/input')
search_box.send_keys('Blockchain')
search_box.send_keys(Keys.RETURN)
```

۴) حال باید صبر کنیم تا قسمت Drop down مربوط به انتخاب نحوه Sort نتایج در سایت Load شود.

```
WebDriverWait(driver, 120).until(
    ec.presence_of_all_elements_located((By.XPATH,

'/html/body/div[5]/div/div/div[3]/div/xpl-root/main/div/xpl-search-
results/div/div[2]/div[2]/xpl-results-list/div[2]/xpl-select-
dropdown/div/button'))
)
```

۵) سپس باتوجه به پارامتر sort انتخاب می‌کنیم که روی کدام بخش کلیک شود. البته بعد از نوشتن کد متوجه شدیم این قسمت را نیز می‌توانستیم طوری بنویسیم که محدود به دو حالت Relevance و Newest نباشد. یعنی شرط‌های بیشتر روی پارامتر sort می‌گذاشتیم و تمامی حالت‌های ممکن برای Sort شدن نتایج را در نظر می‌گرفتیم!

```
if sort == "newest":
    drop_down = driver.find_element(By.XPATH,

'/html/body/div[5]/div/div/div[3]/div/xpl-root/main/div/xpl-search-
results/div/div[2]/div[2]/xpl-results-list/div[2]/xpl-select-
dropdown/div/button')
    drop_down.click()

    newest = driver.find_element(By.XPATH,

'//*[@id="xplMainContent"]/div[2]/div[2]/xpl-results-list/div[2]/xpl-select-
dropdown/div/div/button[2]')
    newest.click()
```

۶) بعد لیستی را به منظور ذخیره نتایج در نظر می‌گیریم. چون قصد داریم نتایج ۵ صفحه اول را بررسی کنیم به یک حلقه از ۱ تا ۵ نیاز داریم (عدد ۵ هم می‌توانست به عنوان پارامتر ورودی تابع crawl_ieee_xplore در نظر گرفته شود تا تابعی که نوشتیم پویاتر باشد!). سپس به یک حلقه با ۲۵ دور برای بررسی ۲۵ مقاله موجود در هر صفحه نیاز داریم که چون XPathهای آن‌ها شامل اعداد ۳ تا ۲۷ بود، ما هم این حلقه را از ۳ تا ۲۷ در نظر گرفتیم (عدد ۲۵ نیز می‌توانست به عنوان پارامتر ورودی تابع crawl_ieee_xplore باشد!).

```
articles = []

for i in range(1, 6):

    print(f'\n********** Page number {i} **********\n')

    for j in range(3, 28):

        print(f'Article number {j - 2}:')
```

۷) حال باید صبر کنیم تا تمامی مقالات موجود در صفحه Load شوند.

```
WebDriverWait(driver, 120).until(
    ec.presence_of_all_elements_located((By.XPATH,

'/html/body/div[5]/div/div/div[3]/div/xpl-root/main/div/xpl-search-
results/div/div[2]/div[2]/xpl-results-list/div[27]/xpl-results-
item/div[1]/div[1]/div[2]/h3/a'))
)
```

۸) یک چالش اصلی که با آن مواجه شدیم کار نکردن تابع driver.back() بود. بعد از باز کردن هر مقاله و استخراج اطلاعات آن، نیاز داشتیم به صفحه قبل بازگشته و سراغ مقاله بعدی برویم. اما این تابع در مواردی به درستی کار نمی‌کرد. حتی بعد از جست‌وجو سعی کردیم به روش‌های دیگری این کار را انجام دهیم. به عنوان مثال استفاده از تابع driver.execute_script("window.history.go(-1)") که متاسفانه باز هم جواب نداد. لذا تصمیم گرفتیم URL صفحات را قبل از باز کردن مقالات ذخیره کنیم و بعد از استخراج اطلاعات مقاله، به URL ذخیره‌شده بازگردیم.

```
back_url = driver.current_url
```

۹) سپس بررسی می‌کنیم تا بخشی که قصد باز کردن و ذخیره اطلاعات آن را داریم حتما مقاله باشد و از نوع کتاب یا دوره درسی نباشد. اگر از نوع مقاله بود با کلیک بر روی عنوان آن استخراج اطلاعات آن را آغاز می‌کنیم.

```
type_ = driver.find_element(By.XPATH,
                            f'/html/body/div[5]/div/div/div[3]/div/xpl-
root/main/div/xpl-search-results/div/div[2]/div[2]/xpl-results-
list/div[{j}]/xpl-results-
item/div[1]/div[1]/div[2]/div/div[1]/span[2]/span[2]').text
if type_ != "Conference Paper":
    continue

open_article = driver.find_element(By.XPATH,

f'/html/body/div[5]/div/div/div[3]/div/xpl-root/main/div/xpl-search-
results/div/div[2]/div[2]/xpl-results-list/div[{j}]/xpl-results-
item/div[1]/div[1]/div[2]/h3/a')
open_article.click()

WebDriverWait(driver, 120).until(
    ec.presence_of_all_elements_located((By.XPATH, '/html'))
)
```

۱۰) بعد از باز شدن صفحه مربوط به مقاله مورد نظر، اطلات خواسته‌شده را (در صورت وجود) استخراج می‌کنیم. به دلیل تکراری و واضح بودن این عمل از توضیح خط‌به‌خط کدهای مربوط به این قسمت خودداری شده است. لازم به ذکر است در مواردی که با پیدا کردن Elementها در صفحه مشکل داشتیم؛ به جای XPath از Full XPath استفاده کردیم.

۱۱) حال که اطلاعات مورد نظر را استخراج کردیم، آن‌ها در لیست نتایج ذخیره می‌کنیم.

```
articles.append({
    "title": title,
    "Cites in Papers": papers,
    "Cites in Patent": patent,
    "Full Text Views": views,
    "Publisher": publisher,
    "DOI": doi,
    "Date of Publication": date,
    "abstract": abstract,
    "Published in": published,
    "Authors": authors,
    "IEEE Keywords": ieee_keywords,
    "Author Keywords": author_keywords
})
```

۱۲) سپس به URL صفحه قبل که آن را قبلا ذخیره کرده بودیم بازمی‌گردیم و بررسی سایر مقالات را به طریق مشابه ادامه می‌دهیم.

```
driver.get(back_url)
```

۱۳) بعد از اتمام بررسی مقالات موجود در یک صفحه، لازم است تا به صفحه بعد برویم. این کار را به صورت زیر انجام می‌دهیم. ابتدا صبر می‌کنیم تا Button مربوط به رفتن به صفحه بعد به طور کامل Load شود، روی آن کلیک کرده و برای Load شدن کامل صفحه جدید نیز صبر می‌کنیم.

```
WebDriverWait(driver, 120).until(
    ec.presence_of_all_elements_located((By.CLASS_NAME, f'stats-Pagination_{i
+ 1}'))
)

next_button = driver.find_element(By.CLASS_NAME, f'stats-Pagination_{i + 1}')
next_button.click()

WebDriverWait(driver, 120).until(
    ec.presence_of_all_elements_located((By.XPATH,

'/html/body/div[5]/div/div/div[3]/div/xpl-root/main/div/xpl-search-
results/div/div[2]/div[2]/xpl-results-list/div[27]/xpl-results-
item/div[1]/div[1]/div[2]/h3/a'))
)
```

۱۴) بعد از این که اطلاعات مربوط به تمامی مقالات در تمامی صفحات را بررسی کردیم، نتایج را هم در یک فایل با فرمت json. ذخیره می‌کنیم و هم به عنوان خروجی تابع برمی‌گردانیم.

```
with open(f'{sort}.json', 'w') as json_file:
    json.dump(articles, json_file, indent=4)
return articles
```

۱۵) تابع نوشته‌شده را با دو پارامتر relevance و newest فراخوانی می‌کنیم و نتایج هر دو فراخوانی را در یک فایل با نام Articles.json ذخیره می‌کنیم.

```
relevance = crawl_ieee_xplore("relevance")
newest = crawl_ieee_xplore("newest")
result = [relevance, newest]
with open('Articles.json', 'w') as result_file:
    json.dump(result, result_file, indent=4)
```

مشارکت اعضای تیم در انجام این پروژه کاملا برابر و برای همه اعضا در همه بخش‌ها ۱۰ از ۱۰ بوده است.