

YBIGTA Engineering Study Session (3 & 4)

Hyunjoe Yoo

hyunjoe.yoo@mail.mcgill.ca

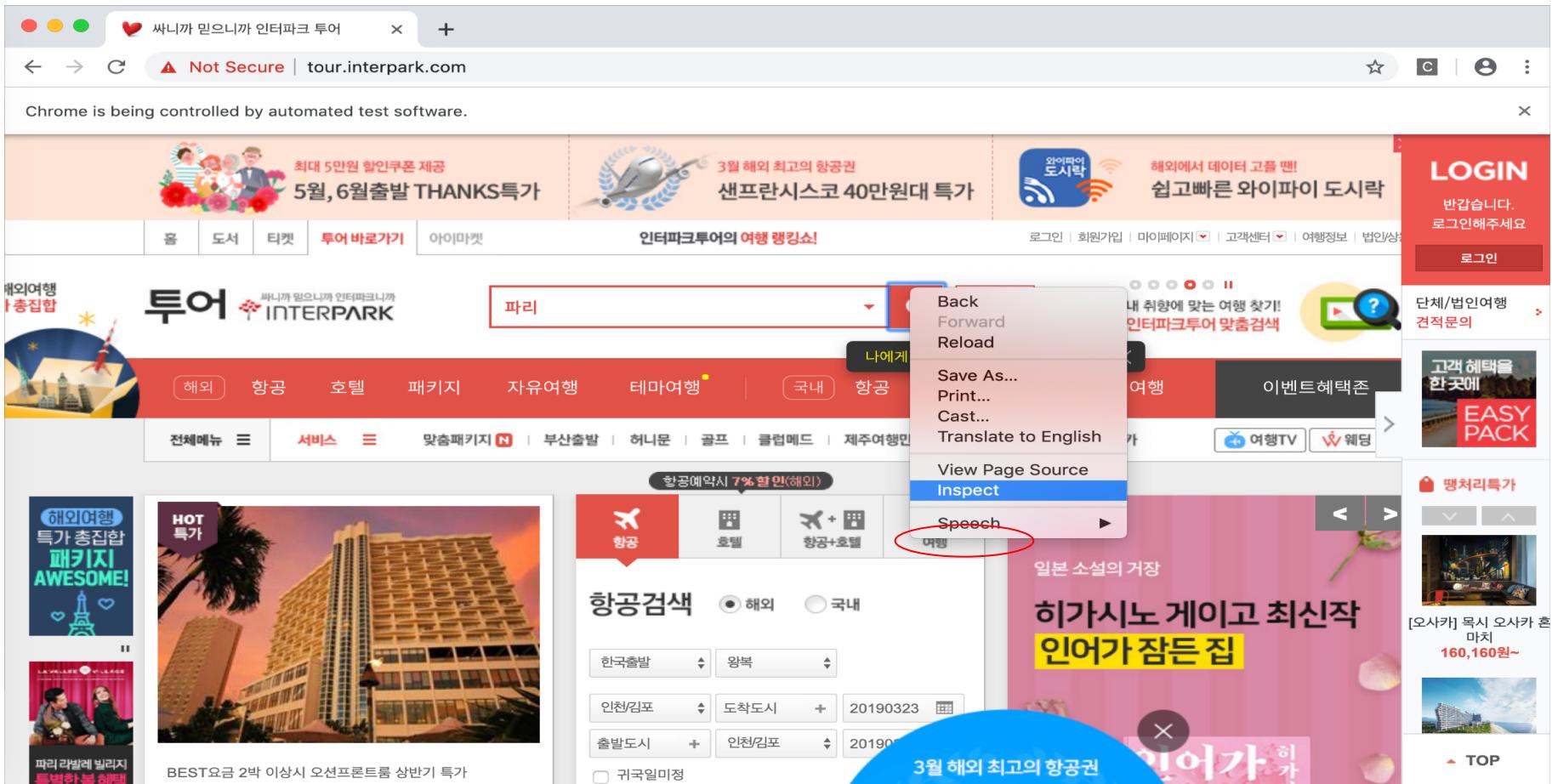
(3강): Crawling study week 2

- 수정할 경우 → .clear() → send_keys('content')

```
#3_web site 접속
driver.get(main_url) ✓

#4_검색창을 찾아서 검색어를 입력
driver.find_element_by_id('SearchGNBText').send_keys(key_word) ✓
#수정 하고 싶을 때! 뒤에 내용 붙는거 방지하기 위해 --> .clear() --> send_keys('content')
```

Selenium 모듈 실습 1 (검색)



Search-btn

The screenshot shows a Chrome browser window with the URL tour.interpark.com. The page displays a travel booking interface for Interpark Travel. A search bar at the top contains the text "파리". To the right of the search bar, there is a "LOGIN" section and a dropdown menu. The developer tools are open, with the "Elements" tab selected. In the CSS panel, a button with the class "search-btn" is highlighted with a red circle. The CSS code for this button includes:

```
button.search-btn {
    type="button";
    onclick="searchBarModule.ClickForSearch();";
    title="검색";
}
```

The browser's address bar shows a warning: "Not Secure". The status bar at the bottom right indicates the date and time: "Thu Mar 21 10:41:05 PM".

검색버튼을 찾아서 클릭!

- .search-btn
- Button.search-btn

```
#5_검색 버튼 클릭  
driver.find_element_by_css_selector('button.search-btn').click()
```

Import

- 대기를 해줘야 오류 방지가 됨!
- 타겟으로 찾고자 하는 요소가 올라올 때 까지 기다리기
- 화면에서 화면으로 넘어갈 때 꼭 있어야 오류 방지!

```
#잠시 대기! --> 페이지가 로드되고 나서 데이터를 즉각적으로 획득하는 행위는
#명시적 대기 --> 특정 요소가 로케이트 때 까지 대기
from selenium.webdriver.common.by import By ✓
#명시적 대기를 위해
from selenium.webdriver.support.ui import WebDriverWait ✓
from selenium.webdriver.support import expected_conditions as EC ✓
```

정보 로드

```
#사전에 필요한 정보를 로드 --> 디비혹스 웹, 배치 파일에서 인자로 받아서 셋팅  
main_url = "http://tour.interpark.com/"  
keyword = '파리'
```

Webdriver wait

```
try:  
    element = WebDriverWait(driver, 10).until(  
        #지정한 한 개 요소가 올라오면 웨이트 종료  
        EC.presence_of_element_located(By.)  
  
    )  
except Exception as e:  
  
    #암묵적 대기 --> DOM이 다 로드 될 때까지 대기 하고 먼저 로드되면 바로 진행  
    #절대기 대기 --> time.sleep(10) --> 클라우드 페어 (디도스 방어 솔루션)
```

해외여행 카테고리로 이동



[139개의 상품평](#)



호텔 키리아드 파리 베르시 빌리지

Hotel Kyriad Paris Bercy Village

등급: ★★★★☆☆☆

17 Rue Baron Le Roy [지도보기](#)



86,240 원~

평점8.2

[230개의 상품평](#)

스팟 (206)



프랑스/파리 숙소

아카디아 오페라

파리의 비즈니와 극장지역인 Montmartre와 the Opera Garnier 사이에 위치하며, Opera Garnier와 백화점



[Look Up "해외호텔" 더보기 ▶](#)

Copy

Search Google for "해외호텔"

Print...

Inspect

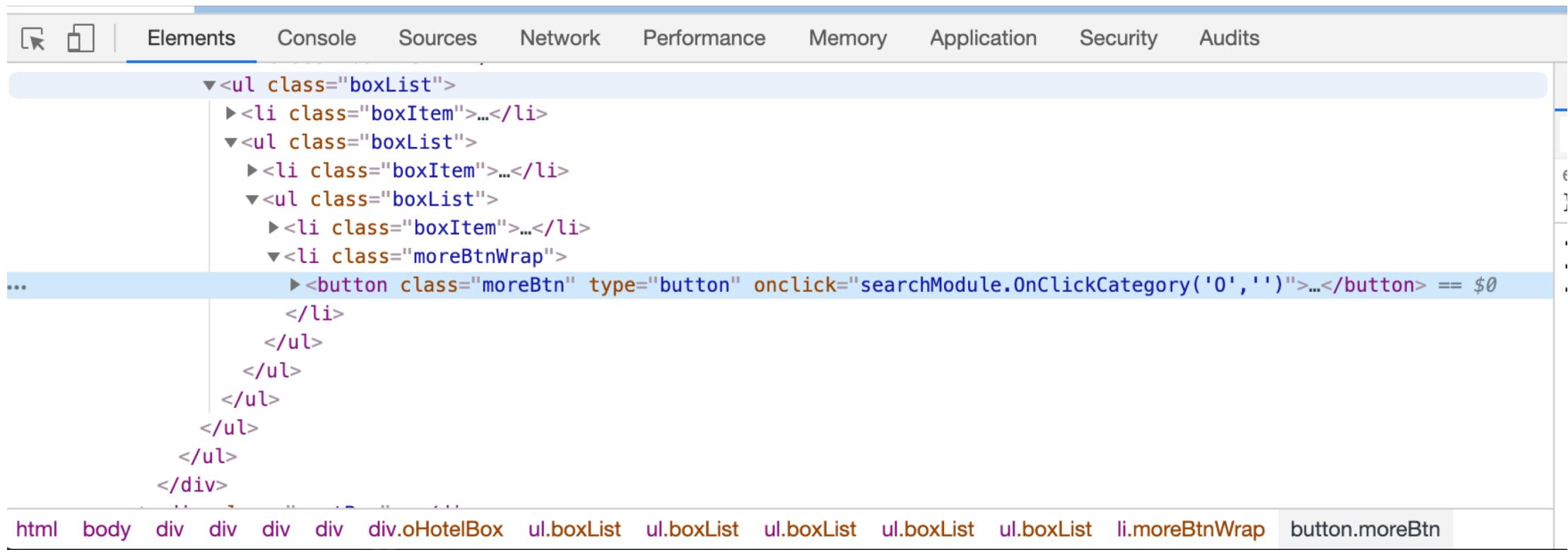
Speech

Services

[올 시즌 파리 베르시](#)

3성급 호텔로 총객실은 364개 제공되며, 이 중 금연룸은 138개이다. 파리의 동쪽의 비지니스 지역에 위치하고 오페라와 백화점에서 10분 거리이다. Bercy 공원을 내려다 보

moreBtn?



The screenshot shows the Chrome DevTools Elements tab open, displaying a hierarchical tree of HTML elements. The tree is rooted at a `<ul class="boxList">` element. This main list contains several `<li class="boxItem">` items. One of these items contains another `<ul class="boxList">` element, which in turn contains a `<li class="boxItem">` item. This pattern repeats, creating a nested list structure. The final item in the innermost list is a `<li class="moreBtnWrap">`, which contains a single `<button class="moreBtn" type="button" onclick="searchModule.OnClickCategory('0', '')">...</button>`. The `onclick` attribute of the button is highlighted in blue, indicating it is selected. The bottom of the screenshot shows the browser's developer console with a list of selected CSS classes: `html`, `body`, `div`, `div`, `div`, `div`, `div.oHotelBox`, `ul.boxList`, `ul.boxList`, `ul.boxList`, `ul.boxList`, `ul.boxList`, `li.moreBtnWrap`, and `button.moreBtn`.

```
<ul class="boxList">
  <li class="boxItem">...</li>
  <ul class="boxList">
    <li class="boxItem">...</li>
    <ul class="boxList">
      <li class="boxItem">...</li>
      <li class="moreBtnWrap">
        <button class="moreBtn" type="button" onclick="searchModule.OnClickCategory('0', '')">...</button>
      </li>
    </ul>
  </ul>
</ul>
</ul>
</div>
```

html body div div div div.oHotelBox ul.boxList ul.boxList ul.boxList ul.boxList ul.boxList li.moreBtnWrap button.moreBtn

CLASS_NAME

```
try:  
    element = WebDriverWait(driver, 10).until(  
        #지정한 한 개 요소가 올라오면 웨이트 종료  
        EC.presence_of_element_located(By.CLASS_NAME, 'oTravelBox')  
    )  
except Exception as e:
```

Error Handling

```
except Exception as e:  
    print('오류 발생', e)
```

암묵적 대기

- 요소를 찾을 특정 시간동안 DOM 폴링을 지시

```
#요소를 찾을 특정 시간 동안 DOM 폴링을 지시 ( 발견되면 진행)
driver.implicitly_wait(10)
```

게시판 진입

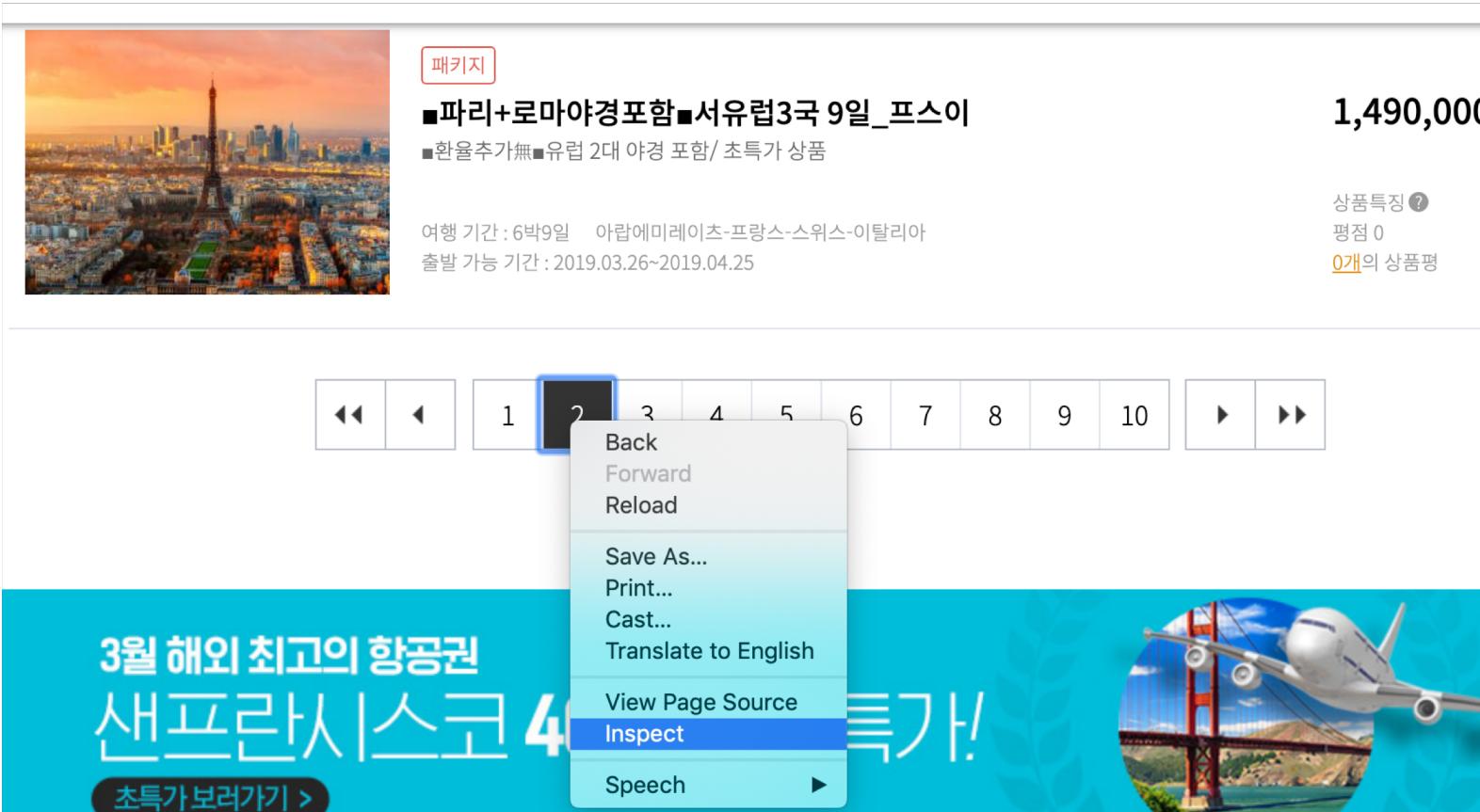
```
#더보기 눌러서 게시판 진입하기  
driver.find_element_by_css_selector('.oTravelBox>.boxlist>.moreBtnWrap>.moreBtn').click()
```

Selenium 모듈 실습 2 (4강)

게시판 스캔

게시판 스캔

- 페이지 2로 넘어간 뒤 버튼을 검사!



searchModule.OnClick

```
▼<ul>
  <li role="button" tabindex="1" onclick="searchModule.SetCategoryList(1, '')">1</li>
  <li class="active" role="button" tabindex="2">2</li>
  <li role="button" tabindex="3" onclick="searchModule.SetCategoryList(3, '')">3</li>
  <li role="button" tabindex="4" onclick="searchModule.SetCategoryList(4, '')">4</li>
  <li role="button" tabindex="5" onclick="searchModule.SetCategoryList(5, '')">5</li>
  <li role="button" tabindex="6" onclick="searchModule.SetCategoryList(6, '')">6</li>
  <li role="button" tabindex="7" onclick="searchModule.SetCategoryList(7, '')">7</li>
  <li role="button" tabindex="8" onclick="searchModule.SetCategoryList(8, '')">8</li>
  <li role="button" tabindex="9" onclick="searchModule.SetCategoryList(9, '')">9</li>
  <li role="button" tabindex="10" onclick="searchModule.SetCategoryList(10, '')">10</li>
</ul>
```

Import time

```
2 #명시적 대기 --> 특정 요소가 로드되 때 까지 대기
3 from selenium.webdriver.common.by import By ✓
4 #명시적 대기를 위해
5 from selenium.webdriver.support.ui import WebDriverWait ✓
6 from selenium.webdriver.support import expected_conditions as EC ✓
7
8 import time
9
10 #다음은 명시적 대기 예제입니다.
11 #다음은 명시적 대기 예제입니다.
```

JAVA Script 구동하기

```
#searchModule.SetCategoryList(1, '') 스크립트 실행
#16은 임시값 (게시물이 넘어갔을 때 현상 확인을 위해 숫자를 오버해서 넣음)
for page in range(1, 16)
    try:
        #자바 스크립트 구동하기
        driver.execute_script("searchModule.SetCategoryList(%s,'')" % page)
        time.sleep(2)
        print("%s 페이지 이동" % page)
```

Exception 관리

```
except Exception as e1:  
    print ( '오류', e1 )
```

THANK YOU