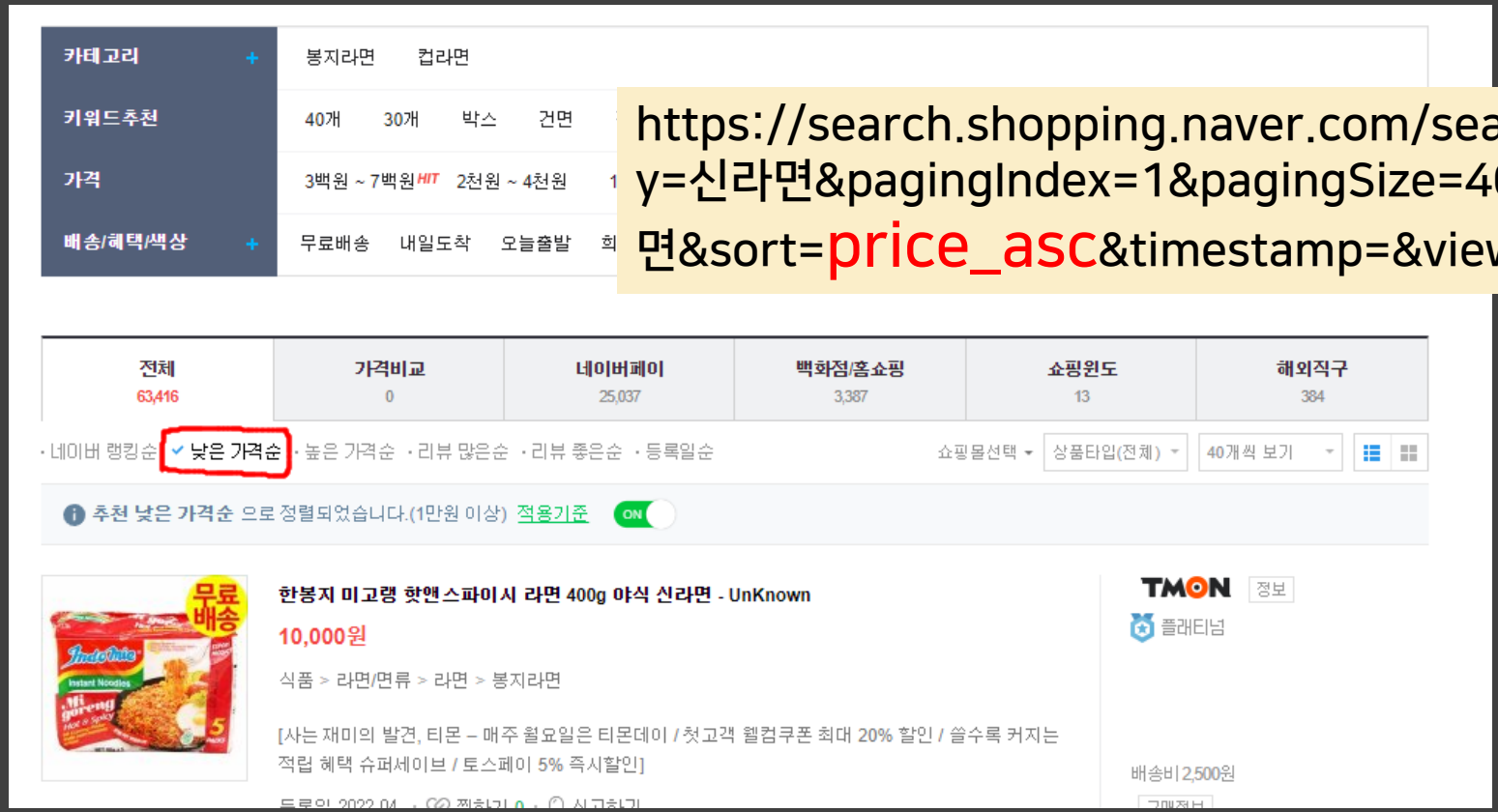
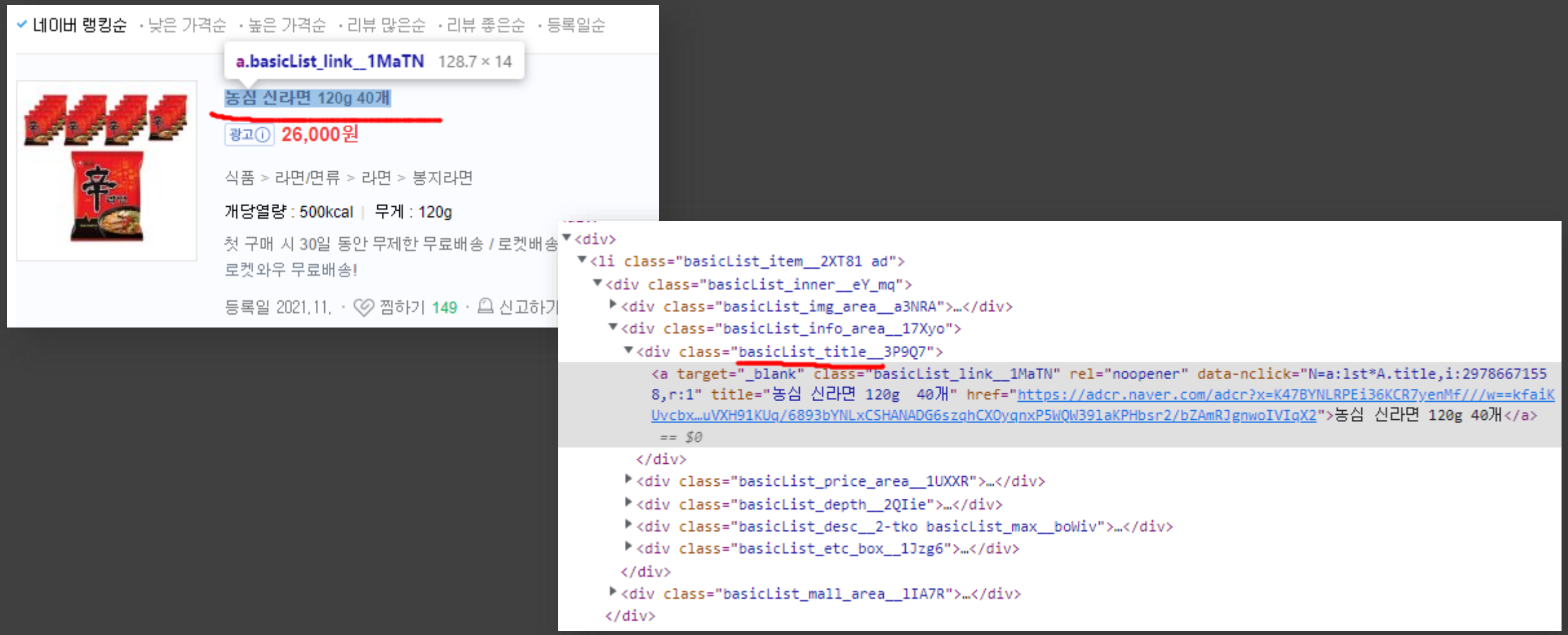
제 14 강 Web Scraping (2)

과업 #2

- 네이버 쇼핑 신라면 가격 최저가 리스트 만들기



https://search.shopping.naver.com/search/all?frm=NVSHATC&origQuery=신라면&pagingIndex=1&pagingSize=40&productSet=total&query=신라면&sort=price_asc&timestamp=&viewType=list


네이버 랭킹순 · 낮은 가격순 · 높은 가격순 · 리뷰 많은순 · 리뷰 좋은순 · 등록일순
a.basicList_link__1MaTN 128.7 × 14
농심 신라면 120g 40개
광고 26,000원
식품 > 라면/면류 > 라면 > 봉지라면
개당열량 : 500kcal | 무게 : 120g
첫 구매 시 30일 동안 무제한 무료배송 / 로켓배송
로켓와우 무료배송!
등록일 2021.11. · 찜하기 149 · 신고하기

<li class="basicList_item__2XT81 ad">

…

<a target="_blank" class="basicList_link__1MaTN" rel="noopener" data-nclick="N=a:lst*A.title,i:2978667155
8,r:1" title="농심 신라면 120g 40개" href="https://adcr.naver.com/adcr?x=K47BYNLRPEi36KCR7yenMf///w==kfaiK
Uvcbx…uVXH91KUq/6893bYNLxCSHANADG6szqhCXOyqnxP5WQW39laKPHbsr2/bZAmRJgnwoIVIqX2">농심 신라면 120g 40개
== $0

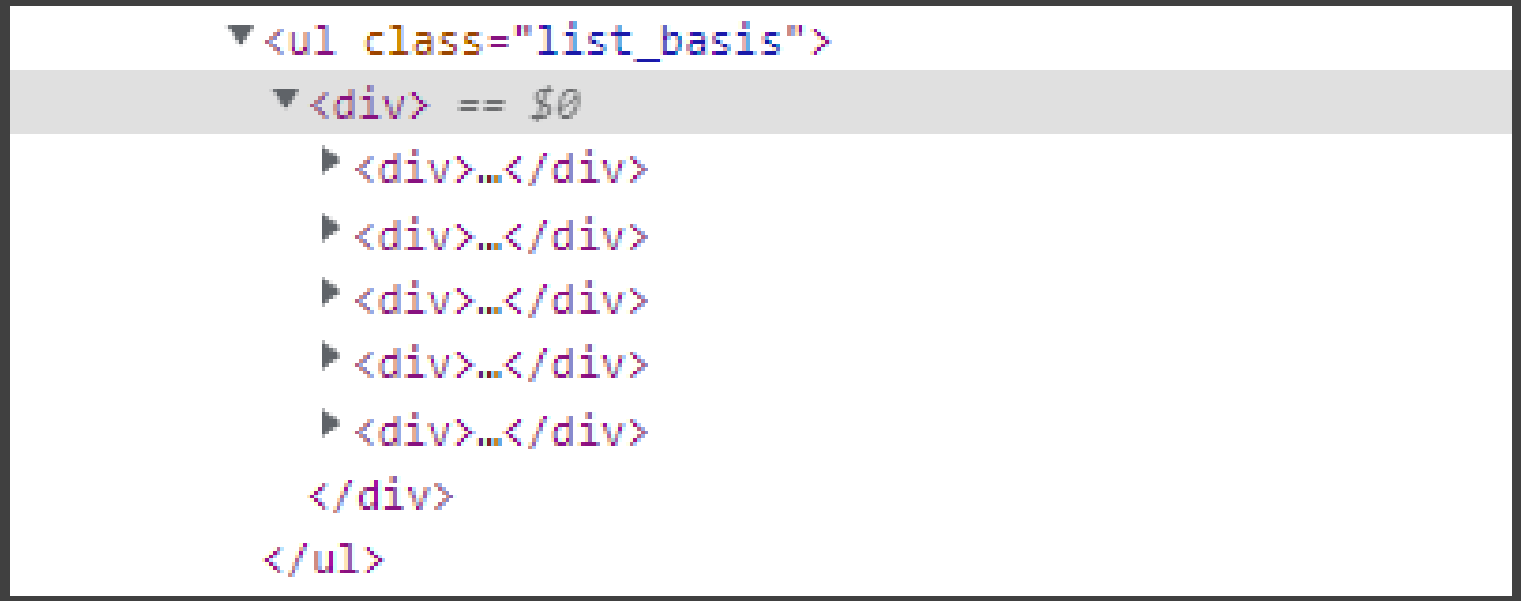
…
…
…
…

…

```
url =
f'https://search.shopping.naver.com/search/all?frm=NVSHATC&origQuery={keyword}&pagingIndex=1
&pagingSize=40&productSet=total&query={keyword}&sort=price_asc&timestamp=&viewType=list'


r = requests.get(url, headers=headers)
r.raise_for_status()

soup = BeautifulSoup(r.text, 'lxml')
elms = soup.find_all(class_=re.compile(r'^basicList_title'))

for e in elms:
    title = e.a['title']
    price = e.next_sibling.find(class_=re.compile('^price_num')).string
    print(f'{price} : {title}')
```

문제점

- JavaScript 를 통한 반응형 웹의 경우, request module 로 제대로 읽어낼 수 없음.



Selenium

- 웹 자동화 테스트 도구.
  - 각종 사용자의 입력(클릭, 텍스트입력, 리스트 선택, ….)을 프로그램으로 처리.

- 인터랙티브 웹 싸이트 크롤링에 활용.

- 설치
  - selenium 라이브러리 설치 - pip install selenium
  - 자동화를 위한 브라우저 프로그램 설치 – working folder 아래에 복사.
    - Chrome driver – PC 에 설치되어 있는 chrome 의 버전에 맞는 드라이버 설치 필요.
    - https://chromedriver.chromium.org/downloads

Playwright

- Microsoft 사가 개발한 웹 테스팅 및 자동화 프레임워크

- Selenium과의 비교 우위
  - 자동 대기 – 엘리먼트들이 준비될 때까지 대기
  - 테스트 레코딩 & 재생
  - 여러 개의 브라우저에서 동시 실행

- 설치 (2단계)
  - pip install pytest-playwright
  - playwright install

네이버 홈

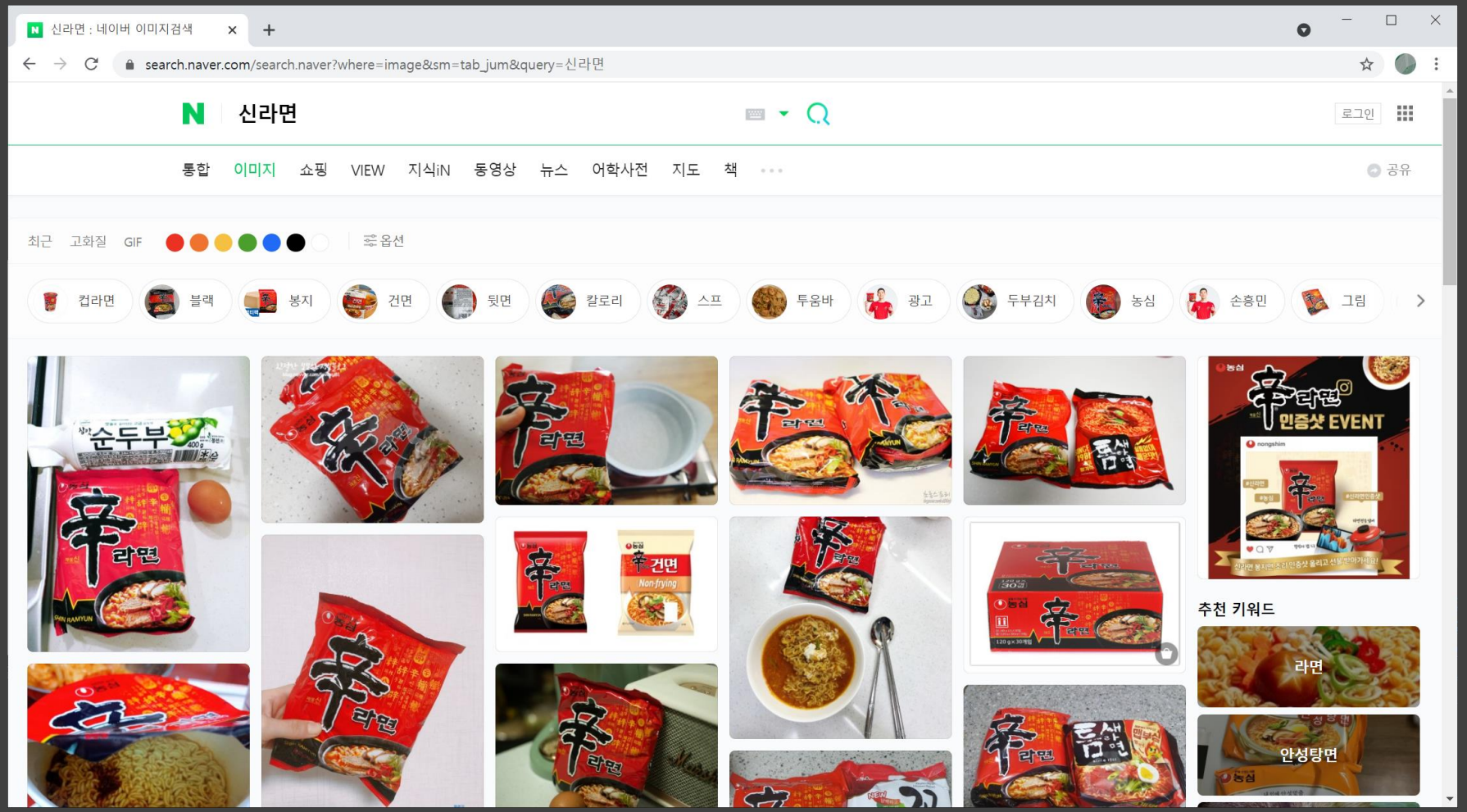
```
from playwright.sync_api import sync_playwright

p = sync_playwright().start()
browser = p.chromium.launch(headless=False)
page = browser.new_page()
page.goto("https://www.naver.com")
print(page.title())
browser.close()
p.stop()
```

```python
from playwright.sync_api import sync_playwright
from bs4 import BeautifulSoup
import re
import time

keyword = '신라면'
url =
'https://search.shopping.naver.com/search/all?frm=NVSHATC&origQuery='+keyword+'&pagingIndex=1&pagingSize=10&productSet=total
&query='+keyword+'&sort=price_asc&timestamp=&viewType=list'

p = sync_playwright().start()
browser = p.chromium.launch(headless=False).new_context(
    user_agent='Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.0.0
Safari/537.36',
    viewport={'width': 1920, 'height': 2048}
)
page = browser.new_page()
page.goto(url)

src_size = 0
while src_size < len(page.content()):
    src_size = len(page.content())
    page.keyboard.press('End')
    time.sleep(1)

soup = BeautifulSoup(page.content(), 'lxml')
elms = soup.find_all(class_=re.compile(r'^basicList_title'))

for e in elms:
    title = e.a['title']
    price = e.next_sibling.find(class_=re.compile('^price_num')).text
    print(f'{price=} : {title=}')

browser.close()
p.stop()
```

현

과업 #3.

- 신라면 이미지 자동 다운로드



```
01a4ecce5c96b13d3230451e&g=663960999185414" class="link_thumb _imageBo
x _infoBox" title="[라면추천] 모두가 알고있고 모두가 맛있어하는 신기한 얼
큰라면 신라면 !!!" role="button" aria-pressed="false">
  <img src="https://search.pstatic.net/common/?src=http%3A%2F%2Fblogfi
  les.naver.ne…fIC09z5fJQ5Aqcg.JPEG.liberal13803%2F20220506%25A3%25DF0
  03625.jpg&type=a340" class="_image _listImage" data-org-width="340"
  data-org-height="340" alt="[라면추천] 모두가 알고있고 모두가 맛있어하는
  신기한 얼큰라면 신라면 !!!" style="height: 232px; opacity: 1;"> == $0
  <i class="spimg ico_selected"></i>
  ::after
```

```
keyword = '신라면'
url = 'https://search.naver.com/search.naver?where=image&sm=tab_jum&query=' + keyword

p = sync_playwright().start()
browser = p.chromium.launch(headless=False).new_context(
    user_agent='Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/109.0.0.0 Safari/537.36'
)
page = browser.new_page()
page.goto(url)

soup = BeautifulSoup(page.content(), 'lxml')

elms = soup.select('img._image._listImage')
for n, e in enumerate(elms):
    print(n, e['src'], e)
```

로딩 늦어지는 리소스에 대한 처리

```
for n, e in enumerate(elms):
    caption = e['alt']
    image_url = e.get('data-lazy-src', e['src'])
    print(f'{n} {caption} {image_url}')
```

requests.get 를 이용한 다운로드

```
def download_image(image_url):
    r = requests.get(image_url)
    r.raise_for_status()
    if r.headers['Content-Type']  == 'image/jpeg':
        fname = f'img_{uuid.uuid4().int}.jpg'
        with open(fname, 'wb') as wf:
            print(f'Downloading {fname} .....')
            wf.write(r.content)


for e in elms:
    image_url = e.get('data-lazy-src', e['src'])
    download_image(image_url)
```