

08 웹 크롤링



경고 : 본 강의자료는 연세대학교 학생들을 위해 수업 목적으로 제작.게시된 것이므로, 수업목적 이외의 용도로 사용할 수 없으며, 다른 사람과 공유할 수 없습니다.
위반에 따른 법적 책임은 행위자 본인에게 있습니다.



연세대학교
YONSEI MIRAE
CAMPUS

- ◆ **requests**와 **BeautifulSoup**을 활용하여 웹 크롤링을 하는 방법
- ◆ **selenium**을 활용하여 웹 크롤링을 하는 방법

8.1 웹 크롤링 개요

8.2 웹 데이터 자동 수집

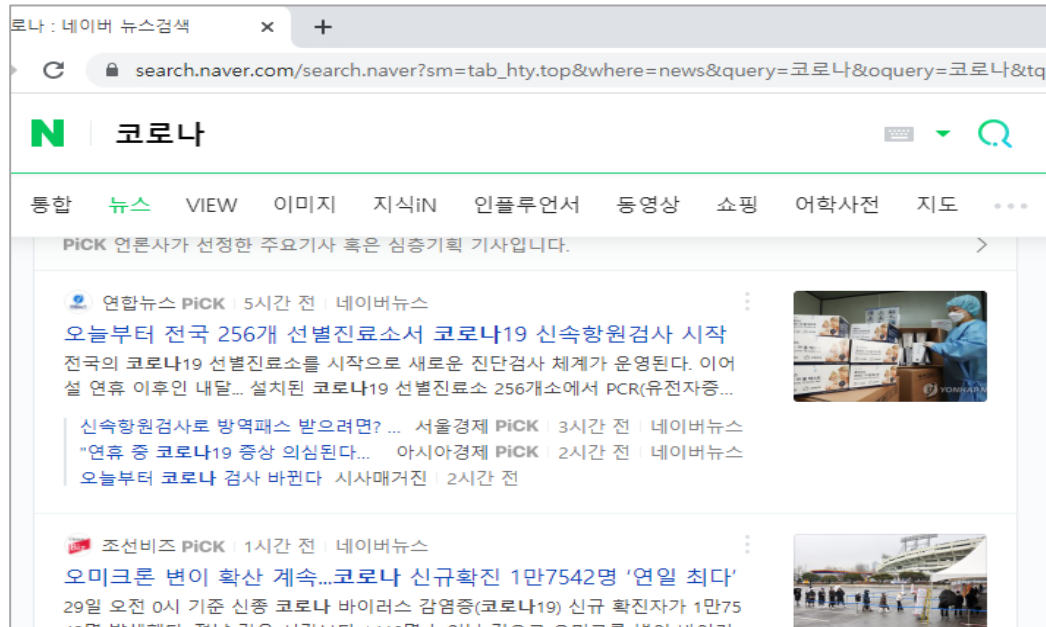
8.3 웹 브라우저 제어

웹 크롤링 절차 및 주요 패키지



● 웹 크롤링이란?

- 웹 크롤링은 웹 페이지 내에 특정 정보를 자동화된 방법으로 수집하는 활동
- 아래 그림과 같이 특정 회사와 관련된 뉴스를 크롤링해서 결과를 엑셀로 저장
- 이 외에도 일일 환율 정보, 시황 정보 수집 등의 업무에 웹 크롤링을 활용



2022-01-29 네이버 뉴스 크롤링 결과_검색어 코로나 - Excel							
파일 홈 삽입 레이아웃 수식 데이터 검토 보기 Q 수행할 작업을 알려주세요							
크롤링 날짜							
A	B	C	D	E	F	G	H
1	크롤링 날짜	제목	링크				
2	2022-01-29	오늘부터 전국 256개 선별진료소서 코로나19 신속항원검사 시작	http://yna.kr/AKR20220128185500530?did=1195m				
3	2022-01-29	오미크론 변이 확산 계속...코로나 신규확진 1만7542명 '연일 최다'	https://biz.chosun.com/topics/topics_social/2022/01/29/MWCQWPQ2G				
4	2022-01-29	유재석, 코로나19 음성... "스케줄 정상 진행"	http://www.newsis.com/view/?id=NISX20220129_0001742456&clD=106				
5	2022-01-29	설 연휴 첫 날, 코로나19 신규확진 1만7542명.. "또" 역대 최다	http://www.fnnews.com/news/202201291002240932				
6	2022-01-29	코로나19 국내유입 2년...누적 확진자 80만명 넘겨	https://news.imaail.com/page/view/2022012910003117060				
7	2022-01-29	코로나 검사, 오늘부터 바뀐다...PCR-신속항원검사 병행	http://www.newsis.com/view/?id=NISX20220128_0001742246&clD=102				
8	2022-01-29	[브리핑] 정부 "3일부터 지정 동네병원서 코로나 검사·처방·재택치료"	https://news.sbs.co.kr/news/endPage.do?news_id=N1006622078&pln1				
9	2022-01-29	직접 코로나 검사	http://yna.kr/PYH20220129016700013?did=1196m				
10	2022-01-29	2월3일부터 동네병원에서도 코로나19 검사·치료 받는다	https://www.khan.co.kr/national/health-welfare/article/202201281104				
11	2022-01-29	코로나 증상에도 제주도 여행한 '감남모녀'...역대 손배소 승소	http://news.mt.co.kr/mtview.php?no=2022012817353794023				
12	2022-01-29	3일부터 전국 호흡기전담클리닉서 코로나 검사·치료까지(종합)	http://yna.kr/AKR20220128074151530?did=1195m				
13	2022-01-29	'부스터샷' 맞은 조세호, 코로나 확진...유재석 또 PCR 검사	https://www.chosun.com/national/welfare-medical/2022/01/28/H3LPC				

● 웹 크롤링 절차

- 웹 크롤링은 **requests, BeautifulSoup, selenium** 패키지 및 **크롬 개발자 도구**를 활용
- HTML(Hyper Text Markup Language) 태그 및 CSS(Cascading Style Sheets)에 대한 기초 지식 필요

1) HTML 문서 요청 및 추출하기

requests 또는 **selenium** 패키지를 활용하여 HTML 문서 추출

2) HTML 문서 분석하기

크롬 개발자 도구, **HTML** 태그, **CSS** 선택자를 활용해 분석

3) 원하는 데이터 추출하기

requests, selenium, BeautifulSoup 활용, 원하는 위치의 데이터 추출

4) 찾은 데이터 가공 및 저장하기

추출한 데이터를 가공하고 엑셀 또는 텍스트 파일로 저장

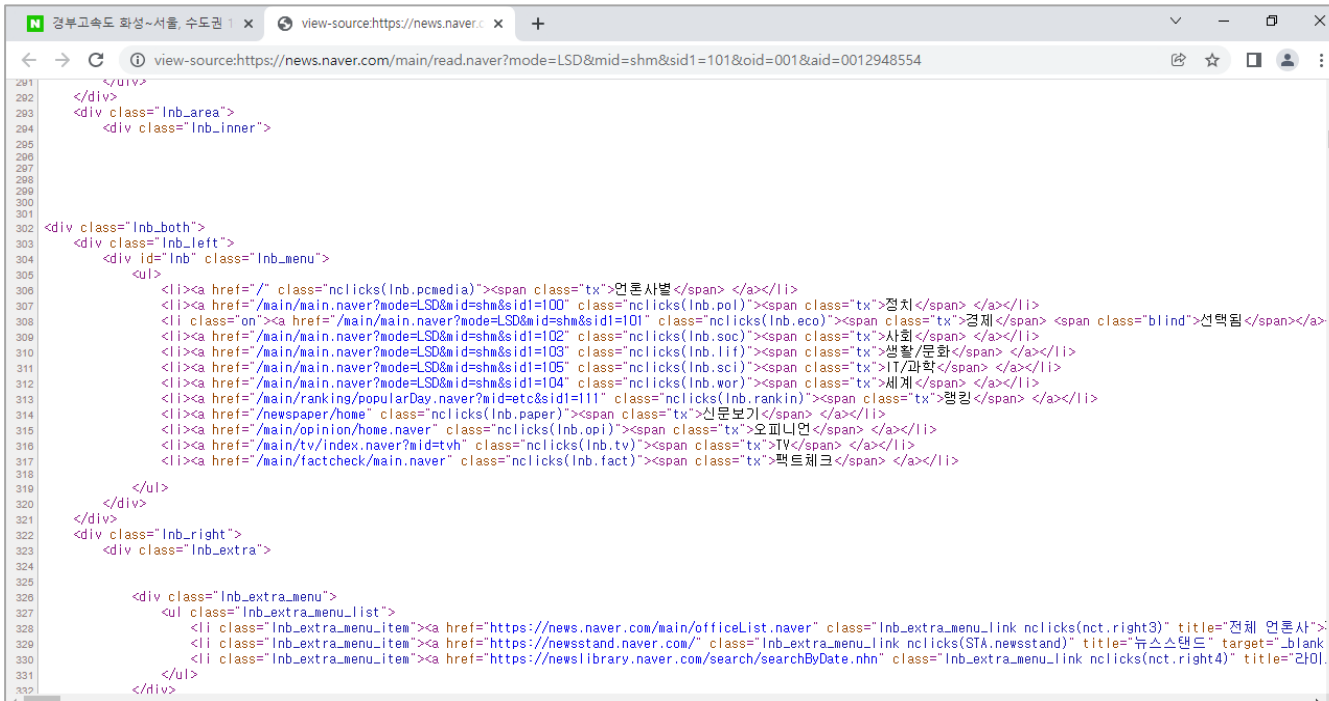
● 파이썬 웹 크롤링 패키지

- **requests**와 **selenium**은 원하는 웹 페이지의 **HTML 소스**를 **추출**하는 역할
- **BeautifulSoup**은 추출한 HTML을 **분석**하는 데 사용

패키지	특징
requests	브라우저 없이 URL에 접속, 결과값(HTML)을 추출함 <ul style="list-style-type: none">- 웹 브라우저를 실행하지 않고 특정 URL에 요청(request)을 보낼 수 있음- 응답(response)된 데이터의 헤더(Header), 바디(HTML) 등의 정보 확인- 빠르고 안정적이지만 동적 웹페이지를 크롤링하기 위해서는 많은 분석이 필요
selenium	브라우저를 자동으로 실행해 URL에 접속, 결과값(HTML)을 추출함 <ul style="list-style-type: none">- 시뮬레이션 하듯이 웹 브라우저를 직접 실행해 URL에 접속할 수 있음- 응답된 데이터의 헤더(Header), 바디(HTML) 등의 정보를 확인할 수 있음- 느리고 불안정하지만 동적 웹페이지 크롤링이 쉽고 초보자가 사용하기 좋음
BeautifulSoup	requests와 selenium으로 받아온 내용(HTML)을 분석 <ul style="list-style-type: none">- requests나 selenium을 통해 얻은 HTML 데이터를 파싱(parsing)해 객체로 구조화- HTML 수집은 앞선 두 모듈이 담당하지만, 분석은 BeautifulSoup이 담당- HTML 태그와 CSS 속성을 이용해 원하는 정보를 찾을 수 있는 기능 제공

● HTML이란?

- HTML(HyperText Markup Language)은 웹 페이지의 모습을 표현하기 위한 마크업 언어
- 마크업 언어는 태그(Tag)를 사용해 문서나 데이터의 구조를 표현하므로, HTML 페이지는 태그와 실제 데이터로 구성되어 있는 문서라고 할 수 있음
- 아래 그림(네이버 뉴스 사이트의 HTML 소스 코드)

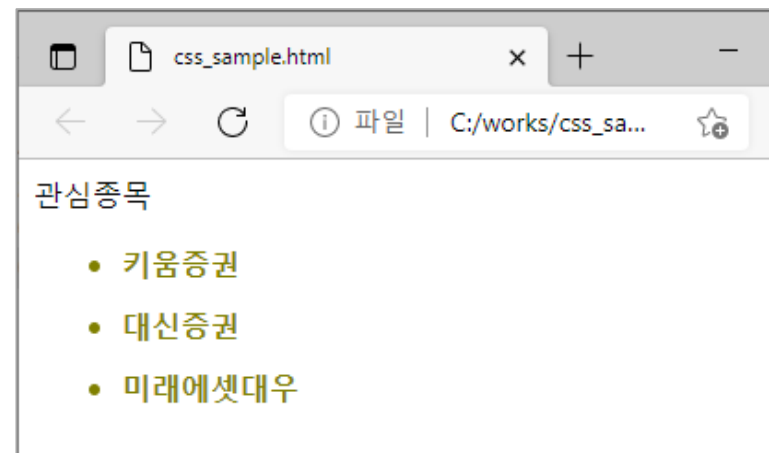


```
291 </div>
292 </div>
293 <div class="lnb_area">
294   <div class="lnb_inner">
295
296
297
298
299
300
301
302 <div class="lnb_both">
303   <div class="lnb_left">
304     <div id="lnb" class="lnb_menu">
305       <ul>
306         <li><a href="/" class="nclicks(lnb.pcmedia)"><span class="tx">연혼사별</span> </a></li>
307         <li><a href="/main/main.naver?mode=LSD&mid=shm&sid1=100" class="nclicks(lnb.pol)"><span class="tx">정치</span> </a></li>
308         <li class="on"><a href="/main/main.naver?mode=LSD&mid=shm&sid1=101" class="nclicks(lnb.eco)"><span class="tx">경제</span> </a></li>
309         <li><a href="/main/main.naver?mode=LSD&mid=shm&sid1=102" class="nclicks(lnb.soc)"><span class="tx">사회</span> </a></li>
310         <li><a href="/main/main.naver?mode=LSD&mid=shm&sid1=103" class="nclicks(lnb.life)"><span class="tx">생활/문화</span> </a></li>
311         <li><a href="/main/main.naver?mode=LSD&mid=shm&sid1=105" class="nclicks(lnb.sci)"><span class="tx">IT/과학</span> </a></li>
312         <li><a href="/main/main.naver?mode=LSD&mid=shm&sid1=104" class="nclicks(lnb.wor)"><span class="tx">세계</span> </a></li>
313         <li><a href="/main/ranking/popularDay.naver?mid=etc&sid1=111" class="nclicks(lnb.rankin)"><span class="tx">랭킹</span> </a></li>
314         <li><a href="/newspaper/home" class="nclicks(lnb.paper)"><span class="tx">신문보기</span> </a></li>
315         <li><a href="/main/opinion/home.naver" class="nclicks(lnb.opi)"><span class="tx">오피니언</span> </a></li>
316         <li><a href="/main/tv/index.naver?mid=tvh" class="nclicks(lnb.tv)"><span class="tx">TV</span> </a></li>
317         <li><a href="/main/factcheck/main.naver" class="nclicks(lnb.fact)"><span class="tx">팩트체크</span> </a></li>
318       </ul>
319     </div>
320   </div>
321   <div class="lnb_right">
322     <div class="lnb_extra">
323
324
325
326     <div class="lnb_extra_menu">
327       <ul class="lnb_extra_menu_list">
328         <li class="lnb_extra_menu_item"><a href="https://news.naver.com/main/officeList.naver" class="lnb_extra_menu_link nclicks(nct.right3)" title="전체 언론사">
329         <li class="lnb_extra_menu_item"><a href="https://newsstand.naver.com/" class="lnb_extra_menu_link nclicks(STA.newsstand)" title="뉴스스탠드" target="_blank"
330         <li class="lnb_extra_menu_item"><a href="https://newslibrary.naver.com/search/searchByDate.nhn" class="lnb_extra_menu_link nclicks(nct.right4)" title="라이
331       </ul>
332     </div>
333   </div>
```

● CSS란?

- CSS(Cascading Style Sheets)는 HTML로 작성된 문서를 웹 브라우저에 표현할 때 서식을 정해주는 언어
- 웹 크롤링 시 HTML 문서에 있는 특정 데이터에 접근하기 위해 HTML 태그와 함께 CSS를 자주 활용
- 일반적으로 HTML 문서 내 **<style>** 태그 안에서 HTML 각 요소의 **배경색, 폰트** 등을 정의하는 형태로 사용
- 다음은 HTML 문서 내에서 **<style>** 태그로 서식을 지정한 CSS 사용 예제

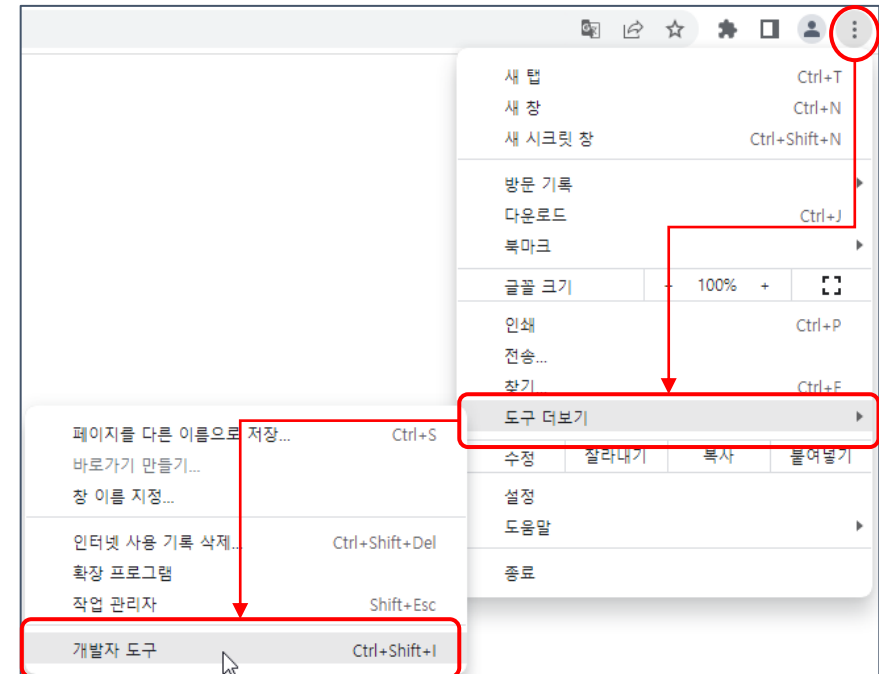
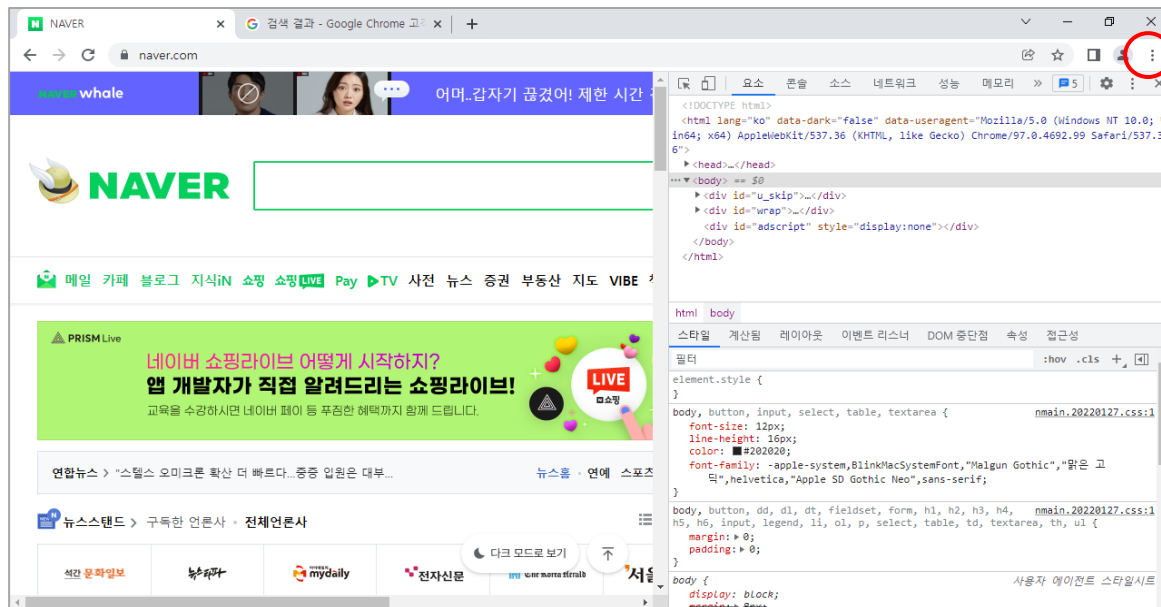
```
관심종목<br>
<ul>
  <li>키움증권</li>
  <li>대신증권</li>
  <li>미래에셋대우</li>
</ul>
<style>
  li { width: 120px; margin: 8px; padding-bottom: 4px;
        font-weight: bold; color: olive }
</style>
```



크롬 개발자 도구로 웹 구조 파악하기



● 크롬 개발자 도구 활성화

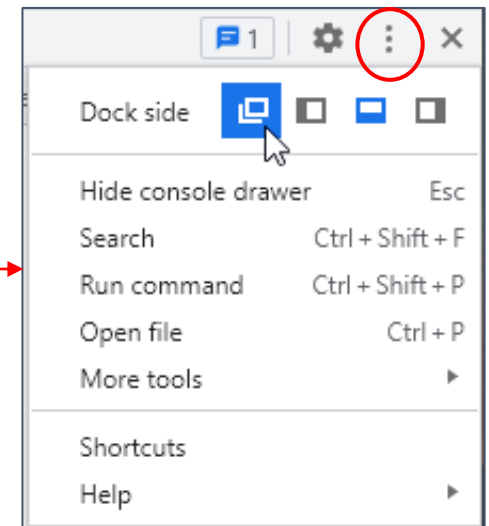
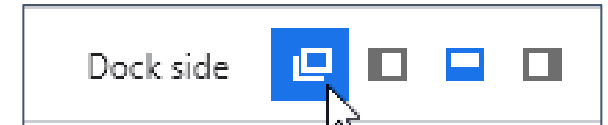
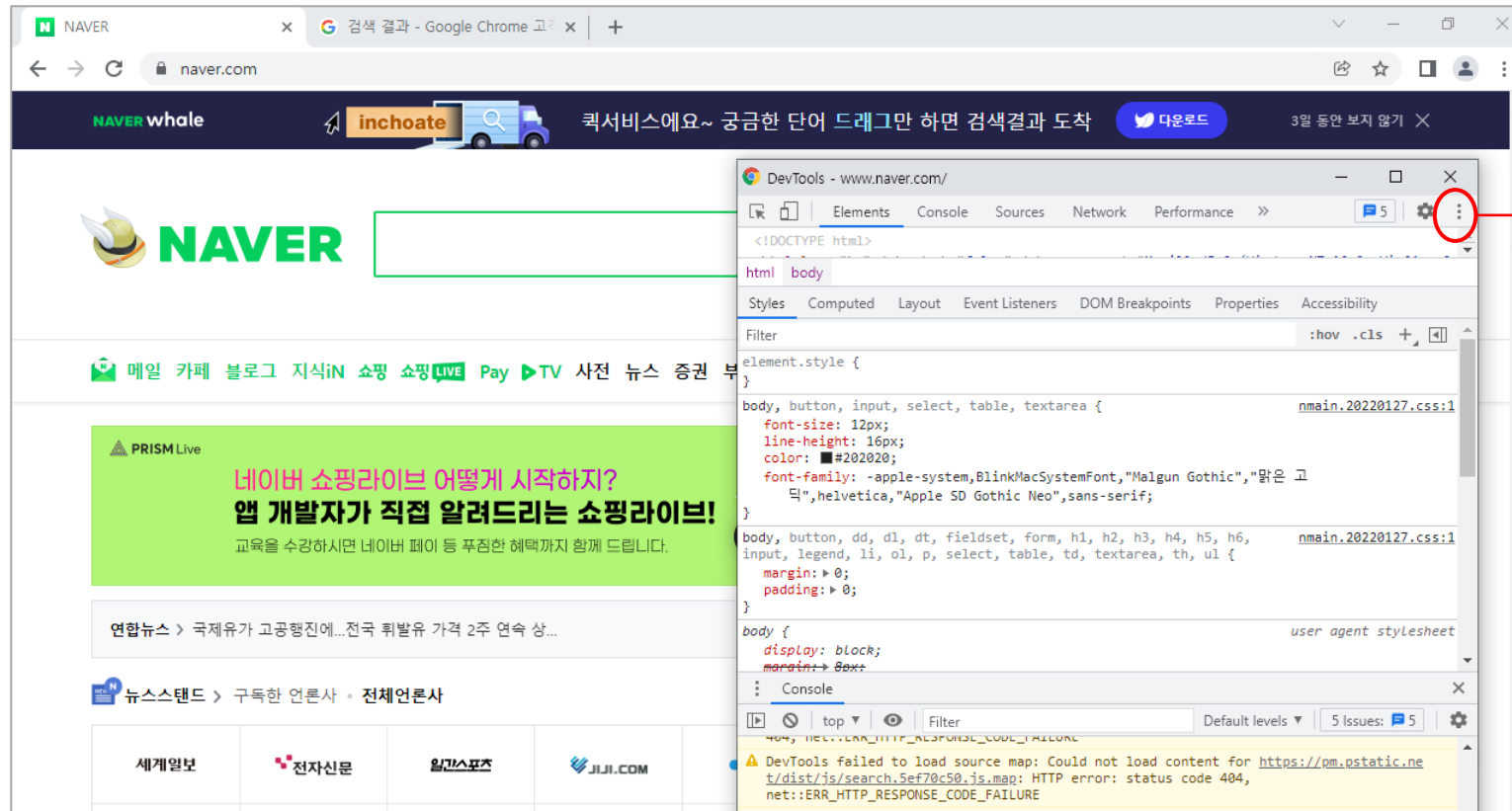
- 크롬(Chrome) 개발자 도구는 웹 페이지의 구조를 분석하기 위한 툴로, **크롬에 내장되어 있는** 기능임
- HTML 뿐만 아니라, 자바스크립트, CSS로 구성된 웹 페이지의 구조를 쉽게 파악할 수 있기 때문에 웹 페이지의 오류를 수정하거나 기능을 개선할 때 주로 사용
- 원하는 정보가 있는 특정 태그나 CSS 셀렉터를 쉽게 찾기 때문에 웹 크롤링할 때도 유용하게 사용
- 크롬 실행 후 네이버 홈페이지(www.naver.com)에 접속 → 크롬의 우측상단에 있는 [Chrome 맞춤설정 및 제어] 아이콘 **⋮** 을 클릭 → [도구 더보기] - [개발자 도구] 메뉴를 선택하여 개발자 도구를 실행 또는 **[F12] 키**



크롬 개발자 도구로 웹 구조 파악하기

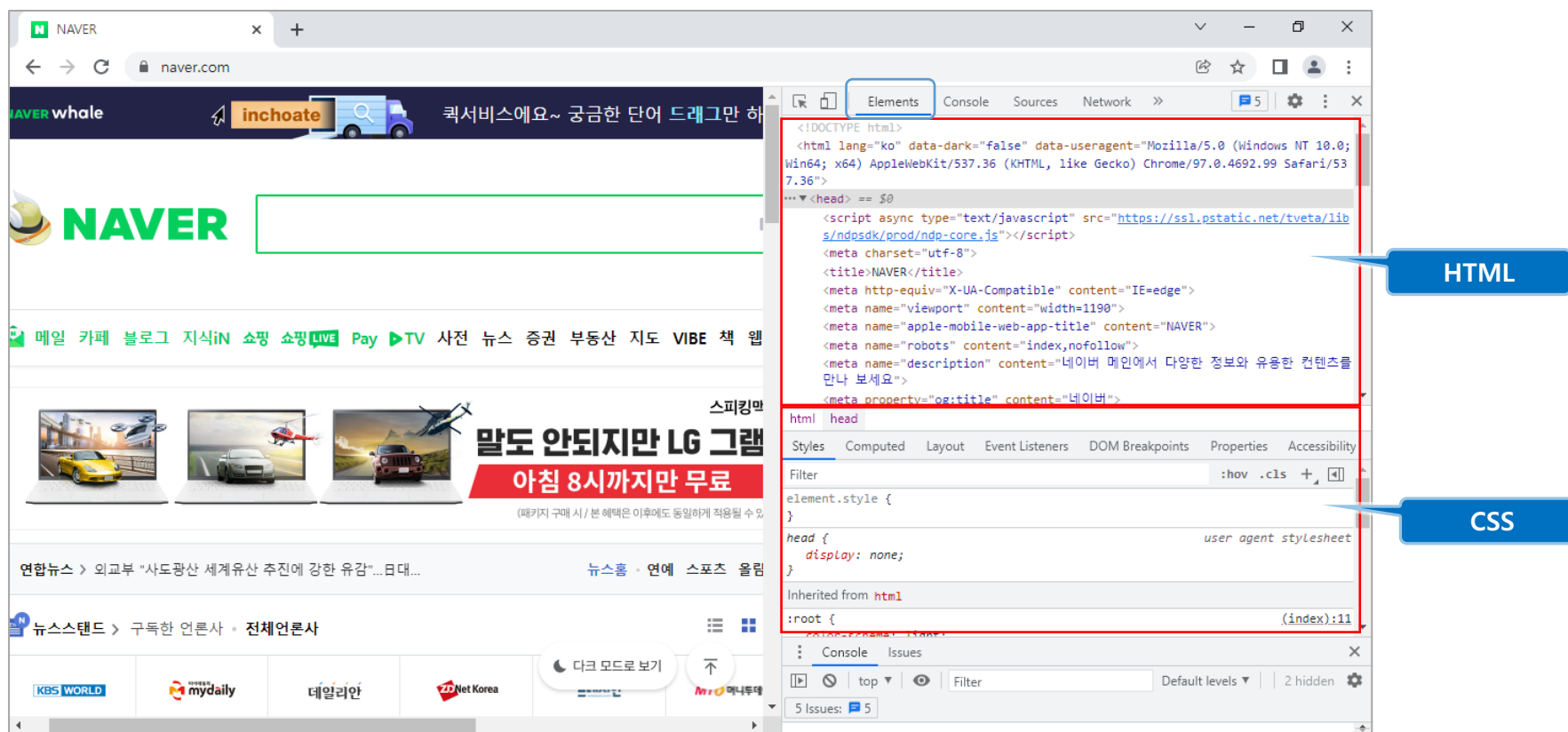
● 크롬 개발자 도구 창 배치

- 개발자 도구의 우측 상단에서 [Customize and control DevTool]  아이콘을 클릭하면 [Dock side] 옵션에서 개발자 도구의 위치를 지정할 수 있음. 또한, 개발자 도구를 분리할 수도 있음
- [Dock side]에서 Undock into separate window  를 선택하면 개발자 도구 창이 별도의 화면으로 분리



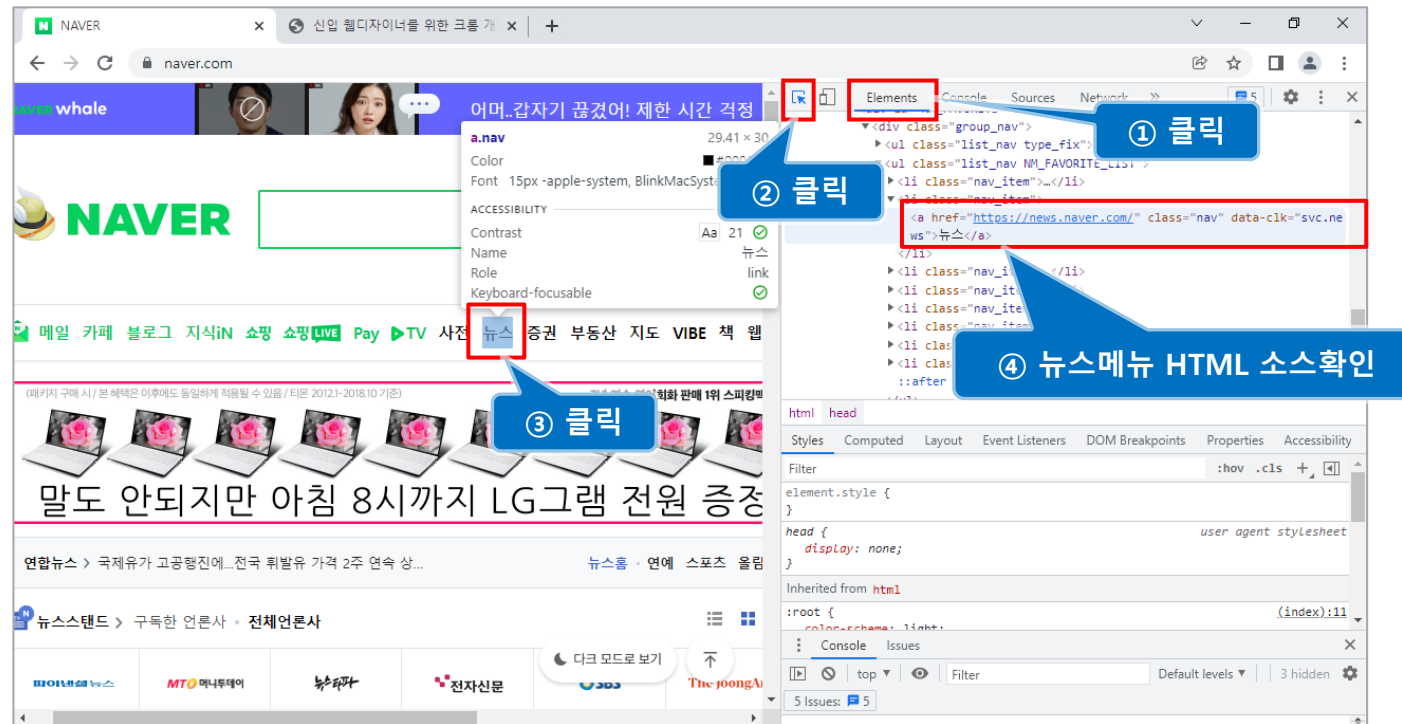
● HTML 소스 영역과 CSS 영역

- 개발자 도구 창은 웹페이지의 HTML 소스를 볼 수 있는 영역과 CSS 구성을 파악할 수 있는 영역이 있음
- 마우스로 두 영역간 창 크기 조절 가능
- 웹 크롤링시 필요한 태그나 CSS selector 들은 HTML 소스에서 찾을 수 있음



● [Elements] 탭

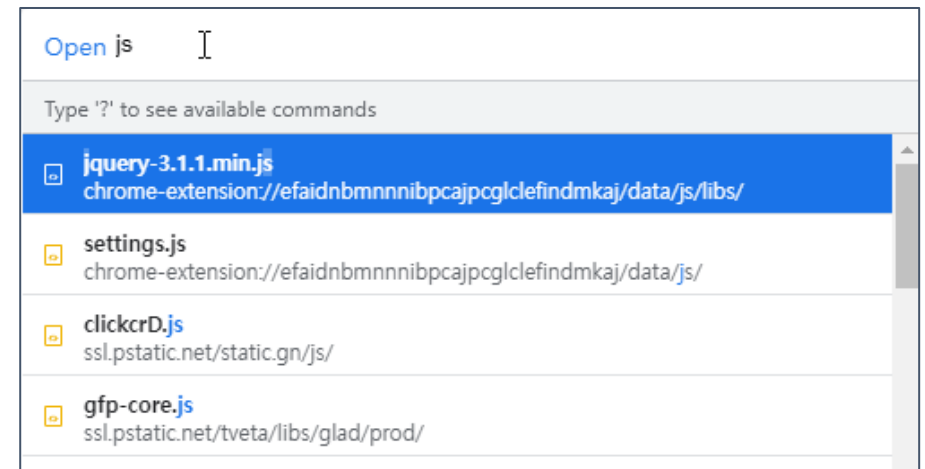
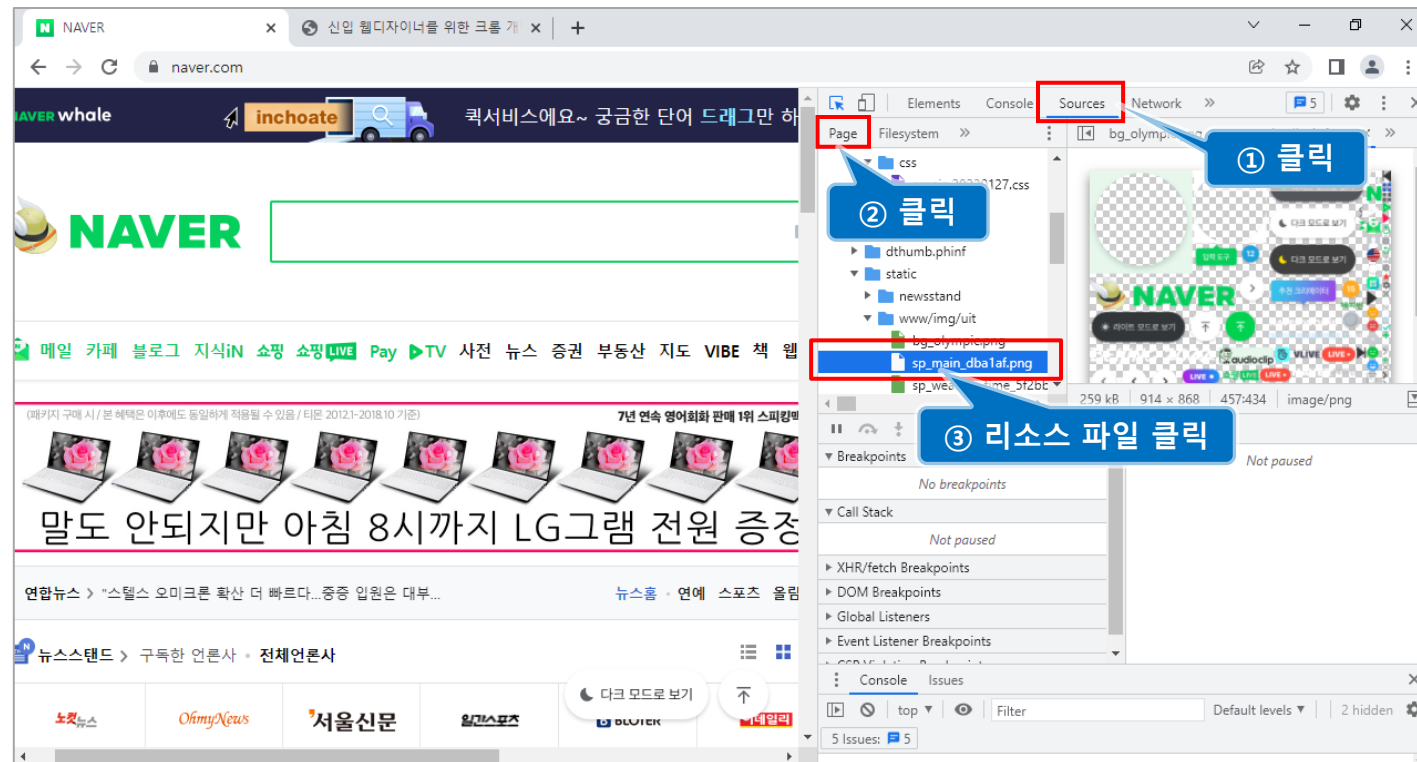
- 개발자 도구 창에서 ① [Elements] 탭을 클릭하면 웹 페이지에 대응되는 소스 코드가 보임
- ② Inspector 아이콘을 클릭해서 활성화시킨 후, ③ 웹 페이지의 특정 위치에 마우스를 올리면 ④ 해당되는 HTML 소스를 확인 가능
- 반대로 [Elements] 탭의 소스를 선택하면 웹 페이지 영역에서 해당되는 위치가 표시됨
- 아래 그림처럼 네이버의 뉴스 메뉴에 마우스를 가져 갔을 때 개발자 도구 창에서 해당 HTML 소스를 볼 수 있음



크롬 개발자 도구로 웹 구조 파악하기

● [Sources] 탭

- [Sources] 탭에서는 웹 페이지에 포함된 모든 리소스 파일을 확인할 수 있음
- 왼쪽 도메인별 리소스 트리를 펼쳐서 특정 리소스 파일을 더블클릭하면 텍스트 형태는 소스, 이미지 파일은 이미지 자체를 보여줌
- 도메인별로 리소스를 표시하기 때문에 원하는 리소스를 찾기 어려우면 **[Ctrl]+[P]**(Open File)를 눌러 입력란에 검색어 입력



HTML 소스 가져오기

➤ 웹 크롤링 절차 이해

- 웹 크롤링 시 제일 처음 할 일은 웹 서버로부터 **특정 웹 페이지를 받아오는 것**
- HTML 페이지, 이미지, 문서 파일 등 필요한 리소스를 받아 오기 위해서는 웹 브라우저를 통해 해당 리소스의 위치인 **URL을 웹 서버에 요청**하고, 웹 서버는 웹 브라우저에게 **요청한 리소스를 보내줌**
- 보내준 리소스의 **Content-Type**이 **HTML** 형태이면 웹 브라우저가 랜더링해서 사용자에게 보여주고, 텍스트 형식이 아닌 **이진 파일은 로컬 PC에 다운로드할 수 있게 해줌**
- ✓ **Content-Type** : 웹 서버와 클라이언트(웹 브라우저) 간 정보를 주고 받을 때 사용하는 자료 형식
 - HTML, Text, Application, Multipart, Audio 등이 있음

HTML 소스 가져오기

● requests 모듈 설치하기

➤ request 개요

- 웹 크롤링을 하려면 파이썬 모듈을 사용해서 웹 서버와 상호작용을 해야 함
- **requests** 는 파이썬 모듈 만으로 원하는 웹 페이지에 HTML 요청을 보내고 응답 데이터를 수집할 수 있도록 만든 패키지
- 웹 페이지의 HTML/JSON 뿐만 아니라 이미지, 음성, PDF 등 파일 데이터 수집 기능까지 제공
- HTTP request 에 필요한 기능 지원(**get/post** 방식 선택, **parameter/form data/cookie 보내기** 등)
- 웹 브라우저 없이 구동하므로 코드가 짧고 속도가 매우 빠르며, 안정적인 크롤링이 가능
- 랜더링된 웹 페이지 화면이 보이지 않으므로 크롬 개발자 도구를 통한 사전 분석이 중요

➤ request 모듈 설치하기

- Python과 함께 배포되지 않으므로 별도로 설치해야 한다.
- 주피터 노트북에서 아래 **pip** 명령어 입력으로 설치

```
!pip install requests
```

● HTML 소스 가져오기

1. GET 방식

- 특정 사이트에 HTML 페이지 등의 자원을 요청할 때 **파라미터를 URL 뒤에 붙이는 방식**
- GET은 주로 웹 브라우저가 **웹 서버에 데이터를 요청할 때** 사용

2. POST 방식

- **URL에 붙이지 않고 별도의 데이터로 전달하는 방식**
- 웹 브라우저가 **웹 서버에 데이터를 전달하기 위해** 사용

● HTML 소스 가져오기

➤ GET 방식

- requests.get() 함수에 파라미터로 URL과 verify를 사용
- verify=False
 - ✓ SSL(secure Sockets Layer) 인증서 오류 문제가 발생해도 가져오도록
- 가져온 내용을 출력하는 방법
 - ✓ print(res.content) : HTML 형식을 보여주는데, 텍스트, 이미지, zip 파일 등 이진 데이터도 볼 수 있음
 - ✓ print(res.text) : 한글 출력 가능. 텍스트 위주의 웹 페이지일 경우 유용

```
import requests
res = requests.get("http://www.naver.com", verify = False)
print(res.content)
#print(res.text)
```

```
# requests 모듈 불러오기
# get으로 naver 홈페이지 가져오기
# 내용 출력
```

```
b'<!doctype html>
<html lang="ko" data-dark="false"> <
head> <meta charset="utf-8"> <title>NAVER</title> <meta http-equiv="X-UA-Compatib
le" content="IE=edge"> <meta name="viewport" content="width=1190"> <meta name="ap
ple-mobile-web-app-title" content="NAVER"/> <meta name="robots" content="index,no
follow"/> <meta name="description" content="네이버 메인에서 다양한 정보와 유용한 컨텐
츠를 만나 보세요"/> <meta property="og:title" content="네이버"> <meta property="o
g:url" content="https://www.naver.com/"> <meta property="og:image" content="http
s://s.pstatic.net/static/www/mobile/edit/2016/0705/mobile_212852414260.png"> <met
a property="og:description" content="네이버 메인에서 다양한 정보와 유용한 컨텐츠
를 만나 보세요"/> <meta name="twitter:card" content="summary"> <meta name="twitte
r:title" content=""> <meta name="twitter:url" content="https://www.naver.com/"> <
```

```
<!doctype html>
<html lang="ko" data-dark="false"> <head>
<meta charset="utf-8"> <title>NAVER</title> <meta http-equiv="X-UA-Compatible"
content="IE=edge"> <meta name="viewport" content="width=1190"> <meta name="apple-
mobile-web-app-title" content="NAVER"/> <meta name="robots" content="index,nofoll
ow"/> <meta name="description" content="네이버 메인에서 다양한 정보와 유용한 컨텐
츠를 만나 보세요"/> <meta property="og:title" content="네이버"> <meta property="o
g:url" content="https://www.naver.com/"> <meta property="og:image" content="http
s://s.pstatic.net/static/www/mobile/edit/2016/0705/mobile_212852414260.png"> <met
a property="og:description" content="네이버 메인에서 다양한 정보와 유용한 컨텐츠
를 만나 보세요"/> <meta name="twitter:card" content="summary"> <meta name="twitte
r:title" content=""> <meta name="twitter:url" content="https://www.naver.com/"> <
```


● HTML 소스 가져오기

➤ GET 방식으로 간단하게 파라미터를 전달하는 방법

- `requests.get()` 에 파라미터 속성을 사용하지 않고 url 에 **?** 및 **=** 기호를 사용해 파라미터를 바로 붙여서 요청해도 됨

```
res = requests.get("https://finance.naver.com/item/sise.nhn?code=005930", verify = False)
print(res.text)
```

출력 결과

```
<meta property="og:title" content="네이버 금융"/>
<meta property="og:image" content="https://ssl.pstatic.net/static/m/stock/im/2016/08/og_stock-200.png"/>
<meta property="og:url" content="https://finance.naver.com"/>
<meta property="og:description" content="국내 해외 증시 지수, 시장지표, 펀드, 뉴스, 증권사 리서치 등 제공"/>
```

```
<meta property="og:type" content="article"/>
<meta property="og:article:thumbnailUrl" content=""/>
<meta property="og:article:author" content="네이버금융"/>
<meta property="og:article:author:url" content="http://FINANCE.NAVER.COM"/>
```

```
<link rel='stylesheet' type='text/css' href='https://ssl.pstatic.net/imgstock/static.pc/20210803162128/css/finance_header.css'>
```

HTML 소스 가져오기

● HTML 소스 가져오기

➤ POST 방식

- 특정 사이트에 HTML 페이지 등의 자원을 요청할 때 파라미터를 URL에 붙이지 않고 별도의 데이터로 전달하는 방식
- 별도 항목인 **request body**에 넣어 요청
- url에 파라미터가 노출되지 않기 때문에 **보안**에 상대적으로 유리
- **<form>** 태그를 사용해 사용자로부터 정보를 입력받는 경우, 또는 서버로 전송해야 할 데이터의 **용량이 클** 경우, **POST** 방식을 사용
- 아래 코드는 삼성전자 시황을 가져오는 코드이며 실행결과는 **requests.get()**과 동일

```
res = requests.post("https://finance.naver.com/item/sise.nhn?code = 005930", verify = False)
print (res.text)
```

● HTML 소스를 데이터로 변환하기

➤ BeautifulSoup이란?

- 복잡하게 작성되어 있는 HTML 문서를 **구조화된 데이터 형태로 변환**하는 파이썬 패키지
- HTML을 데이터로 변환하는 과정을 파싱(Parsing)이라고 하는데 파이썬에서 코드를 통해 접근할 수 있는 형태로 변환하는 것을 말함
- CSS Selector 를 지원하여 데이터 탐색이 간편하고, 원하는 데이터만 추출해 별도로 정리 가능
- HTML 데이터 수집은 **requests**로, 수집한 데이터의 필터링과 선택은 **BeautifulSoup**으로 수행

➤ BeautifulSoup 설치 : 주피터 노트북에서 아래 pip 명령어 입력으로 설치

```
!pip install bs4           # BeautifulSoup 설치
```

HTML 소스 가져오기

● HTML 소스를 데이터로 변환하기

➤ HTML 태그를 이용해 원하는 속성 찾기

- BeautifulSoup의 **find()**, **find_all()** 함수를 이용해서 html 페이지내 원하는 태그를 찾을 수 있음

➤ html 소스를 분석용 데이터로 파싱(변환)하기

① BeautifulSoup 모듈을 불러오고 html_txt 변수에 html 코드와 텍스트를 입력해서 분석할 자료를 생성

② BeautifulSoup() 함수를 이용해 html 소스를 분석용 데이터로 파싱(변환)

• `soup = BeautifulSoup(html_txt, "html.parser")` 코드로 파싱하면 결과가 soup 변수에 저장

```
from bs4 import BeautifulSoup
```

```
# BeautifulSoup 모듈 불러오기
```

```
# ① html_txt 변수에 html 코드와 텍스트 입력
```

```
html_txt = "<p class = 'weather' id = 'tw'>오늘의 날씨</p> <h1> 한때 소나기가 내리겠습니다. </h1>"
```

```
soup = BeautifulSoup(html_txt, "html.parser") # ② 파싱하기, 분석 가능한 데이터 형태로 변환
soup
```

```
<p class="weather" id="tw">오늘의 날씨</p> <h1> 한때 소나기가 내리겠습니다.</h1>
```

● HTML 소스를 데이터로 변환하기

➤ find() 함수로 원하는 속성 찾기

- soup에 다양한 함수를 붙여 사용할 수 있는데 태그를 찾는 함수는 **find()**
- **print (tag)**는 find() 함수에서 찾은 **p** 태그내 모든 속성값을 보여 줌

```
tag = soup.find("p")                                # p 태그를 찾아 tag에 저장
print (tag)                                         # tag 출력
print(tag.name)                                    # tag 명 출력
print( tag.attrs )                                # tag 속성 출력
print(tag.attrs["class"])                          # tag 속성 중 class만 출력
print(tag.attrs["id"])                            # tag 속성 중 id만 출력
print(tag.text)                                    # tag 내 텍스트 출력
```

출력 결과

<p class="weather" id="tw">오늘의 날씨</p>

p
{'class': ['weather'], 'id': 'tw'}
['weather']
tw
오늘의 날씨

<p class="weather" id="tw">오늘의 날씨</p> <h1> 한때 소나기가 내리겠습니다.</h1>

HTML 소스 가져오기

- HTML 소스를 데이터로 변환하기

- CSS 를 이용해 원하는 속성 찾기

- 웹 크롤링 시 태그를 찾을 때는 `find()` 보다는 `select()`나 `select_one()` 함수를 주로 사용
 - `find()` : html 태그를 통해
 - `select()` : html 태그와 CSS를 이용해 원하는 속성을 정확하게 찾을 수 있음

➤ html 소스를 분석용 데이터로 파싱(변환)하기

```
from bs4 import BeautifulSoup          # BeautifulSoup 모듈 불러오기
html_txt = """
<html>
  <head><title>BS page</title></head>
  <body>
    <h1 class="portal_cls">검색포털</h1>
    <p>
      <a href="http://www.daum.net">다음 바로가기</a><br>
      <ul>
        <li>장미 축제
          <a href="http://www.daum.net/rose">장미 축제 바로가기</a><br>
        <li>불꽃 축제
          <a href="http://www.daum.net/fireworks">불꽃 축제 바로가기</a><br>
        </ul>
      <a href="http://www.naver.com">네이버 바로가기</a>
    </p>
    <a href="http://www.google.com" class="alink_cls">구글</a>
    <p class="footage_cls" id="company">2021, ABC Company </p>
    <p class="footage_cls" id="addr">Korea</p>
  </body>
</html>
"""

soup = BeautifulSoup(html_txt , "html.parser") # 파싱하기, 분석 가능한 데이터 형태로 변환
```


➤ 특정 태그 찾기

- `<h1>` 태그를 `select_one()`으로 찾아 텍스트만 출력하는 코드임
- `find()` 함수와 유사하게 사용 가능
- `select_one()` 은 조건에 맞는 태그를 하나만 가져옴

```
tag = soup.select_one("h1")  
print (tag.text)
```

```
# <h1> </h1> 태그 추출  
# tag 안에 있는 텍스트만 출력
```

출력결과

검색포털

- `<h1>` 태그를 `select()` 함수를 사용
- `select()` 함수로 태그를 추출하면 **list형 자료로 반환**되기 때문에 반복문을 활용해 데이터를 하나씩 가져와야 함

```
tag_list = soup.select("h1")  
for tag in tag_list:  
    print (tag.text)
```

```
# <h1> ~ </h1> 태그 모두 찾기  
# tag_list 내 데이터를 하나씩 읽어오기  
# 텍스트만 출력
```

출력결과

검색포털

➤ 자식/자손 태그 찾기

- 자식 태그를 표현할 때는 **>** 기호로 표현
 - ✓ `soup.select("body p > a")` 코드는 `<body>`의 `p` 태그 하위에 있는 `a` 태그를 모두 추출하라는 의미
- 자손 태그를 표현할 때는 **공백(white space)**으로 표현
 - ✓ 예) `soup.select("body p a")` 코드는 `<body>` `p` 태그 하위의 `a` 태그를 추출하고, 그 하위의 모든 태그도 추출

```
tag_list = soup.select("body p > a")
for tag in tag_list:
    print (tag.text)
```

```
# <body><p> 태그 하위의 <a> 태그 추출
# tag_list 내 데이터를 하나씩 읽어오기
# 텍스트만 출력
```

출력결과

다음 바로가기
네이버 바로가기

```
<body>
<h1 class="portal_cls">검색포털</h1>
<p>
  <a href="http://www.daum.net">다음 바로가기</a><br>
  <ul>
    <li>장미 축제
      <a href="http://www.daum.net/rose">장미 축제 바로가기</a><br>
    <li>불꽃 축제
      <a href="http://www.daum.net/fireworks">불꽃 축제 바로가기</a><br>
  </ul>
  <a href="http://www.naver.com">네이버 바로가기</a>
</p>
<a href="http://www.google.com" class="alink_cls">구글</a>
<p class="footage_cls" id="company">2021, ABC Company </p>
<p class="footage_cls" id="addr">Korea</p>
</body>
```

➤ 자식/자손 태그 찾기

- 자식 태그를 표현할 때는 **>** 기호로 표현
 - ✓ `soup.select("body p > a")` 코드는 `<body>`의 `p` 태그 하위에 있는 `a` 태그를 모두 추출하라는 의미
- 자손 태그를 표현할 때는 **공백(white space)**으로 표현
 - ✓ 예) `soup.select("body p a")` 코드는 `<body>` `p` 태그 하위의 `a` 태그를 추출하고, 그 하위의 모든 태그도 추출

```
tag_list = soup.select("body p a")
for tag in tag_list:
    print (tag.text)
```

```
# <body><p> 태그 하위의 <a> 태그 그 하위 모든 태그 추출
# tag_list 내 데이터를 하나씩 읽어오기
# 텍스트만 출력
```

출력결과

다음 바로가기
장미 축제 바로가기
불꽃 축제 바로가기
네이버 바로가기

```
<body>
<h1 class="portal_cls">검색포털</h1>
<p>
  <a href="http://www.daum.net">다음 바로가기</a><br>
  <ul>
    <li>장미 축제
      <a href="http://www.daum.net/rose">장미 축제 바로가기</a><br>
    <li>불꽃 축제
      <a href="http://www.daum.net/fireworks">불꽃 축제 바로가기</a><br>
  </ul>
  <a href="http://www.naver.com">네이버 바로가기</a>
</p>
<a href="http://www.google.com" class="alink_cls">구글</a>
<p class="footage_cls" id="company">2021, ABC Company </p>
<p class="footage_cls" id="addr">Korea</p>
</body>
```

- HTML 소스를 데이터로 변환하기

- CSS 클래스로 태그 찾기 : select() 함수의 파라미터에 **.클래스명** 으로 입력

```
tag_list = soup.select(".footage_cls")          # footage_cls 클래스 추출
for tag in tag_list:                             # tag_list 내 데이터를 하나씩 읽어오기
    print (tag.text)                             # 텍스트만 출력
```

----- 출력결과 -----

2021, ABC Company
Korea

- CSS ID로 태그 찾기 : select() 함수의 파라미터에 **#아이디명** 으로 입력

```
tag_list = soup.select("#company")              # id가 company인 태그 추출
for tag in tag_list:                             # tag_list 내 데이터를 하나씩 읽어오기
    print (tag.text)                             # 텍스트만 출력
```

----- 출력결과 -----

2021, ABC Company

- HTML 소스를 데이터로 변환하기

- 태그 속성(attribute) 값을 이용해서 태그 찾기

```
tag_list = soup.select("a[href]")
for tag in tag_list:
    print(tag.text, tag.attrs["href"])
```

```
# a 태그 중 속성 값 href 를 갖는 태그 추출
# tag_list 내 데이터를 하나씩 읽어오기
# 텍스트와 url 출력
```

출력결과

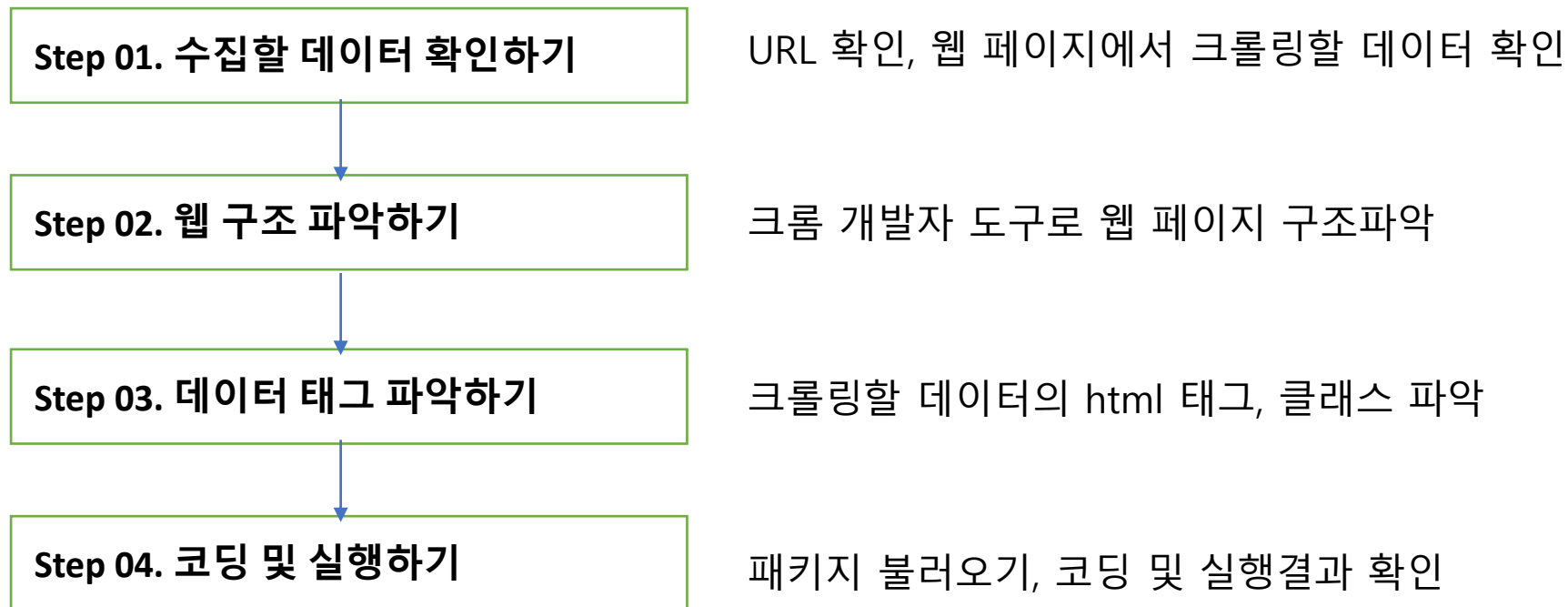
다음 바로가기 <http://www.daum.net>
장미 축제 바로가기 <http://www.daum.net/rose>
불꽃 축제 바로가기 <http://www.daum.net/fireworks>
네이버 바로가기 <http://www.naver.com>
구글 <http://www.google.com>

서점 베스트셀러 정보 가져오기

● 실습 개요

- 국내 최대 서점인 교보문고 웹 페이지에 접속해서 베스트셀러 도서, 저자, 가격 정보를 가져오기

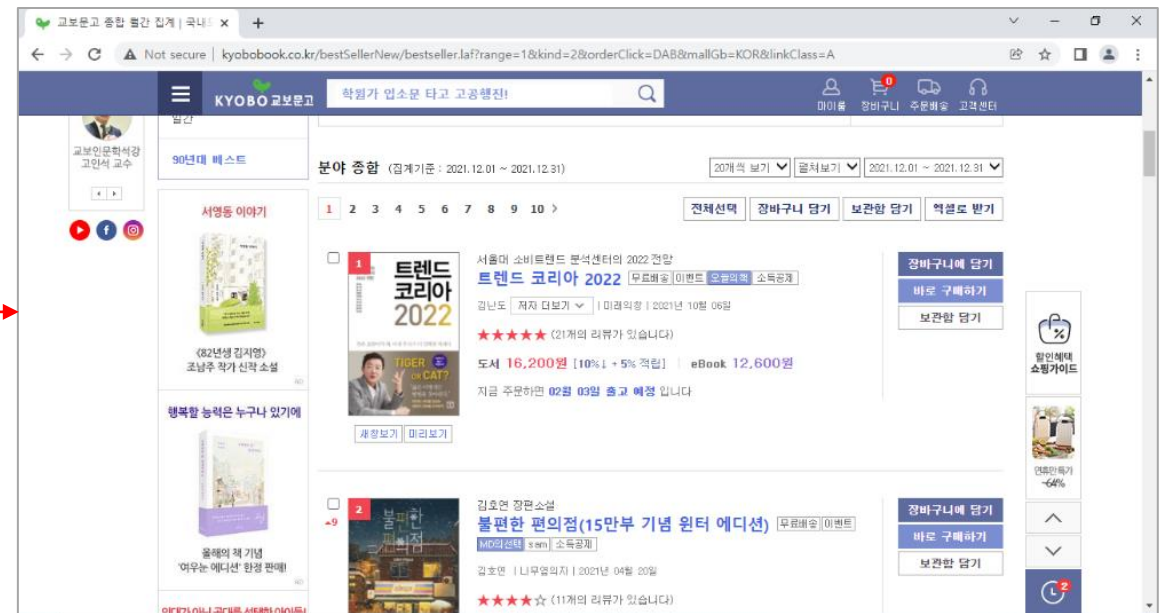
【실습절차】



서점 베스트셀러 정보 가져오기

● Step 01. 수집할 데이터 확인하기

- 교보문고 베스트셀러 목록이 있는 웹 페이지 주소를 확인하고, 도서, 저자, 출판사, 가격 데이터가 어느 위치에 있는지 대략적으로 확인
- 크롬으로 웹 페이지 접속
 - 크롬으로 교보문고 베스트셀러 웹 페이지인 "<http://www.kyobobook.co.kr/bestSellerNew/bestseller.laf>"에 접속한 후, 왼쪽 메뉴바에서 [월간]을 선택
 - 한 페이지에 종합 월간 베스트셀러 목록이 20개씩 보여지는 것을 확인



서점 베스트셀러 정보 가져오기

● Step 01. 수집할 데이터 확인하기


➤ 데이터 위치 확인

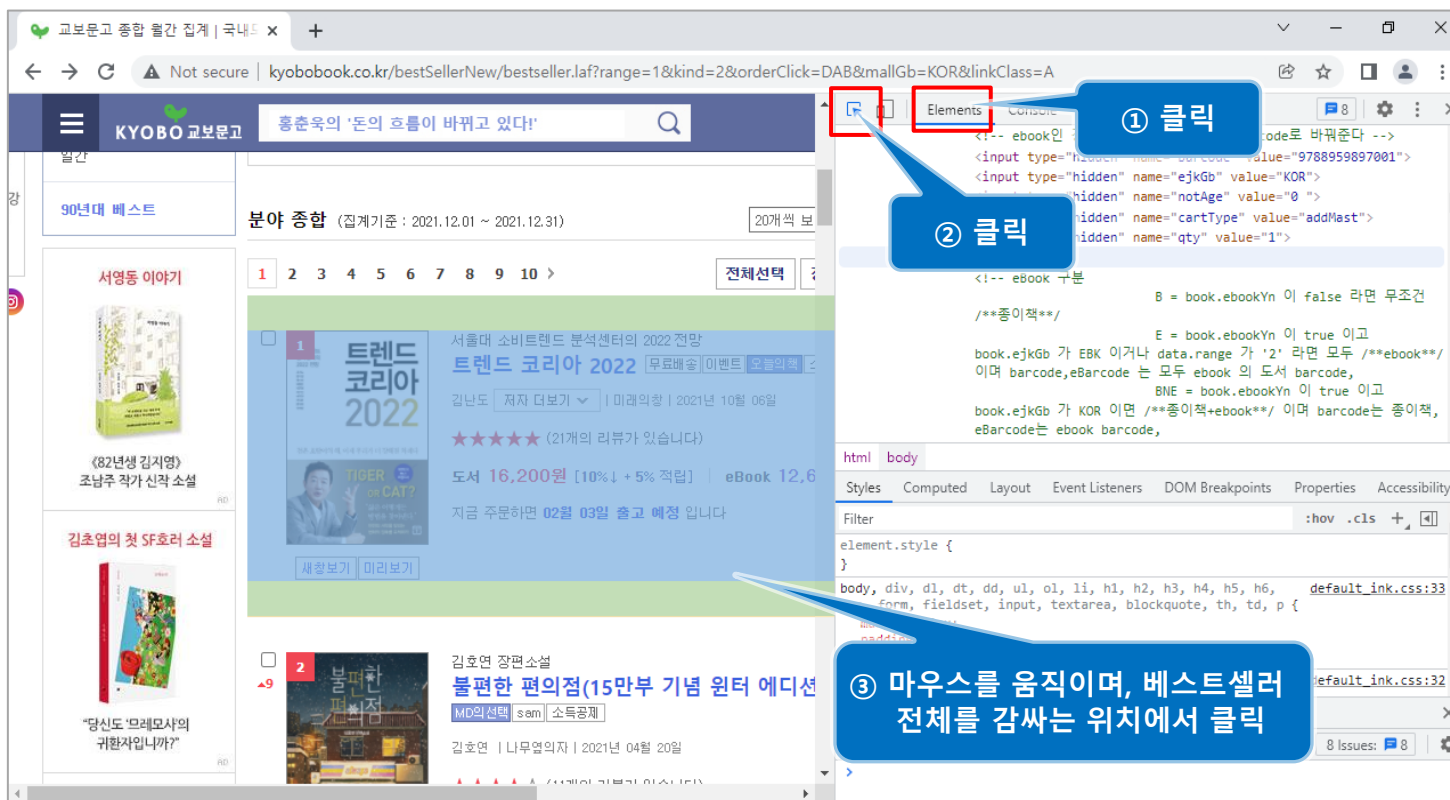
- 정확한 데이터를 가져오기 위해서는 어떤 웹 페이지에 어떤 데이터가 있는지 확인
- 도서제목을 누르면 도서별 상세 페이지가 보이는데 도서명, 저자, 가격 등이 일관성 있게 배치되어 있음



서점 베스트셀러 정보 가져오기

● Step 02. 웹 구조 파악하기

- 크롬 개발자 도구를 활용해 베스트셀러 목록 페이지와 도서별 상세 페이지 URL 을 확인 후 웹 페이지 접속
 - 베스트셀러 목록 페이지로 돌아와 [F12] 키를 누르면 화면의 우측에 크롬 개발자 도구가 나타남
 - [Elements] 탭에서 Inspector  아이콘을 클릭한 후, 도서 목록에서 마우스를 움직이면서 원하는 도서 전체가 선택될 때 클릭하면 해당 HTML 코드를 확인

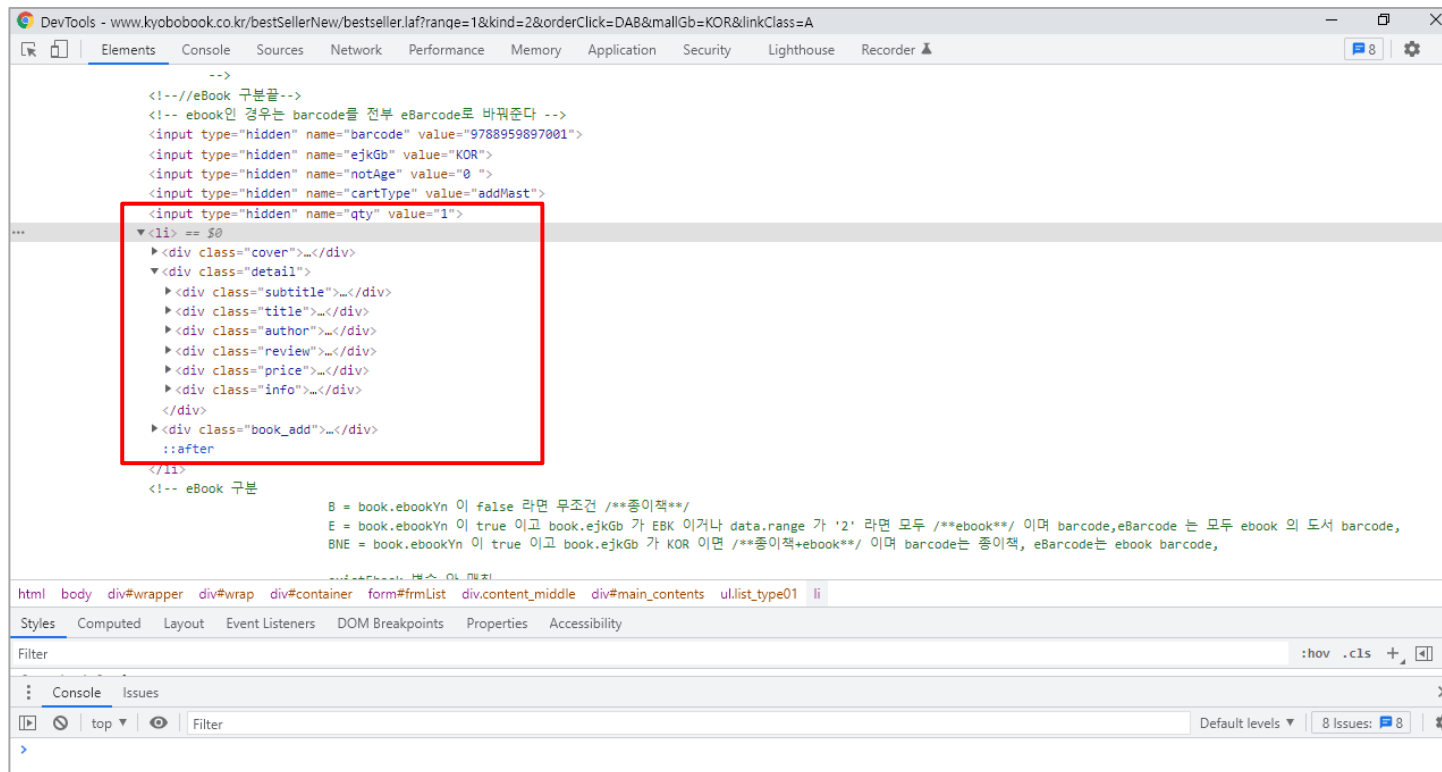


서점 베스트셀러 정보 가져오기

● Step 02. 웹 구조 파악하기

➤ 웹 구조 파악


- html 소스창에서 마우스 커서를 움직여 태그를 분석해보면 도서는 **** 태그로 구성되어 있음
- 하위 구성요소로 "cover", "detail", "book_add"라는 클래스를 가지고 있음
- <div class="detail"> 태그를 클릭하면 하위 요소인 title, author, review, price 클래스도 확인할 수 있음

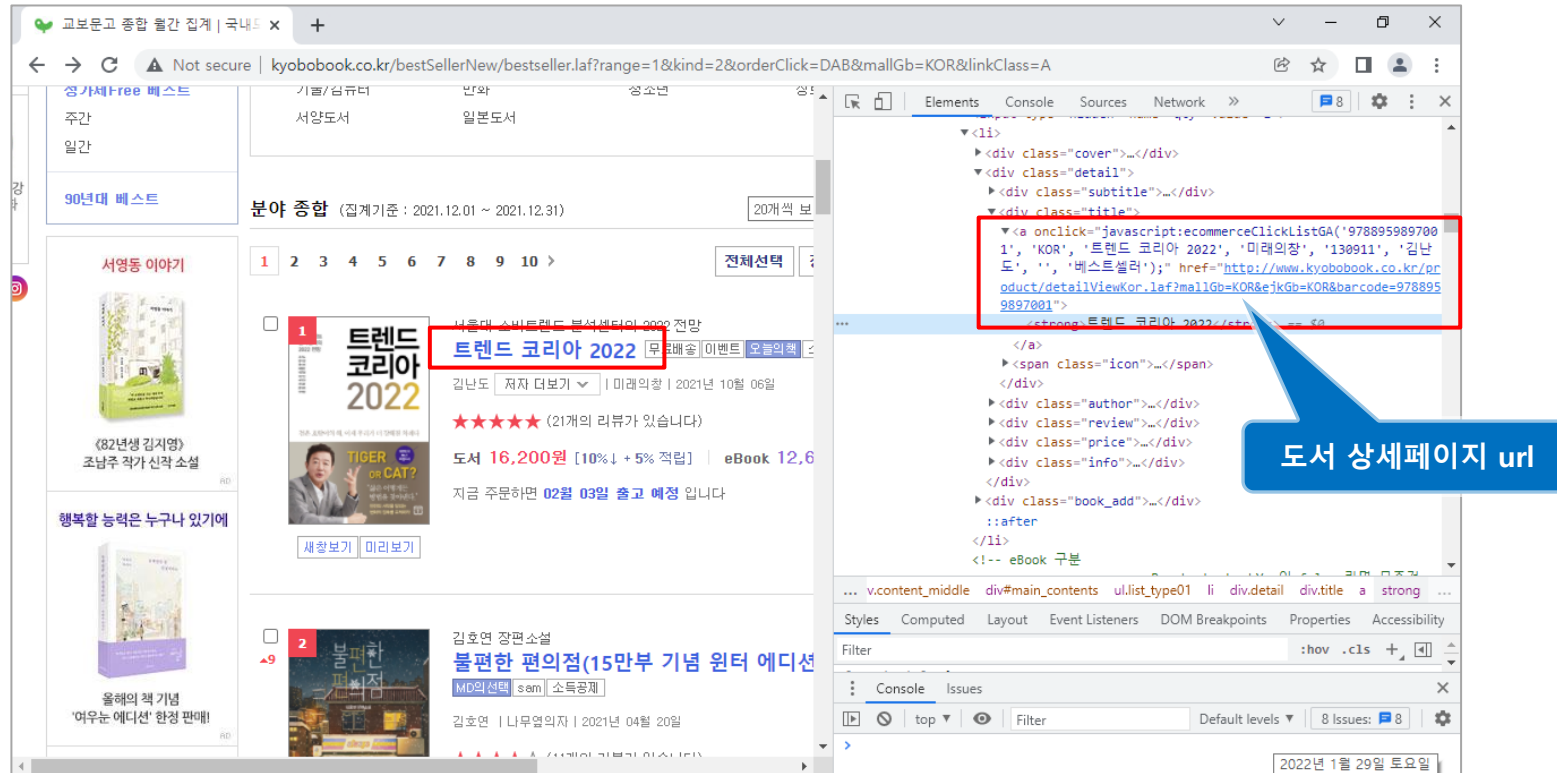


서점 베스트셀러 정보 가져오기


● Step 02. 웹 구조 파악하기

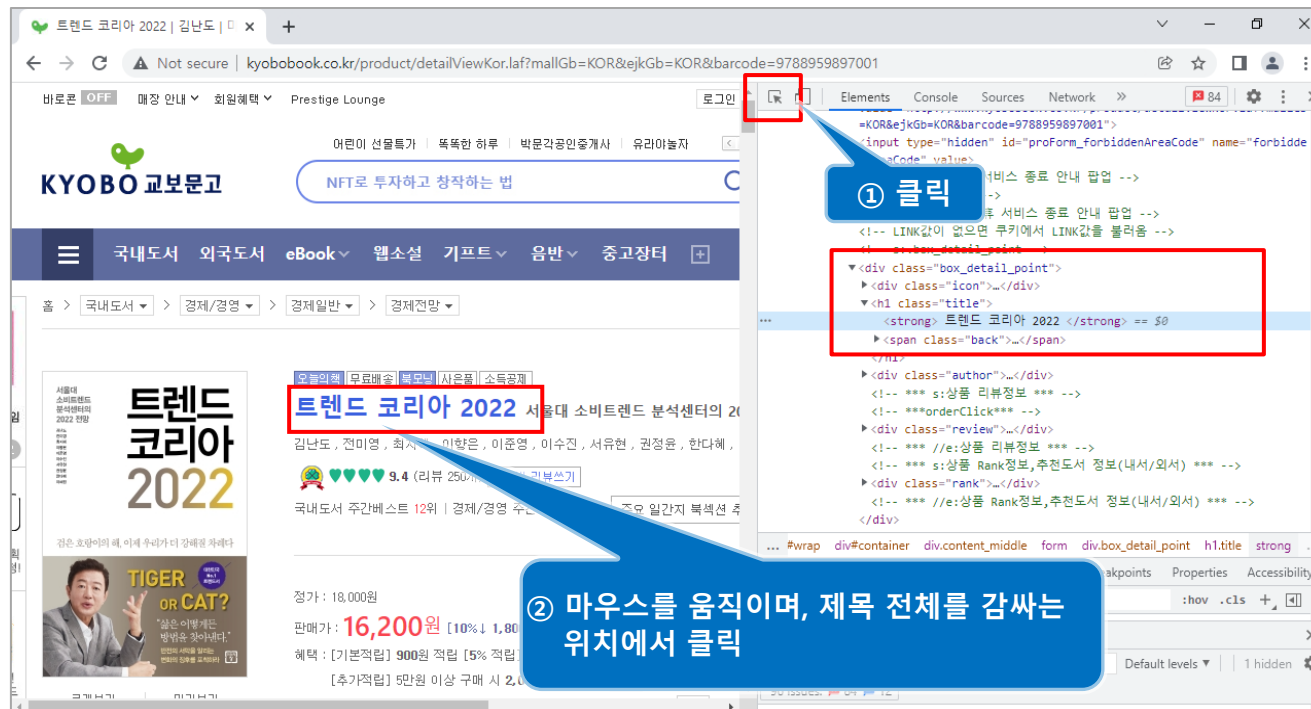
➢ 연결된 웹 페이지 확인

- Inspector  아이콘이 활성화된 상태에서 베스트셀러 1위 도서의 제목을 클릭해보면 개발자 도구 창에 ` ... ` 태그가 하이라이트되는 것을 확인
- 도서별 상세 페이지 주소가 `<a href>` 태그 안에 있는 것도 확인



● Step 03. 데이터 태그 파악

- 도서별 상세 페이지에서 도서, 저자, 도서 가격을 가져온다고 할 때 데이터가 어떤 클래스 안에 있는지, 어떤 태그로 구성되어 있는지 확인해야 함
- 데이터별 태그 확인
 - 도서 제목을 클릭해 도서별 상세 페이지로 들어감
 - 도서별 상세 페이지에서 Inspector 아이콘  을 클릭한 후, 도서의 제목을 클릭하면 해당 위치의 HTML을 확인
 - `<div class="box_detail_point"><h1 class="title"> 도서 ` 태그로 구성되어 있음



● Step 03. 데이터 태그 파악

➤ 태그 정리

- **Inspector** 아이콘을 활용하여 도서, 저자, 가격 태그를 모두 확인한 후 정리해보면 다음 표와 같음
- 크롤링 코드 작성 시 데이터별 태그를 검토한 후, 적합한 코드를 작성해야 함

데이터명	태그
도서	<div class="box_detail_point"><h1 class="title"> 도서
저자	<div class="author"> 저자
도서 가격	 가격

- HTML 파싱 후 for 문으로 동일한 태그를 불러와 데이터를 추출하는 과정이기 때문에, 추출할 데이터 구조나 태그가 원본과 다르면 추출할 수 없다.

● Step 04. 코딩 및 실행하기

- 교보문고의 베스트셀러 웹 페이지에서 도서 20개의 상세 웹 페이지 주소를 추출한 후, 도서별 상세정보인 도서, 저자, 가격 데이터를 크롤링
- 패키지 불러오기

```
import requests  
from bs4 import BeautifulSoup as bs  
import pandas as pd
```

```
# 웹 리소스를 가져오는 모듈  
#데이터 변환(파싱) 및 원하는 태그 추출 시 사용  
# 데이터프레임
```

● Step 04. 코딩 및 실행하기

kyobobook.co.kr/bestSellerNew/bestseller.laf?range=1&kind=2&orderClick=DAB&mallGb=KOR&linkClass=A

➤ 교보문고의 월간 베스트셀러 웹 페이지 가져오기

- **requests.get()** 함수로 교보문고 월간 베스트셀러 웹 페이지 리소스를 가져와 html 변수에 저장
- **BeautifulSoup**을 이용해 html 변수에 들어있는 데이터를 파싱
- 파싱한 정보를 담고 있는 book을 출력하면 교보문고 베스트셀러 웹 페이지의 HTML 소스 코드들을 볼 수 있음

```
res = requests.get("http://www.kyobobook.co.kr/bestSellerNew/bestseller.laf?range = 1  
1&kind=2&orderClick = DAB&mallGb=KOR&linkClass=A") # 웹 페이지 주소를 확인하여 붙여넣기  
html = res.content # 웹 리소스 가져오기  
book = bs(html, "html.parser") # html 파싱  
print(book)
```

출력결과

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
  
<html lang="ko" xml:lang="ko" xmlns="http://www.w3.org/1999/xhtml">  
<!-- s:html:head -->  
<head>  
<title>교보문고 종합 주간 집계 | 국내도서 | 베스트셀러 - 교보문고</title>  
<!--MS의 최신 웹브라우저인 edge 브라우저 호환을 위해 넣어줌-->  
<meta content="IE=edge" http-equiv="X-UA-Compatible"/>  
<meta content="text/html; charset=utf-8" http-equiv="Content-Type">  
<meta content="nocache" http-equiv="Pragma"/>  
<meta content="0" http-equiv="Expires"/>  
<meta content="no-cache" http-equiv="Cache-Control"/>  
<link href="http://image.kyobobook.co.kr/newimages/apps/b2c/kyobo.ico" rel="shortcut icon"/>  
<link href="http://image.kyobobook.co.kr/ink/css/default_ink.css" rel="stylesheet" type="text/css"/>
```

● Step 04. 코딩 및 실행하기

➤ 도서 상세 웹 페이지 주소 추출하기

- 도서별 상세 주소는 `<div class="detail">` 하위에 있는 `<a>` 태그에 `href` 속성으로 입력되어 있다.
- `select_one("div.title > a")`로만 지정하면 원하는 URL 이 아니라 `<a href...>텍스트...` 처럼 태그 정보도 포함되므로 `href` 의 값을 가져오기 위해 `.attrs["href"]`를 추가해야 함

```
book_list = []
for book_detail in book.select("div.detail"):
    book_urls = book_detail.select_one("div.title > a").attrs["href"]
    book_list.append(book_urls)
print(book_list)
```

#book_list 리스트 변수 만들기
#book에서 class=detail 태그 가져오기
#상세주소를 book_urls에 저장
#book_list 변수에 주소를 하나씩 추가

출력결과

```
['http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791196661984', 'http://www.kyobobook.co.kr/product/de
tailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791161571188', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&ba
rcode=9788991622869', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9788959897001', 'http://www.kyobo
book.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791165344252', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?ma
llGb=KOR&ejkGb=KOR&barcode=9791165216603', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=979119105655
6', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791190030922', 'http://www.kyobobook.co.kr/produc
t/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9791130677774', 'http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=K0
```


● Step 04. 코딩 및 실행하기

➤ 도서별 상세 데이터 추출하여 저장하기

- best 라는 데이터 프레임을 만들고 열 이름을 '도서', '저자', '가격', 'url' 로 지정
- 도서 상세 페이지 정보를 가지고 있는 book_list 를 하나씩 불러와 book_url 에 저장한 후 사용하기 위해 for 문 사용

도서별 상세 데이터 추출하기

```
best = pd.DataFrame(columns=["도서","저자","가격","url"])
for index, book_url in enumerate(book_list):
    res = requests.get(book_url)
    html = res.content
    best_book = bs(html, "html.parser", from_encoding = "cp949")
    title = best_book.select_one("h1.title strong").text.strip()
    author = best_book.select_one("span.name a").text.strip()
    price = best_book.select_one("span.sell_price strong").text.strip()
    best.loc[index+1] = (title, author, price, book_url)
best.head()
```

엑셀 파일로 저장하기

```
best.to_excel(" chapter08/bestseller.xlsx")
```

저장할 데이터프레임 만들기

도서 상세 페이지를 하나씩 반환

도서 상세 페이지 리소스 가져오기

HTML 파싱. 한글 깨짐 방지

도서명 추출

저자명 추출

가격 추출

데이터 프레임에 저장

	도서	저자	가격	url
1	작별인사	김영하	12,600	http://www.kyobobook.co.kr/product/detailViewK...
2	흔한남매 10	흔한남매 (원작)	12,150	http://www.kyobobook.co.kr/product/detailViewK...
3	불편한 편의점(40만부 기념 벚꽃 에디션)	김호연	12,600	http://www.kyobobook.co.kr/product/detailViewK...
4	마음의 법칙	폴커 키즈	14,400	http://www.kyobobook.co.kr/product/detailViewK...
5	문재인의 위로	더휴먼 편집부	14,220	http://www.kyobobook.co.kr/product/detailViewK...

● selenium 특징

- selenium은 웹 브라우저 작업을 능동적으로 자동화하기 위한 패키지임
- 키보드를 통한 입력 기능과 마우스 클릭 기능 등을 제공
- 브라우저 인스턴스를 원격으로 제어하고 사용자와 브라우저와의 상호작용 지원
- 웹 페이지에 텍스트 입력, 체크 박스 선택, 링크 클릭, 마우스 이동 및 임의의 JavaScript 실행
- 웹 페이지에서 자료를 수집하기 위한 RPA(Robotic Process Automation) 구축 시 가장 많이 사용하는 패키지임
- BeautifulSoup에 비하면 속도가 느리고 메모리도 상대적으로 많이 사용함

● 소프트웨어 및 패키지 설치하기

- selenium을 사용하여 웹 브라우저를 제어하기 위해 아래의 소프트웨어 및 파이썬 패키지를 설치

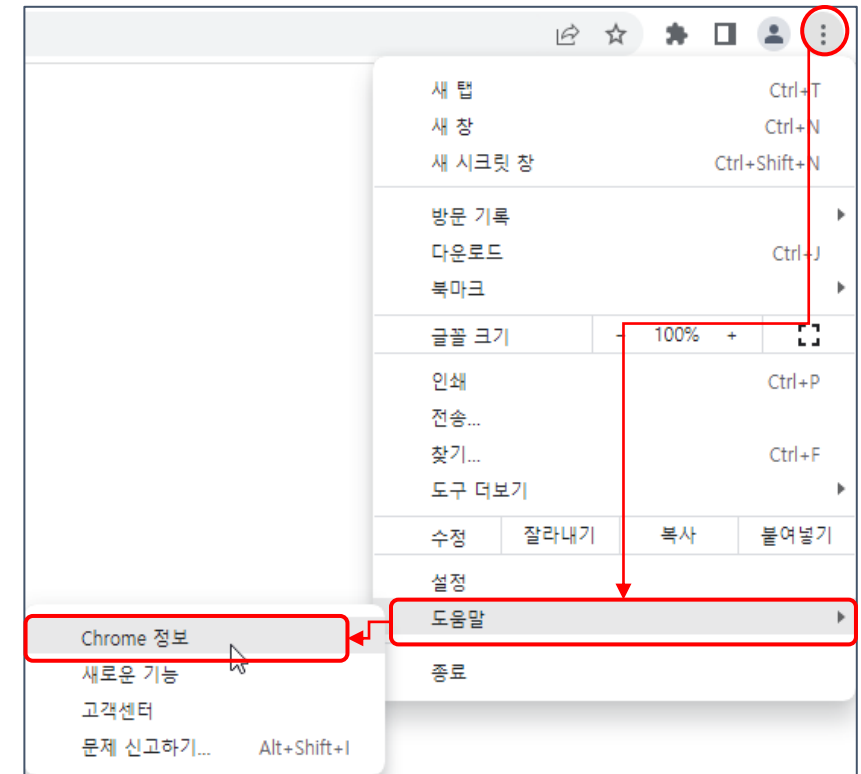
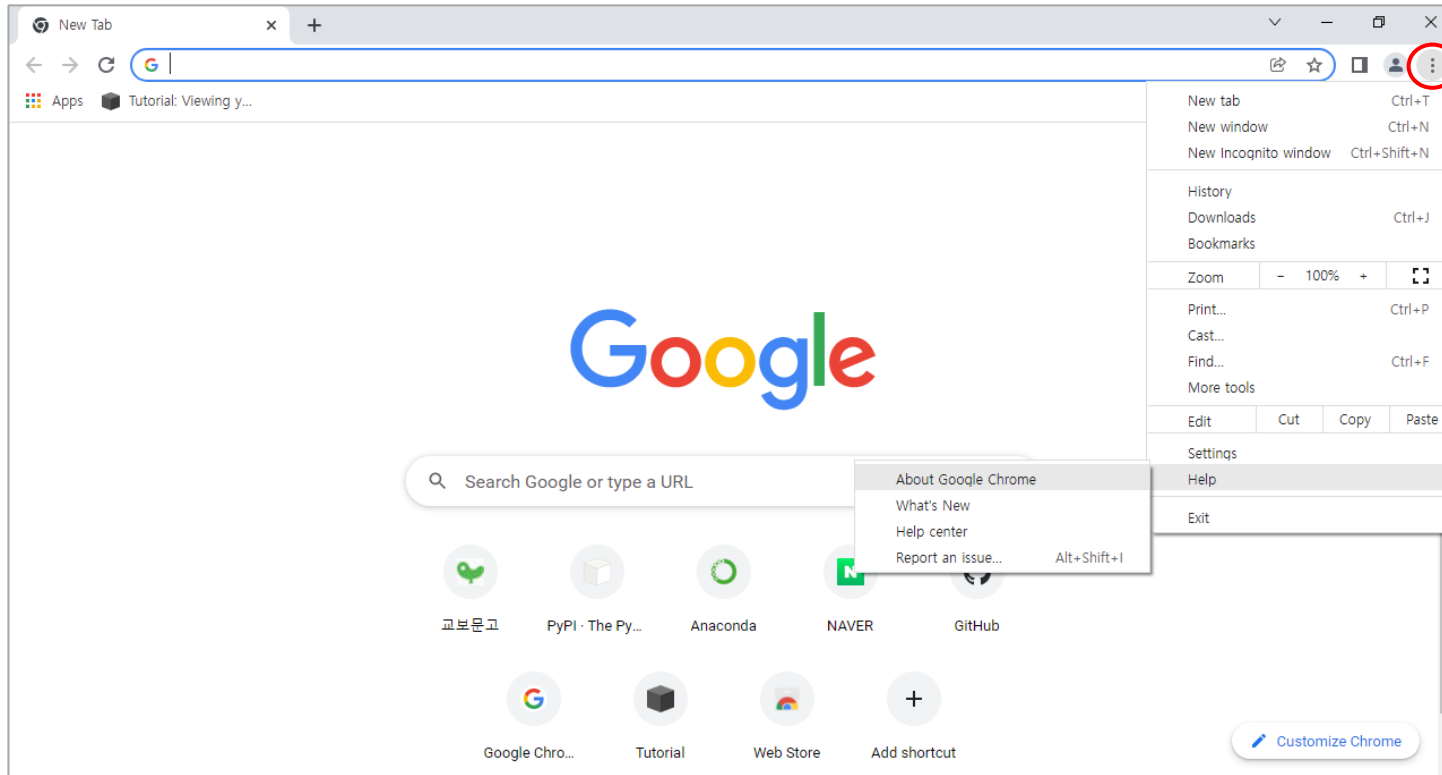
항목	설 명
웹 브라우저	크롬(chrome)
웹 드라이버	브라우저를 제어하기 위한 API를 제공하는 프로그램
selenium	웹 드라이버를 활용하여 웹 브라우저를 제어하는 함수 제공

selenium 개요

● 소프트웨어 및 패키지 설치하기

➤ 웹 드라이버 설치

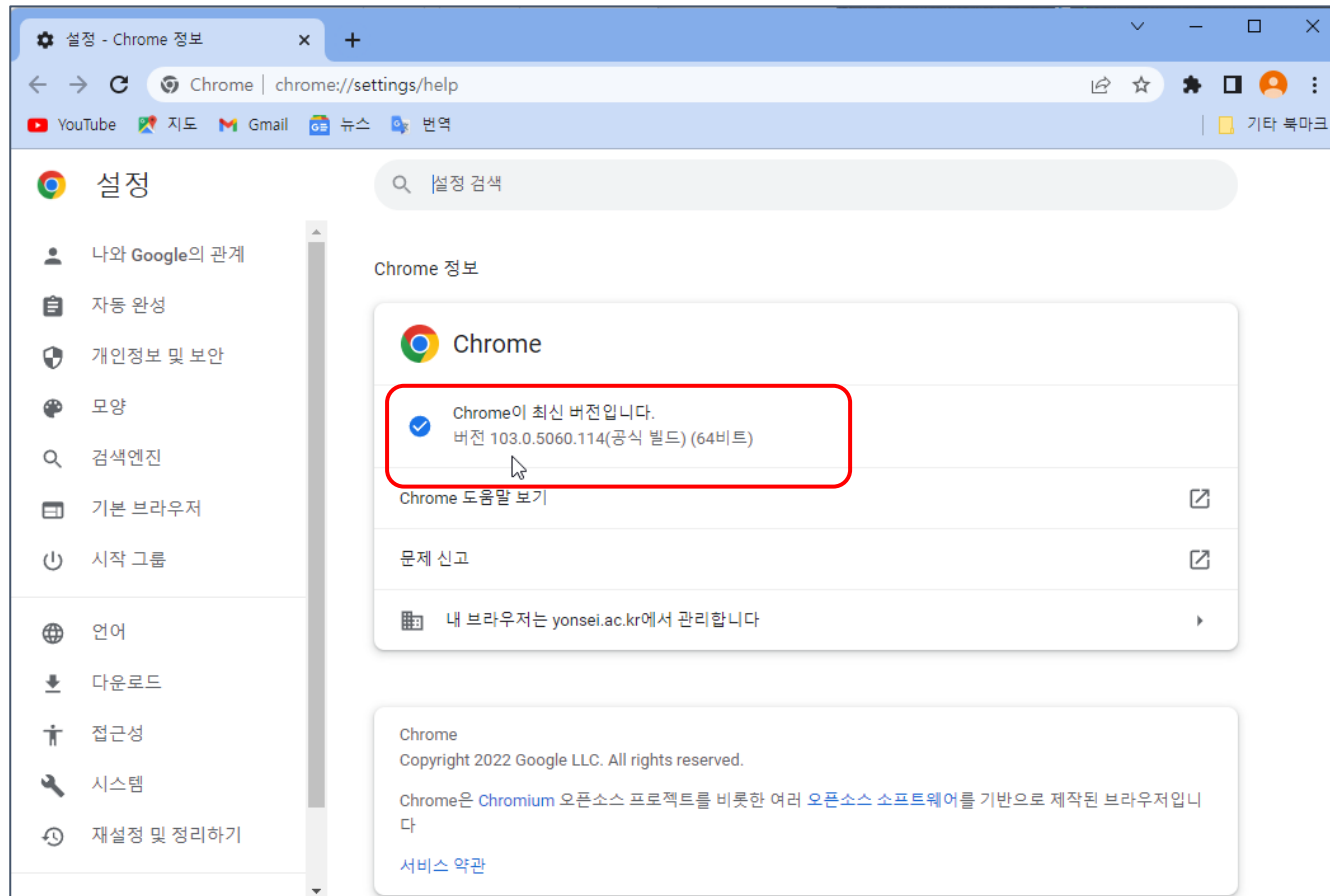
- 웹 드라이버를 설치하기 위해서는 먼저 크롬 버전을 확인해야 함
- 크롬을 실행하고 [Chrome 맞춤설정 및 제어] **⋮** 아이콘을 클릭한 후, [도움말] - [Chrome 정보]를 선택



● 소프트웨어 및 패키지 설치하기

➤ 웹 드라이버 설치

- Chrome 정보 창이 나타나면 현재 버전을 확인



● 소프트웨어 및 패키지 설치하기

➤ 웹 드라이버 설치

- 크롬 드라이버 페이지(<https://chromedriver.chromium.org/downloads>)에 접속해서 크롬 브라우저의 버전과 동일한 웹 드라이버를 다운로드
- windows 사용자는 “chromedriver_win32.zip” 파일을 다운로드
- 압축을 풀면 “**chromedriver.exe**” 파일이 생성되는데 **주피터 노트북을 실행한 폴더(ipynb 파일이 저장되는 폴더 ./Programs)에 복사**

The screenshot shows the ChromeDriver download page. The 'Downloads' section lists several files. The file 'chromedriver_win32.zip' is highlighted with a red box. A red arrow points from this box to the text 'chromedriver.exe'.

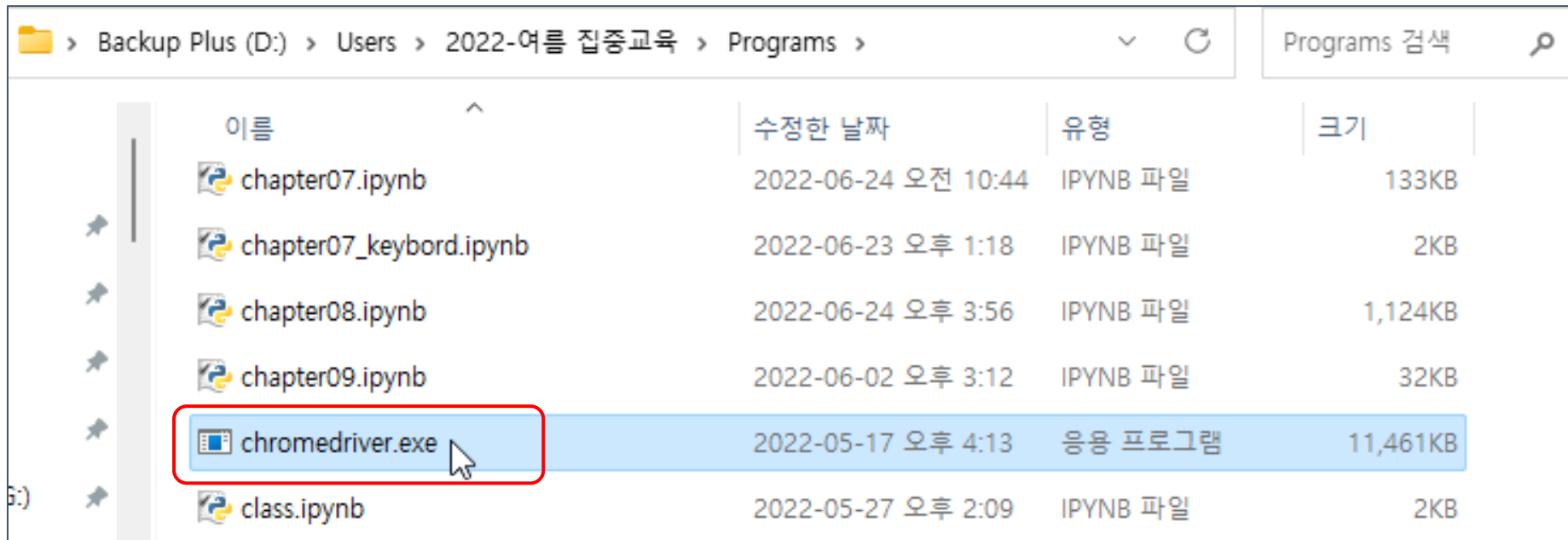
Name	Last modified	Size	ETag
Parent Directory	-	-	-
chromedriver linux64.zip	2022-05-25 09:48:06	5.93MB	29452b3ec1afadc764820f8894fd81ea
chromedriver mac64.zip	2022-05-25 09:48:09	7.89MB	17c3e75d98e7787e5715e10f845c9d09
chromedriver mac64 m1.zip	2022-05-25 09:48:12	7.20MB	398574c4953e9bbc78fb730a28d585d1
chromedriver_win32.zip	2022-05-25 09:48:14	6.07MB	21c5d8c3dd0a59d9c71148dfbdeea380
notes.txt	2022-05-25 09:48:20	0.00MB	9c365c210138ce8af5f878d1b6e6a586

chromedriver.exe

- 소프트웨어 및 패키지 설치하기

- 웹 드라이버 설치

- “chromedriver.exe” 파일을 주피터 노트북을 실행한 폴더(ipynb 파일이 저장되는 폴더 ./Programs)에 복사



- 소프트웨어 및 패키지 설치하기

- selenium : 주피터 노트북에서 아래 pip 명령어 입력으로 설치

```
!pip install selenium
```

웹 로드 및 HTML 소스 가져오기



● 웹 브라우저 열고 닫기

➤ 모듈 불러오기

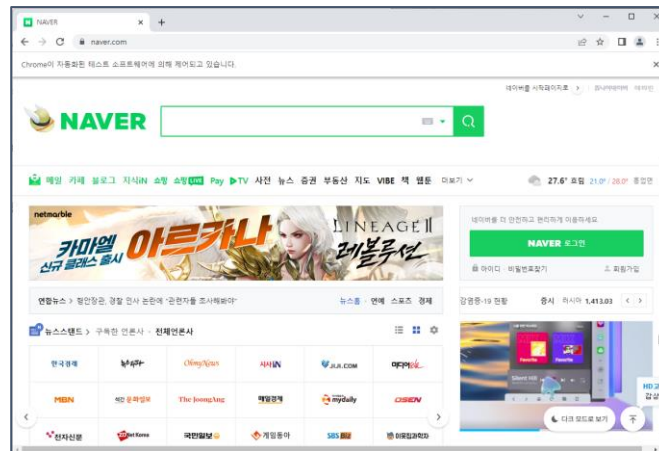
```
import selenium
from selenium import webdriver
```

```
# selenium 모듈 불러오기
# webdriver 모듈 불러오기
```

➤ 크롬 실행 및 네이버 자동 접속

```
URL = "https://www.naver.com"
driver = webdriver.Chrome("chromedriver")
driver.get(URL)
```

```
# 네이버 url을 URL 변수에 저장
# driver 객체 생성
# URL에 저장된 주소를 크롬에서 열기
```



웹 로드 및 HTML 소스 가져오기

- 웹 브라우저 열고 닫기

- 현재 url 확인

```
print(driver.current_url)          # 현재 url 출력
```

```
----- 출력결과 -----  
https://www.naver.com/
```

- 크롬 종료하기

```
driver.close()
```

● HTML 소스 가져오기

➤ 크롬으로 네이버 자동 접속하기

```
import selenium
from bs4 import BeautifulSoup
from selenium import webdriver
URL = "https://www.naver.com"
driver = webdriver.Chrome("chromedriver")
driver.get(URL)
```

selenium 모듈 불러오기
BeautifulSoup 모듈 불러오기
webdriver 모듈 불러오기
네이버 url을 URL 변수에 저장
driver 객체 생성
URL에 저장된 주소를 크롬에서 열기

➤ HTML 소스 파싱하기

```
html = driver.page_source
soup = BeautifulSoup(html, "html.parser")
```

HTML 소스 가져오기
데이터 변환(파싱)

● 텍스트 출력하기

➤ select() 함수로 <body> 내 <p> 태그를 추출하고 반복문을 활용해 텍스트를 출력

```
tag_list = soup.select("body p")          # <body> 내 <p> 태그 추출
for tag in tag_list:                      # tag_list 내 데이터를 하나씩 읽어오기
    print (tag.text)                      # 텍스트 만 출력
driver.close()
```

출력결과


```
ON/OFF 설정은해당기기(브라우저)에 저장됩니다.
동일한 시간대/연령/남녀별 사용자 그룹의관심사에 맞춰 자동완성을 제공합니다.
해당 언론사 사정으로 접근이 제한됩니다.
을(를)구독해지 하시겠습니까?
구독한 언론사에 추가되었습니다.
해당 언론사 사정으로 접근이 제한됩니다.
을(를)구독해지 하시겠습니까?
구독한 언론사에 추가되었습니다.
해당 언론사 사정으로 접근이 일시 제한됩니다.
을(를)구독해지 하시겠습니까?
해당 언론사 사정으로 접근이 제한됩니다.
을(를)구독해지 하시겠습니까?
언론사 구독 설정에서 관심있는 언론사를 구독하시면언론사가 직접 편집한 뉴스들을 네이버 홈에서 바로 보실 수 있습니다.
힘하게 돌아온 두 남녀의 쿨하지 못한 과거사는?
네이버를 더 안전하고 편리하게 이용하세요
```

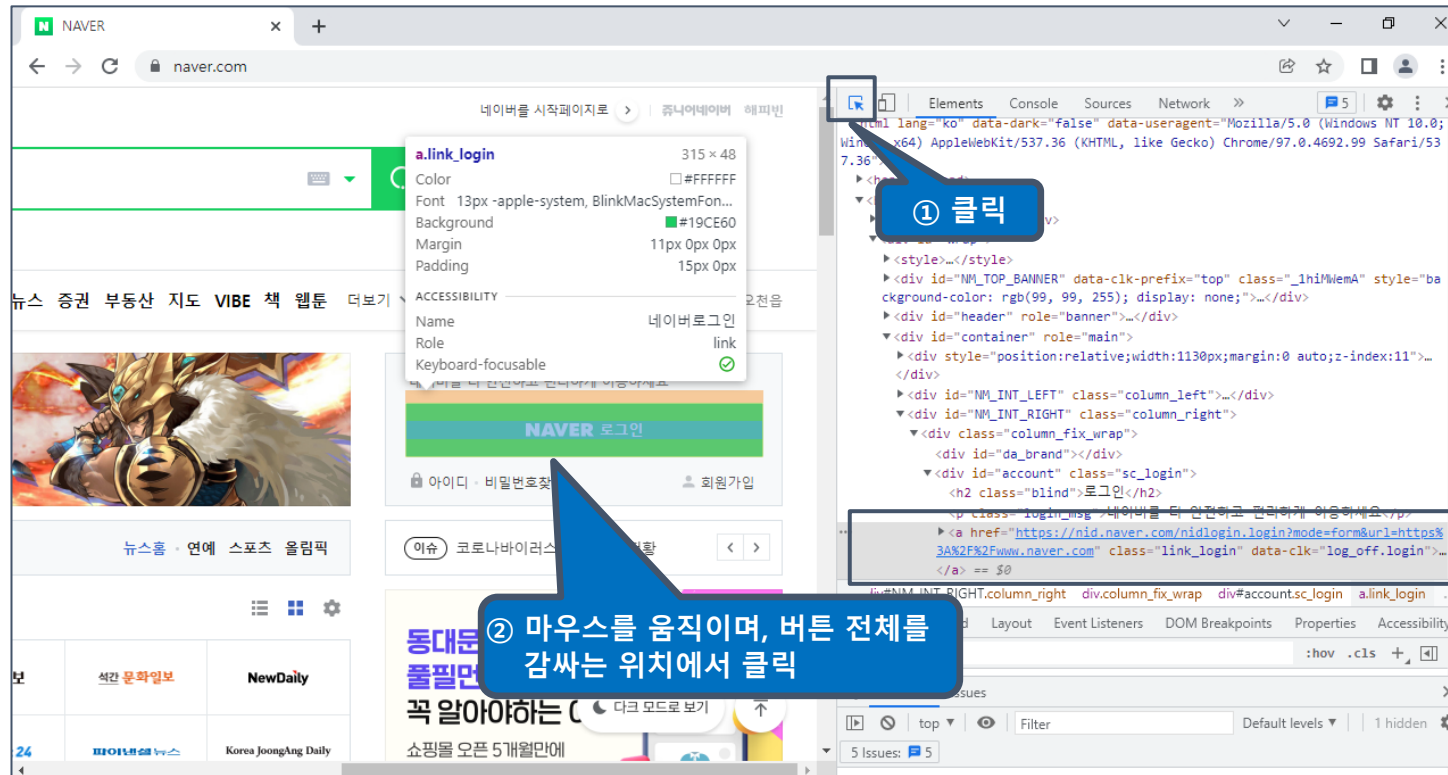
웹 브라우저 제어하기

- ◆ 웹 브라우저를 제어하려면 필요한 웹 페이지 구성요소(버튼, 이미지, 입력창)의 HTML 태그를 먼저 찾아야 한다.
- ◆ 이를 Elements라고 하며, 크롬 개발자 도구를 활용해 찾는다.
- ◆ 이 Elements에 문자열을 입력하거나 마우스를 클릭하면 웹 브라우저 제어가 가능하다.
- ◆ selenium에는 Elements를 찾는 다양한 함수들이 있다.

● 네이버 로그인 버튼 클릭하기

➤ 크롬 개발자 도구 실행

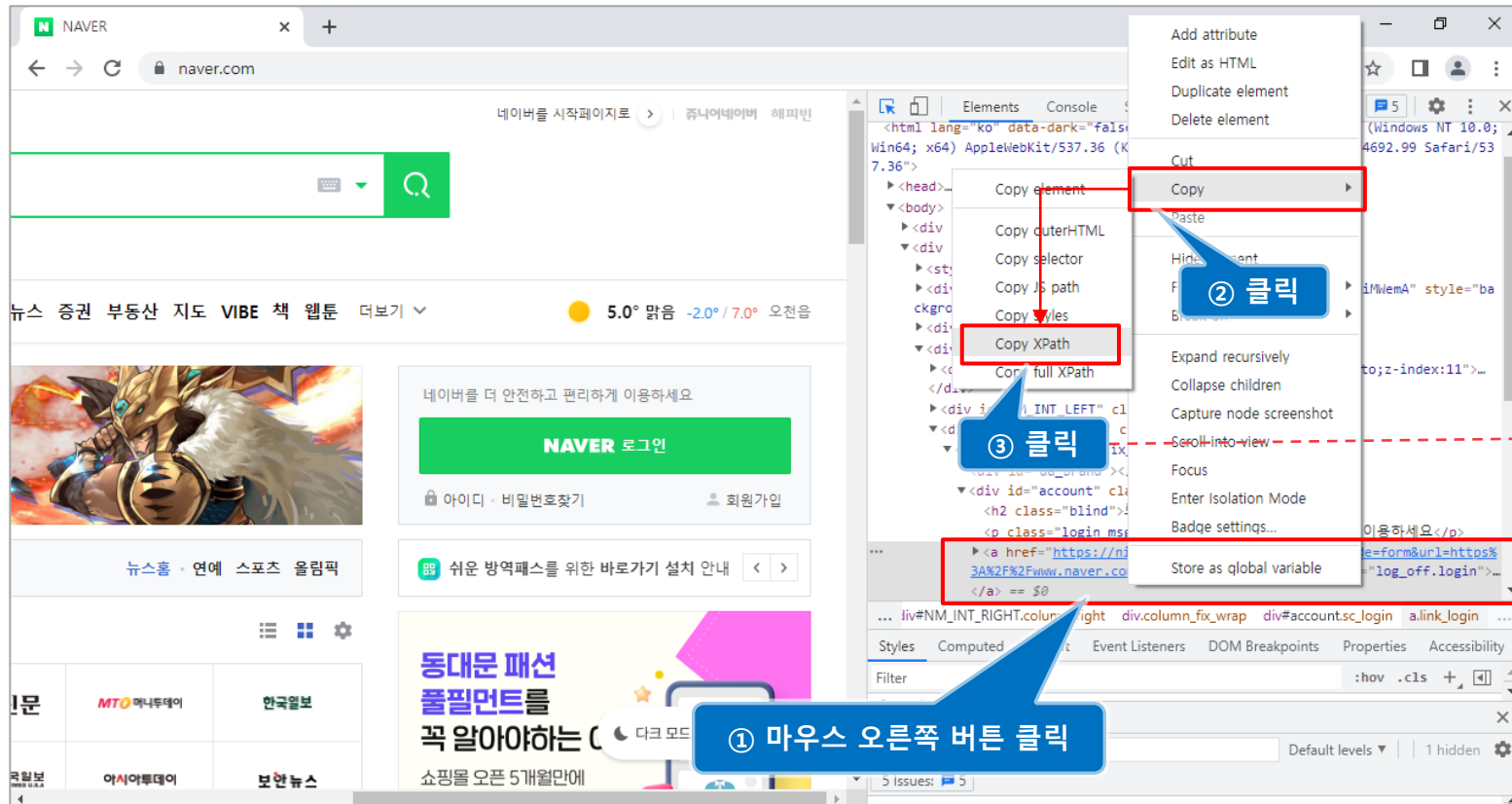
- 네이버의 로그인 버튼의 xpath(HTML 페이지가 조금만 변경되어도 함께 변경)를 찾기 위해,
- 크롬으로 네이버에 접속한 후 [F12] 키를 눌러 개발자 도구를 실행
- Inspector  아이콘을 클릭하고 마우스로 네이버 웹 페이지에 있는 로그인 버튼(**NAVER 로그인**) 을 클릭



● 네이버 로그인 버튼 클릭하기

➤ xpath 복사

- 로그인 버튼의 xpath를 복사하기 위해 개발자 도구에 선택된 html 소스를 마우스 오른쪽 버튼으로 클릭하고 [Copy] - [Copy XPath]를 선택



● 네이버 로그인 버튼 클릭하기

➤ 로그인 버튼 클릭하기

- [Ctrl + v]를 눌러 앞에서 복사한 **xpath**를 **driver.find_element(By.XPATH, ' ')** 함수의 ' ' 안에 붙여 넣으면 네이버 로그인 버튼을 제어 가능
- **btn**에는 xpath로 찾은 네이버 로그인 버튼이 저장되며 **btn.click()** 으로 버튼을 클릭할 수 있음
- 아래 코드를 실행하여 크롬 브라우저에 네이버 로그인 페이지가 실행 된 것을 확인

```
import selenium
from selenium import webdriver
from selenium.webdriver.common.by import By
URL = "https://www.naver.com"
driver = webdriver.Chrome("chromedriver")
driver.get(URL)
btn = driver.find_element(By.XPATH, '//*[@id="account"]/a')
btn.click()
```

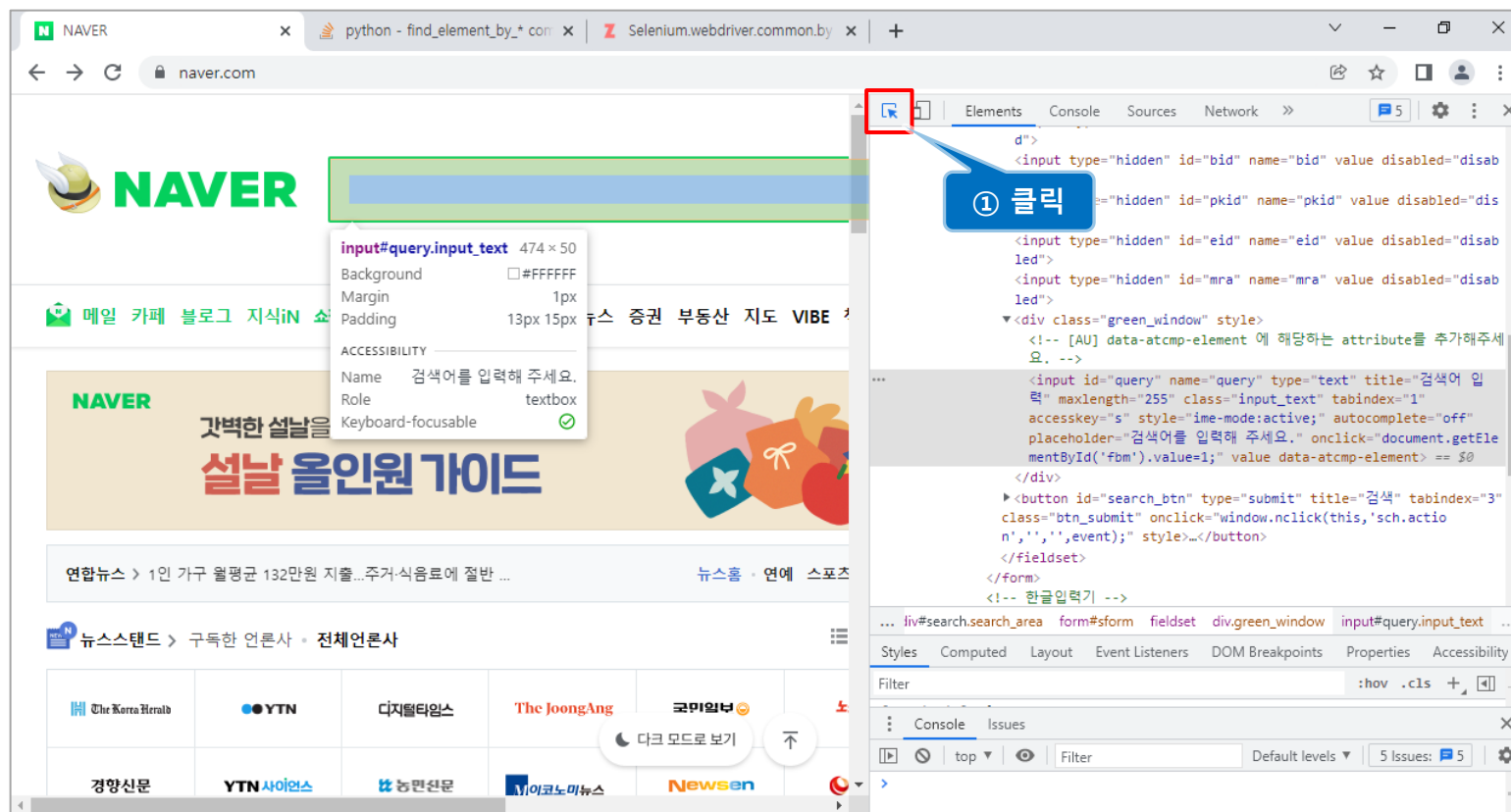
```
# selenium 모듈 불러오기
# webdriver 모듈 불러오기
# selenium의 by 클래스 불러오기
# 네이버 url을 URL 변수에 저장
# driver 객체 생성
# URL에 저장된 주소를 크롬에서 열기
# 네이버 로그인 버튼 선택하기
# 로그인 버튼 클릭하기
```



● 네이버에서 방탄소년단 검색하기

➤ 크롬 개발자 도구 실행

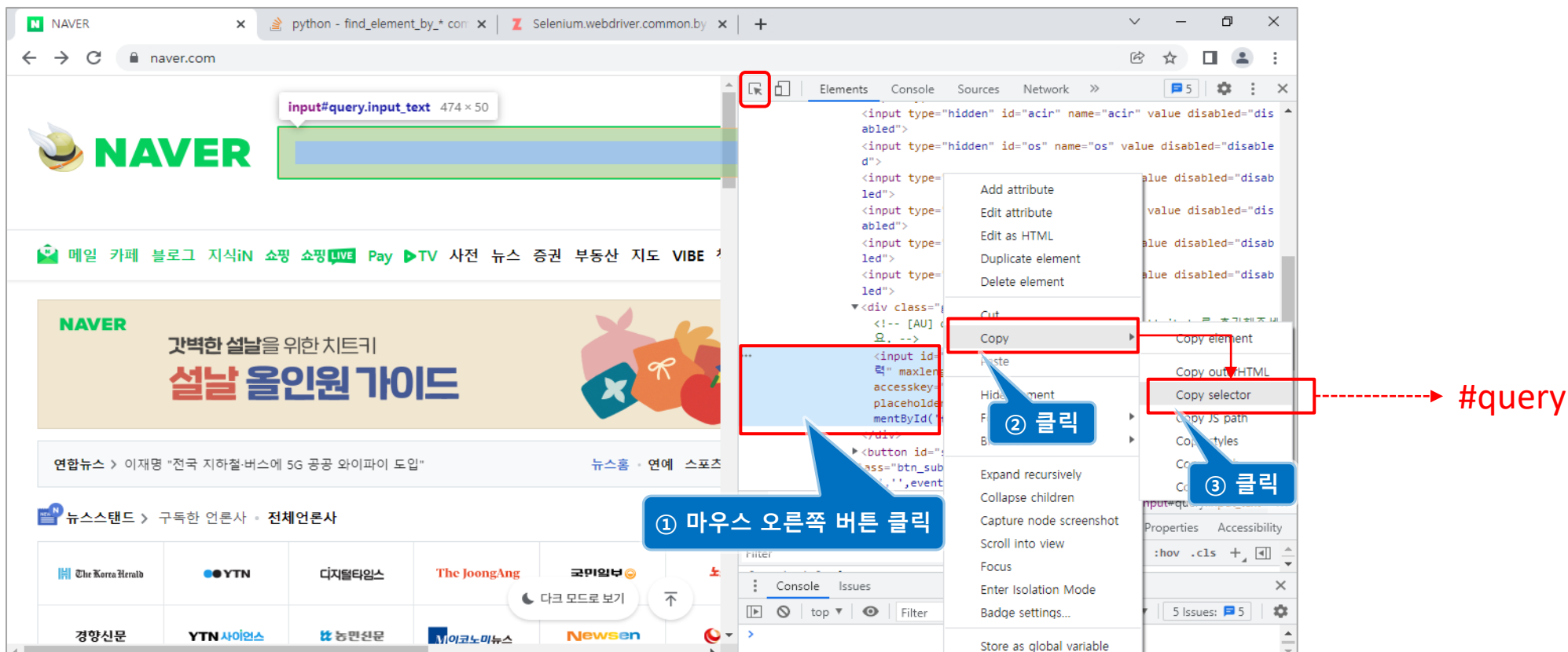
- 네이버에 접속한 후 [F12] 키를 눌러 개발자 도구를 실행
- **Inspector** 아이콘을 클릭하고 마우스로 네이버 웹페이지에 있는 검색창 버튼 클릭



● 네이버에서 방탄소년단 검색하기

➤ css selector 복사

- 검색창 버튼의 css selector를 복사하기 위해 개발자 도구에 선택된 html 소스를 마우스 오른쪽 버튼으로 클릭
- [Copy] – [Copy selector] 를 선택



웹 브라우저 제어하기



● 네이버에서 방탄소년단 검색하기

➤ 검색창에 문자 입력하기

- 복사한 css selector를 `driver.find_element(By.CSS_SELECTOR, " ")` 함수의 " " 안에 붙여 넣음

```
import selenium
selenium 모듈 불러오기
from selenium import webdriver
from selenium.webdriver.common.by import By
URL = "https://www.naver.com"
driver = webdriver.Chrome("chromedriver")
driver.get(URL)
input_tag = driver.find_element(By.CSS_SELECTOR, "#query")
input_tag.send_keys("방탄소년단")
input_tag.send_keys("Wn")
driver.implicitly_wait(3)
```

```
# webdriver 모듈 불러오기
# selenium의 by 클래스 불러오기
# 네이버 url을 URL 변수에 저장
# driver 객체 생성
# URL에 저장된 주소를 크롬에서 열기
# 검색창으로 이동하여
# 검색창에 문자 입력
# enter key 입력
# 최대 3초 대기(3초 지나면 다음 코드 수행)
```

