

# ⋮ 문제해결을 위한 데이터 분석 및 시각화

데이터 수집을 위해 파이썬으로 배우는 웹 크롤러

한성대학교 노은희 교수

“미래로 향하는 새로운 이정표”



# 오늘의 학습

---

## 학습내용

- 웹 크롤러를 이해하고 웹 크롤러를 사용하여 데이터를 수집하는 방법을 익힙니다.

## 용어 이해

---

**웹 크롤러**(web crawler)는 조직적, 자동화된 방법으로 월드 와이드 웹을 탐색하는 컴퓨터 프로그램  
웹 크롤러가 하는 작업을 '웹 크롤링'(web crawling)

웹 페이지를 가져와서 그 안에서 데이터를 추출하는 기술

웹 크롤링은 인터넷에 있는 웹페이지를 방문하여 페이지의 자료를 자동으로 수집하는 작업 의미

웹 스크래핑은 다양한 웹사이트로부터 데이터를 추출하는 기술을 의미

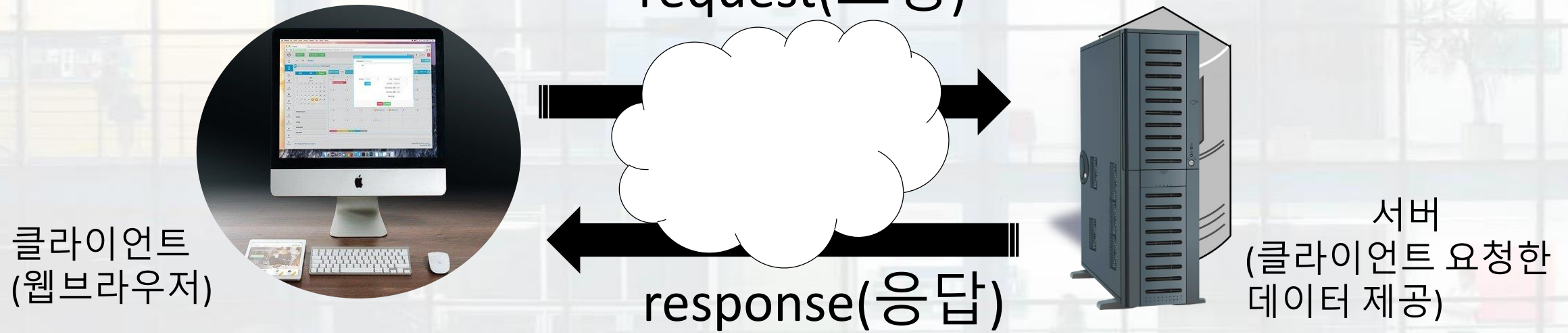
웹 파싱은 웹 상의 자연어, 컴퓨터 언어 등의 일련의 문자열들을 분석하는 프로세스



## 크롤러를 만들기 위한 준비

크롤러는 인터넷을 돌아다니면서 데이터를 수집하는 프로그램

- 서버와 클라이언트 개념 이해
- 웹브라우저 작동방식 이해
- 언어(파이썬)



컴퓨터에서 웹 브라우저로 인터넷을 통해 웹서버에 HTTP형식으로 원하는 정보를 요청(request)  
웹서버가 HTTP형식으로 응답(response)해 HTML파일을 보내준다.

- HTTP(HyperText Transfer Protocol)의 약자로 인터넷 상에서 HTML 문서의 정보를 주고받을 수 있도록 만든 프로토콜 (전송규약)
- HTML : HyperText Markup Language로 웹 페이지의 구조적 구성을 위한 언어
- 웹페이지 : 웹상에 있는 HTML로 구성된 개별 문서

# URL

데이터를 보내기 위해 URL 구조를 이해하기

URL(Uniform Resource Locator)는 네트워크상에서 자원을 요청하는 규약

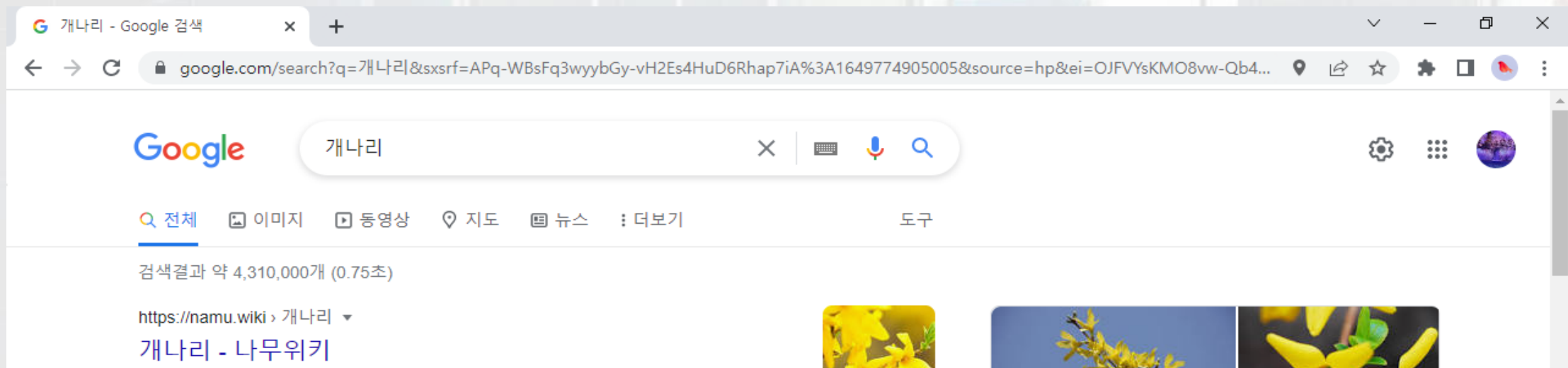
## URL구조

프로토콜://주소 또는 IP:포트번호/리소스 경로?쿼리 스트링

포트번호 : 생략가능, http의 경우 80, https는 443사용

리소스경로 : 클라이언트가 요청하는 리소스의 경로

쿼리스트링 : 클라이언트가 서버에 보내는 데이터





# 크롤링할때 주의할 점

---

## 크롤링을 할 때 주의할 점

웹사이트에서 크롤링봇 접근을 Disallow 하는 페이지는 크롤링을 해서는 안된다. 이는 처벌을 받을 수 있다.  
최상위 도메인주소 뒤에 /robots.txt를 입력하면 접근 허용 여부 콘텐츠를 확인

# 크롤링

## 크롤링을 할 때 주의할 점

- 웹사이트에서 크롤링봇 접근을 Disallow 하는 페이지는 크롤링을 해서는 안된다. 이는 처벌을 받을 수 있다.
- 자동 크롤링 로봇은 사이트 방문시 로봇배제표준 설정파일(robots.txt)를 확인한 후 이를 준수하여 콘텐츠를 수집  
**로봇의 크롤링 허가 여부를 명시해 놓은 파일"**
- 최상위 도메인주소 뒤에 /robots.txt를 입력하면 접근 허용 여부 콘텐츠를 확인

<https://www.google.com/robots.txt>

- Disallow 라고 되어있는 하위 디렉토리 페이지들에서는 크롤링을 할 수 없다.
- \* 크롤링 허용과 저작권 문제는 또 다른 사안이니 주의

← → ↻ 🔒 google.com/robots.txt

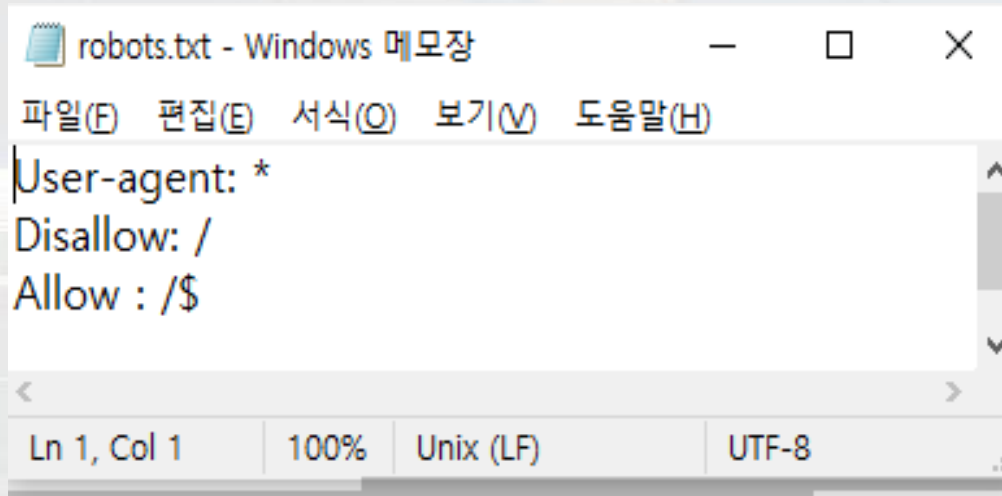
```
User-agent: *
Disallow: /search
Allow: /search/about
Allow: /search/static
Allow: /search/howsearchworks
Disallow: /sdch
Disallow: /groups
Disallow: /index.html?
Disallow: /?
Allow: /?hl=
Disallow: /?hl=*&
Allow: /?hl=*&gws_rd=ssl$
Disallow: /?hl=*&gws_rd=ssl
Allow: /?gws_rd=ssl$
Allow: /?pt1=true$
Disallow: /imgres
Disallow: /u/
Disallow: /preferences
Disallow: /setprefs
Disallow: /default
Disallow: /m?
Disallow: /m/
Allow: /m/finance
...
```

User-agent:로봇의 이름

Allow: 허용

Disallow: 비허용 (만약 이 부분이 비어있으면 모두 허용)

# 크롤링



```
robots.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
User-agent: *
Disallow: /
Allow : /$
Ln 1, Col 1 100% Unix (LF) UTF-8
```

Disallow는 자동화 프로그램의 접근이 허용되지 않는 부분이고, Allow는 허용되는 부분  
/는 모든 페이지 /\$는 첫페이지 의미, \*는 모든것

네이버는 첫 접속 페이지를 제외하고 모든 페이지에서 웹 크롤링 접근을 제한함

User-agent : 로봇의 이름

Allow : 허용

Disallow : 비허용(이 부분이 비어 있으면 모두 허용)





---

# 데이터 수집

## 크롤링을 위해 필요한 라이브러리

요청모듈  
requests

웹페이지를 요청하는 라이브러리로 파이썬에서 웹 데이터를 받아올 때 가장 많이 사용하는 모듈

파싱모듈  
Bs4모듈의  
BeautifulSoup()함  
수 사용

요청모듈로 가져온 html코드를 파이썬이 쓸 수 있도록 코드로 변환

## 크롤링을 위한 준비\_라이브러리 확인 및 설치

BeautifulSoup, Requests 패키지가 설치되었는지 확인

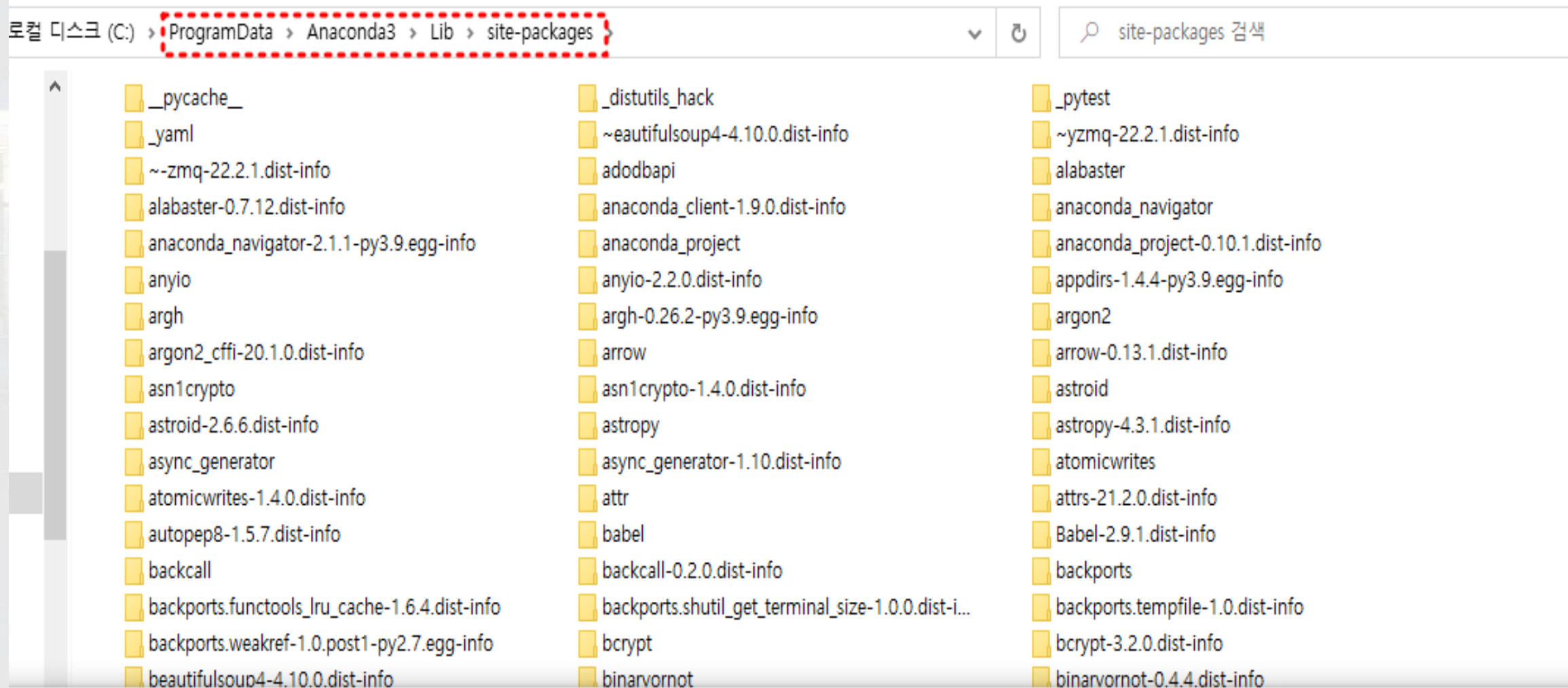
```
Anaconda Powershell Prompt (Anaconda3)
(base) PS C:\Users\노은희> pip list
```

(base) PS C:\Users\사용자명> pip list

```
backcall 0.2.0
backports.functools-lru-cache 1.6.4
backports.shutil-get-terminal-size 1.0.0
backports.tempfile 1.0
backports.weakref 1.0.post1
bcrypt 3.2.0
beautifulsoup4 4.10.0
```

```
Anaconda Powershell Prompt (Anaconda3)
QtPy 1.10.0
regex 2021.8.3
requests 2.26.0
rope 0.19.0
Rtree 0.9.7
ruamel-yaml-conda 0.15.100
scikit-image 0.18.3
scikit-learn 0.24.2
scikit-learn-intelex 2021.20210714.120553
scipy 1.7.1
seaborn 0.11.2
selenium 4.1.3
```

## 설치된 라이브러리





## 크롤링을 위한 준비

BeautifulSoup, Requests 패키지 설치

```
Anaconda Powershell Prompt (Anaconda3)
(base) PS C:\Users\노은희> pip install requests

Anaconda Powershell Prompt (Anaconda3)
(base) PS C:\Users\노은희> pip install beautifulsoup4
```

**BeautifulSoup:** 웹 페이지의 정보를 쉽게 스크랩할 수 있도록 기능을 제공하는 라이브러리  
웹사이트 내의 html코드를 긁어오고 본격적인 데이터 추출을 하기 위함

**Beautiful Soup**은 **HTML** 및 **XML** 문서를 구문 분석하기 위한 **Python** 패키지  
**HTML**에서 데이터를 추출하는 데 사용할 수 있는 구문 분석 된 페이지에 대한  
구문 분석 트리를 만들며, 웹 스크래핑에 유용

**Requests:** HTTP 요청을 보낼 수 있도록 기능을 제공하는 라이브러리

## 크롤링을 위한 준비

```
Anaconda Powershell Prompt (Anaconda3)

(base) PS C:\Users\User> pip install response
Collecting response
  Downloading response-0.5.0-py3-none-any.whl (14 kB)
Requirement already satisfied: matplotlib>=2.2.0 in c:\programdata\anaconda3\lib\site-packages (from response) (3.4.3)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-packages (from response) (1.7.1)
Requirement already satisfied: numpy
Requirement already satisfied: python
0>response) (2.8.2)
Requirement already satisfied: cycl
nse) (0.10.0)
Requirement already satisfied: kiwi
response) (1.3.1)
Requirement already satisfied: pill
onse) (8.4.0)
Requirement already satisfied: pypa
esponse) (3.0.4)
Requirement already satisfied: six
response) (1.16.0)
Installing collected packages: resp
Successfully installed response-0.5.0
(base) PS C:\Users\User> pip list
Package
-----
alabaster
anaconda-client
anaconda-navigator
anaconda-project
anyio
appdirs
argh
DarkStyle
qstylizer
QtAwesome
qtconsole
QtPy
regex
requests
response
rope
Rtree
ruamel-yaml-conda
scikit-image
scikit-learn
scikit-learn-intelex
scipy
seaborn
Send2Trash
setuptools
simplegeneric
singledispatch
sip
six
sniffio
snowballstemmer
sortedcollections
sortedcontainers
soupsieve
Sphinx
sphinxcontrib-applehelp
sphinxcontrib-devhelp
```

## 크롤링을 위한 준비

### find() 함수

find() 함수는 조건을 만족하는 태그를 하나만 가져오는 함수

### find\_all() 함수

find\_all() 함수는 원하는 태그가 여러 개 있을 경우 해당하는 태그를 한꺼번에 가져오는 함수

### 여러 가지의 태그를 찾아야 하는 상황

찾고 싶은 태그를 리스트로 **find\_all()** 함수에 넣어주면 된다.

```
# find_all 함수로 여러 가지 태그를 조회하고 싶다면?  
soup.find_all(['p', 'img'])
```

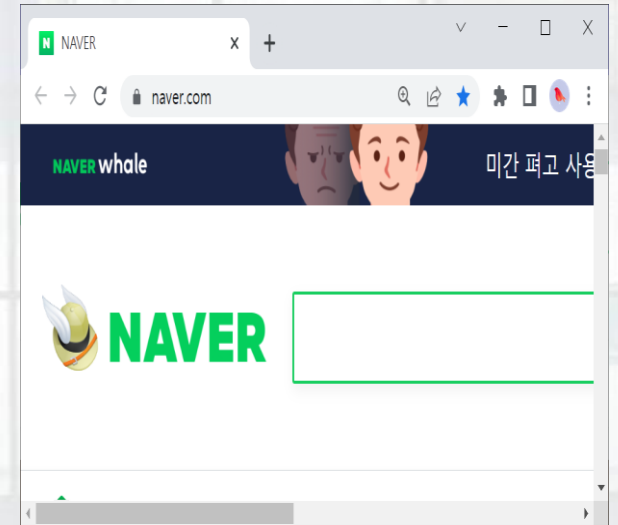
```
[<p align="center">text 1</p>,  
 <p align="center">text 2</p>,  
 <p align="center">text 3</p>,  
 ]
```

# 웹 브라우저로 웹사이트 접속하기

## 웹 브라우저로 웹사이트에 접속하기

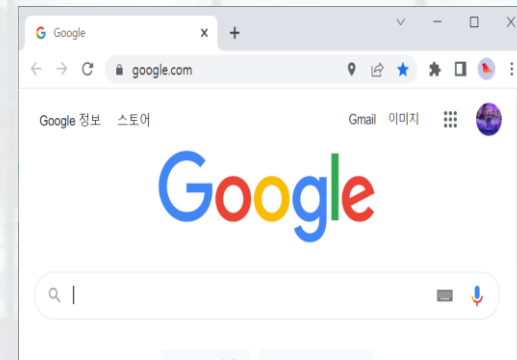
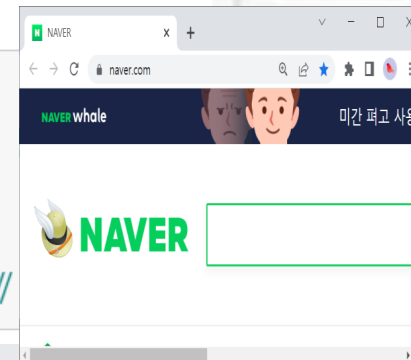
```
1 import webbrowser    # webbrowser 모듈 불러오기
2 url = 'www.naver.com'
3 webbrowser.open(url)  # 변수 url에 지정된 웹사이트(www.naver.com)에 접속
```

True



## 여러개의 웹 사이트에 접속하기

```
1 import webbrowser
2
3 urls = ['www.naver.com', 'www.google.com']
4 for url in urls:
5     webbrowser.open(url)  # 변수 url에 지정된 웹사이트(www.naver.com)에
```







## 응답데이터 Response Content

속성	설명
<b>status_code</b>	응답 상태를 확인
<b>headers</b>	headers정보를 확인
<b>cookies</b>	cookies정보를 확인
<b>encoding</b>	데이터 인코딩을 확인
<b>text</b>	'str' 타입의 데이터
<b>content</b>	bytes 타입의 데이터
<b>.json()</b>	dict 타입의 데이터 일 경우 사용

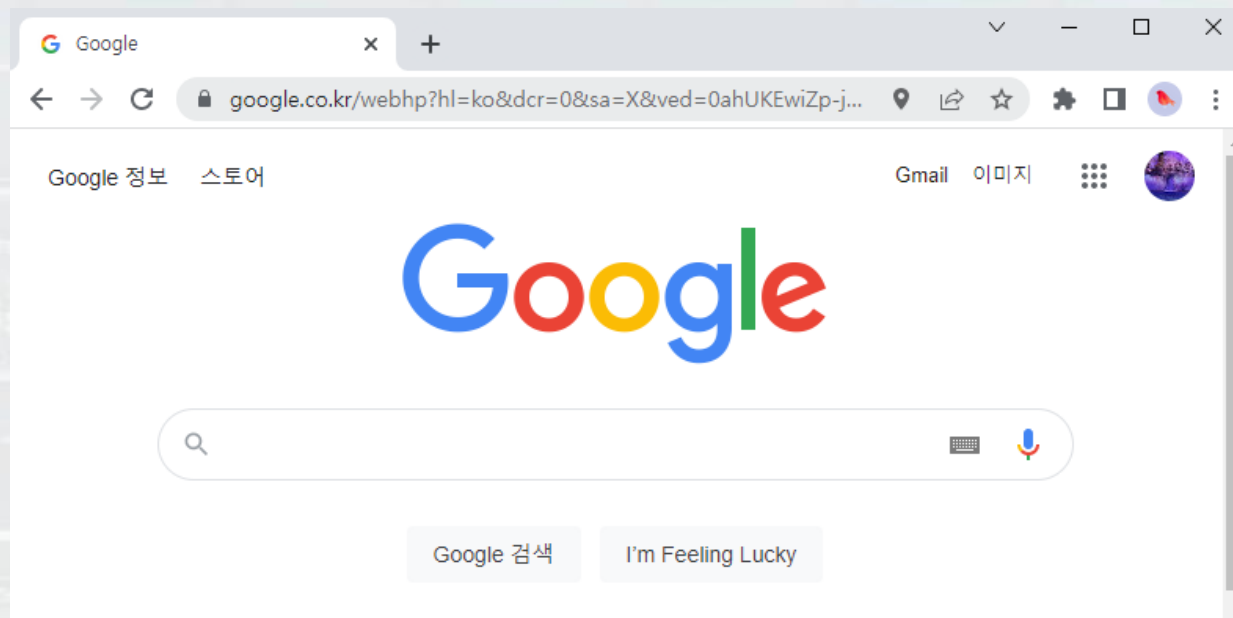
# request 라이브러리

## request 라이브러리 사용

```
1 import requests
2 r = requests.get('http://www.google.com')
3 r
```

<Response [200]>

접속이 잘 되면 Response [200]반환



## request 라이브러리

```
1 import requests
2 r = requests.get('http://www.google.com').text
3 r[0:100]
```

```
'<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="ko"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/googleg/1x/googleg_standard_color_128dp.png" itemprop="image"><title>Google</title><script nonce="a6VQ8Ahr i03WUvJh6F1B5g==">(function(){window.google={kEl:₩'p6NNYrH0AufQ2roPh0uDkA8₩',kEXPl:₩'0,1302536,56873,1709,4349,207,2414,2390,925,1391,383,246,5,1354,4013,1237,1122516,1197774,627,380090,16114,28684,17572,4858,1362,9291,3026,2817,14765,4020,978,13227,3848,4192,6431,7431,15309,5081,885,709,1278,2742,149,562,541,840,6297,3514,606,2025,1775,520,14670,3227,2845,7,17450,15768,552,1851,15756,3,346,230,6459,149,13975,4,1528,2304,7039,25073,2658,7355,32,13628,4437,9358,7428,5815,2542,4094,4052,3,3541,1,14263,2544,25347,2,14022,1931,4317,1272,743,5853,10463,1160,5679,1020,2378,2721,18234,9,2,6,7773,4567,6259,9497,13921,1249,4591,2,6,1239,11862,3106,1538,2794,19,4658,1412,1395,445,2,2,1,6394,565,3831,513,13476,14'
```

# BeautifulSoup 라이브러리로 파싱(Parsing)하기

---

```
from bs4 import BeautifulSoup
```

Python에서 XML parser로서 주로 이용되는 패키지는 lxml

Python의 두 가지 주요 HTML 구문 분석 라이브러리는 lxml과 BeautifulSoup



# HTTP 응답 코드

## 2xx 성공

**200:** 클라이언트의 요청을 정상적으로 수행함.

**201:** 클라이언트에게 생성 작업을 요청 받았고, 생성 작업을 성공함.

**204:** 요청은 성공 했지만 응답할 콘텐츠가 없음.

## 3xx 리다이렉션

**301:** 클라이언트가 요청한 리소스에 대한 **URI**가 영구적으로 변경되었을 때 사용함.

**302:** **301**과 같으나 임시적으로 주소가 바뀌었을 경우 사용함.

**304:** 이전에 방문했을 때의 요청 결과와 다르지 않을 경우 사용함. 캐시된 페이지를 그대로 사용.

**307:** 임시 페이지로 리다이렉트.

# HTTP 응답 코드

## 4xx 클라이언트 오류

- 400:** 클라이언트가 올바르지 못한 요청을 보냄.
- 401:** 로그인을 하지 않아 페이지를 열 권한이 없음.
- 403:** 금지된 페이지, 로그인을 하든 안하든 접근할 수 없음. (관리자 페이지)
- 404:** 찾을 수 없는 페이지, 주소를 잘 못 입력했을 때 사용함.
- 403** 대신에 사용할 수도 있음. (해커들의 공격을 방지하고자 페이지가 없는 것처럼 위장함)
- 408:** 요청 시간이 초과됨.
- 409:** 서버가 요청을 처리하는 과정에서 충돌이 발생한 경우. (회원가입 중 중복된 아이디인 경우)
- 410:** 영구적으로 사용할 수 없는 페이지.

## 5xx 서버 오류

- 501:** 해당 요청을 처리하는 기능이 만들어지지 않음.
- 502:** 서버로 가는 요청이 중간에서 유실된 경우.
- 503:** 서버가 터졌거나 유지 보수 중  
(유지 보수 중일때는 유지 보수중이라는 것을 알려주는 페이지로 전송해주는 것이 좋음)
- 504:** 서버 게이트웨이에 문제가 생겨 시간 초과가 된 경우.
- 505:** HTTP 버전이 달라 요청이 처리할 수 없음.

# BeautifulSoup

## Parsing하기 : HTML코드로 분석하기

### 데이터 찾고 추출하기

```
from bs4 import BeautifulSoup
# 테스트용 html 코드
html = """<html><body><div><span>
    <a href = http://www.naver.com>naver</a>
    <a href = http://www.google.com>google</a>
</span></div></body></html>"""

soup = BeautifulSoup(html, 'lxml') #lxml은 HTML소스를 처리하기 위한 vktj(parser)
soup
```

```
<html><body><div><span> <a href="http://www.naver.com">naver</a> <a href="http://www.google.com">google</a> </span></div></body></html>
```

# Zip 함수 사용법

zip()은 동일한 개수로 이루어진 자료형을 묶어주는 함수

```
IDLE Shell 3.10.2
File Edit Shell Debug Options Window Help
Python 3.10.2 (tags/v3.10.2:a5
MD64) on win32
Type "help", "copyright", "credit
>>> list(zip("abc", "def"))
[('a', 'd'), ('b', 'e'), ('c', 'f')]
>>> list(zip([1,2,3], [4,5,6]))
[(1, 4), (2, 5), (3, 6)]
>>>
```





## 웹크롤링할때 주의사항

---

- HTML 소스코드에서 데이터를 가져오기 위해 특정 규칙을 발견하기
- 파이썬 코드를 이용해서 크롤링을 할 경우 웹 사이트에 빈번하게 접근하면 접근 차단
- 웹 사이트는 예고 없이 변경
- 인터넷상에 공개된 데이터라도 저작권이 있는 경우 존재, 저작권 침해 여부 확인



# 실습하기

---

# 웹 크롤링

## requests 라이브러리 사용

```
import requests  
r = requests.get('http://www.google.com')  
r
```

<Response [200]>

```
import requests  
r = requests.get('http://www.google.com')  
r.text[0:100]
```

'<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="ko"><head><meta content'

## HTML 소스코드를 분석하고 처리하기

```
from bs4 import BeautifulSoup
# 테스트용 html코드
html = """
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Document</title>
</head>
<body>
  <a href = "http://kosis.kr" >국가통계 포털</a><br>
  <a href = "http://www.data.go.kr">공개 데이터 포털</a><br>
  <a href = "http://data.seoul.go.kr"> 서울 열린데이터 광장</a>
</body>
</html>
"""

# BeautifulSoup를 이용해 html 소스 파싱
soup = BeautifulSoup(html, 'lxml')
soup
```

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Document</title>
</head>
<body>
<a href="http://kosis.kr">국가통계 포털</a><br/>
<a href="http://www.data.go.kr">공개 데이터 포털</a><br/>
<a href="http://data.seoul.go.kr"> 서울 열린데이터 광장</a>
</body>
</html>
```

```
print(soup.prettify())
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>
      Document
    </title>
  </head>
  <body>
    <a href="http://kosis.kr">
      국가통계포털
    </a>
    <br/>
    <a href="http://www.data.go.kr">
      공개 데이터 포털
    </a>
    <br/>
    <a href="http://data.seoul.go.kr">
      서울 열린데이터 광장
    </a>
  </body>
</html>
```



```
soup.find('a')
```

```
<a href="http://kosis.kr">국가통계포털</a>
```

```
soup.find('a').get_text()
```

```
'국가통계포털'
```

```
soup.find_all('a')
```

```
[<a href="http://kosis.kr">국가통계포털</a>,  
<a href="http://www.data.go.kr">공개 데이터 포털</a>,  
<a href="http://data.seoul.go.kr">서울 열린데이터 광장</a>]
```

```
site_names = soup.find_all('a')  
for site_name in site_names:  
    print(site_name.get_text())
```

```
국가통계포털  
공개 데이터 포털  
서울 열린데이터 광장
```

```
from bs4 import BeautifulSoup
# 테스트용 Html 코드
html2 = """
<html>
  <head>
    <title> 작품과 작가</title>
  </head>
  <body>
    <h1>책 정보</h1>
    <p id = "book_title"> 이것이 데이터 분석이다 with 파이썬</p>
    <p id = "author">윤기태

    <p id = "book_title"> 데이터 분석을 위한 파이썬 철저 입문</p>
    <p id = "author">최은석

    <p id = "book_title"> 모두의 데이터 분석 with 파이썬</p>
    <p id = "author">송선리, 이현아</p>
  </body>
</html>
"""

soup2 = BeautifulSoup(html2, 'lxml')
soup2.title
```

<title> 작품과 작가</title>

```
soup2.body
```

```
<body>
<h1>책 정보</h1>
<p id="book_title"> 이것이 데이터 분석이다 with 파이썬 </p>
<p id="author">윤기태

    </p><p id="book_title"> 데이터 분석을 위한 파이썬 철저 입문 </p>
<p id="author">최은석

    </p><p id="book_title"> 모두의 데이터 분석 with 파이썬 </p>
<p id="author">송선리, 이현아</p>
</body>
```

```
soup2.body.h1
```

```
<h1>책 정보</h1>
```

```
soup2.find_all('p')
```

```
[<p id="book_title"> 이것이 데이터 분석이다 with 파이썬</p>,<br><p id="author">윤기태<br></p>,<br><p id="book_title"> 데이터 분석을 위한 파이썬 철저 입문</p>,<br><p id="author">최은석<br></p>,<br><p id="book_title"> 모두의 데이터 분석 with 파이썬</p>,<br><p id="author">송선리, 이현아</p>]
```

```
soup2.find_all('p',{'id':"book_title"})
```

```
[<p id="book_title"> 이것이 데이터 분석이다 with 파이썬</p>,<br><p id="book_title"> 데이터 분석을 위한 파이썬 철저 입문</p>,<br><p id="book_title"> 모두의 데이터 분석 with 파이썬</p>]
```

```
soup2.find_all('p',{'id':"author"})
```

```
[<p id="author">윤기태<br></p>,<br><p id="author">최은석<br></p>,<br><p id="author">송선리, 이현아</p>]
```

```
from bs4 import BeautifulSoup
soup2 = BeautifulSoup(html2, 'lxml')

book_titles = soup2.find_all('p', {"id": "book_title"})
authors = soup2.find_all('p', {"id": "author"})

for book_title, author in zip(book_titles, authors):
    print(book_title.get_text() + '/' + author.get_text())
```

이것이 데이터 분석이다 with 파이썬/윤기태

데이터 분석을 위한 파이썬 철저 입문/최은석

모두의 데이터 분석 with 파이썬/송선리, 미현아



```

from bs4 import BeautifulSoup
html3 = """
<html>
  <head>
    <title> 작품과 작가</title>
  </head>
  <body>
    <h1>책 정보</h1>
    <p id = "book_title" class = "book"> 이것이 데이터 분석이다 with 파이썬</p>
    <p id = "author">윤기태

    <p id = "book_title" class = "book"> 데이터 분석을 위한 파이썬 철저 입문</p>
    <p id = "author">최은석

    <p id = "book_title" class = "book"> 모두의 데이터 분석 with 파이썬</p>
    <p id = "author">송선리, 이현아</p>

    <a href = "http://www.yes24.com" class = "bookstore"> yes24 </a>
  </body>
</html>
"""
soup3 = BeautifulSoup(html3, 'lxml')
soup3.select("a.bookstore")

```

```
[<a class="bookstore" href="http://www.yes24.com"> yes24 </a>]
```

```
soup3.select('html body a')
```

```
[<a class="bookstore" href="http://www.yes24.com"> yes24 </a>]
```

```
soup3.select('body a')
```

```
[<a class="bookstore" href="http://www.yes24.com"> yes24 </a>]
```

```
soup3.select('a')
```

```
[<a class="bookstore" href="http://www.yes24.com"> yes24 </a>]
```

```
soup3.select('html a')
```

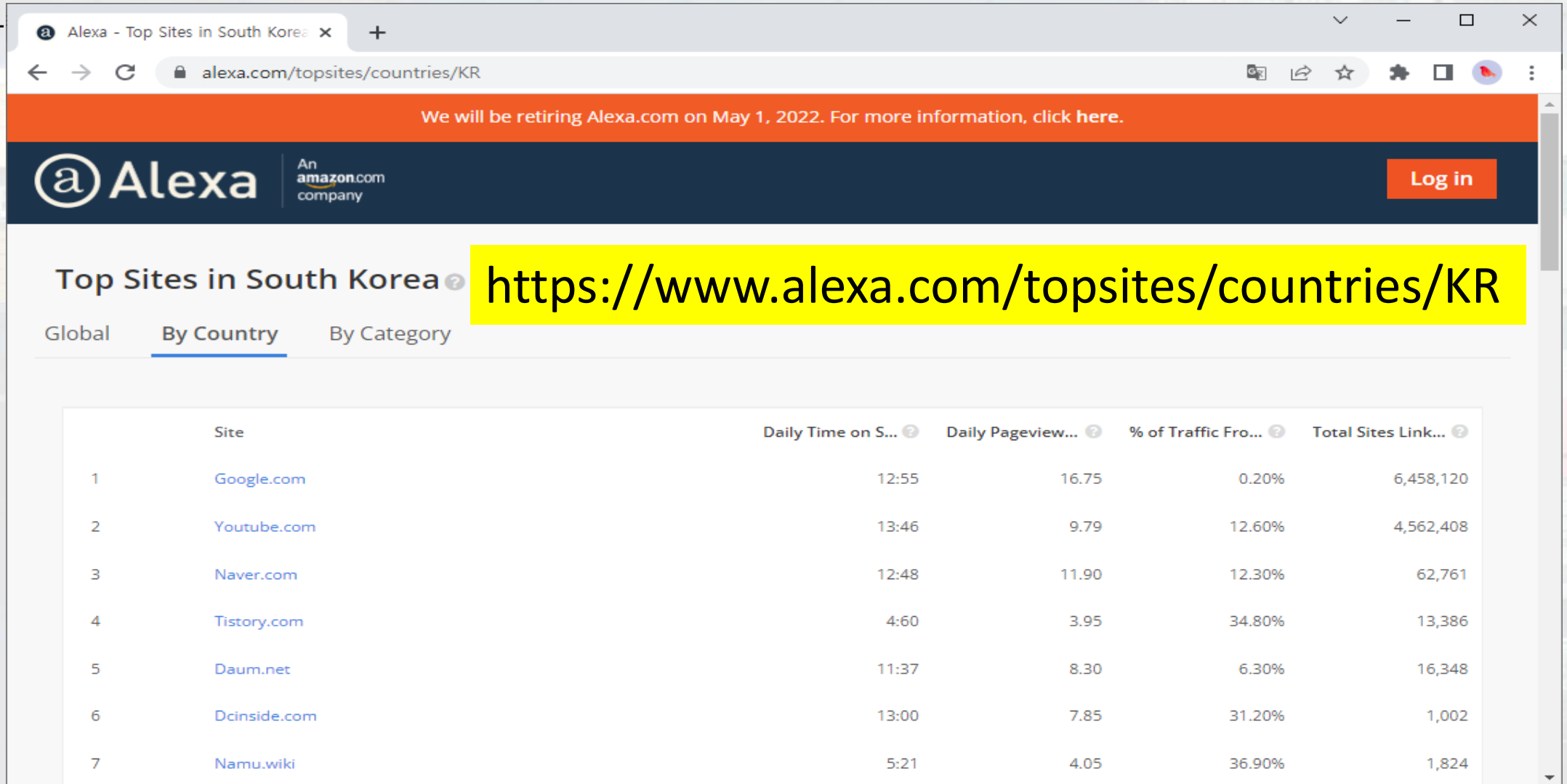
```
[<a class="bookstore" href="http://www.yes24.com"> yes24 </a>]
```



# 웹사이트 순위 실습하기

# 알렉사 사이트에서 순위 데이터 가져오기

'알렉사'는 전 세계 인터넷 사이트의 트래픽을 분석하고, 그 통계치와 각종 순위에 대한 정보를 제공하는 사이트



We will be retiring Alexa.com on May 1, 2022. For more information, click [here](#).

**Alexa** An amazon.com company [Log in](#)

## Top Sites in South Korea ?

[Global](#) [By Country](#) [By Category](#)

	Site	Daily Time on S... ?	Daily Pageview... ?	% of Traffic Fro... ?	Total Sites Link... ?
1	<a href="#">Google.com</a>	12:55	16.75	0.20%	6,458,120
2	<a href="#">Youtube.com</a>	13:46	9.79	12.60%	4,562,408
3	<a href="#">Naver.com</a>	12:48	11.90	12.30%	62,761
4	<a href="#">Tistory.com</a>	4:60	3.95	34.80%	13,386
5	<a href="#">Daum.net</a>	11:37	8.30	6.30%	16,348
6	<a href="#">Dcinside.com</a>	13:00	7.85	31.20%	1,002
7	<a href="#">Namu.wiki</a>	5:21	4.05	36.90%	1,824

# 웹 사이트 코드 분석하기

F12키를 누르고 요소 검사를 합니다.

The screenshot shows a web browser window with the URL `alexa.com/topsites/countries/KR`. The page displays a list of top sites in South Korea, with Google.com at the top. A tooltip shows the dimensions of the Google.com link as 66.69 x 17. The DevTools element inspector is open on the right, showing the HTML structure of the page. The selected element is a `div` with class `td DescriptionCell`, which contains a `p` element with a link to `/siteinfo/google.com`. The breadcrumb trail at the bottom of the DevTools panel shows the path: `div.tableContainer > div.listings.table > div.tr.site-listing > div.td.DescriptionCell`.

Dimensions: iPhone 12 Pro 390 x 844 75% Custom

We will be retiring Alexa.com on May 1, 2022. For more information, click here.

**Alexa** An Amazon.com company

## Top Sites in South Korea

Global By Country By Category

1	<a href="#">Google.com</a>
2	<a href="#">Youtube.com</a>
3	<a href="#">Naver.com</a>
4	<a href="#">Tistory.com</a>
5	<a href="#">Daum.net</a>
6	<a href="#">Dcinside.com</a>
7	<a href="#">Namu.wiki</a>
8	<a href="#">Kakao.com</a>

66.69 x 17

DevTools is now available in Korean!

Always match Chrome's language Switch DevTools to Korean Don't show again

Elements Console Sources

```
<div class="td DescriptionCell"> == $0
  <p class>
    <a href="/siteinfo/google.com">Google.com
  </a>
  </p>
</div>
<div class="td right">...</div>
<div class="td right">...</div>
<div class="td right">...</div>
<div class="td right">...</div>
</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
<div class="tr site-listing">...</div>
```

div.tableContainer div.listings.table div.tr.site-listing div.td.DescriptionCell

Styles Computed Layout Event Listeners DOM Breakpoints Properties

Filter :hov .cls +



## 웹 사이트의 순위를 추출하는 코드 작성하기

```
1 import requests
2 from bs4 import BeautifulSoup
3
4 url = "https://www.alex.com/topsites/countries/KR"
5
6 ranking = requests.get(url).text
7
8 soup = BeautifulSoup(ranking, 'lxml')
9
10 website_ranking = soup.select('p a')
11 website_10_ranking = website_ranking[1:11]
12
13 results = []
14 for rank in website_10_ranking:
15     result = rank.get_text()
16     results.append(result)
17
18 print('[----- 웹 사이트 순위(한국)-----]')
19
20 for k in range(0,10,1):
21     print(k+1, ":", results[k])
```

[출력결과]

```
[----- 웹 사이트 순위(한국)-----]
1 : Google.com
2 : Youtube.com
3 : Naver.com
4 : Tistory.com
5 : Daum.net
6 : Dcinside.com
7 : Namu.wiki
8 : Kakao.com
9 : Instagram.com
10 : Coupang.com
```