

# 데이터 수집 · 분석에 필요한 실전형 웹크롤링

Web Scraping & Web Crawling



강사 최희윤

# 주요 키워드

# 웹 크롤링 관련 주요 키워드

---

1. 웹크롤링 개요 – Day1
2. 웹 브라우저 & HTML 이해하기 – Day1
3. Request & Response – Day1
4. BeautifulSoup – Day-2~3
5. Selenium – Day3~4
6. Scrapy – Day4~5

# 웹크롤링 개요

# 웹 크롤링을 하는 이유

---

매 번 **수동**으로 인터넷에서 데이터를 추출 하고 엑셀로 정리하는 일이 **번거로움**  
데일리로 데이터가 생성된다면 매일 들어가서 데이터를 추출해야 함

하지만,

웹 크롤링으로 **동적 제어**를 통해 특정 페이지에 들어가서 데이터를 추출하여  
엑셀로 저장되도록 **자동화** 한다면  
이러한 번거로움을 줄일 수 있음

# 웹 크롤링을 하는 이유

---

웹 자동화를 통해

- ✓ 주가의 변동을 관찰하다가 특정 패턴이 보이면 이를 통해 뉴스 기사 작성,
  - ✓ 수많은 사이트에 접속해 데이터 수집,
  - ✓ 엑셀 파일의 특정 인물들에게만 메일 전송,

등 **웹을 활용한 대부분의 일들이 웹 자동화를 통해 해결 가능**

# 웹 크롤링이란?

---

- ✓ 웹 자동화를 위해 쓰이는 기술
- ✓ 웹페이지에서 원하는 데이터를 추출해 가공하는 것
- ✓ 많은 정보를 활용하고 분석하기 위해 데이터를 수집하는 행위

## 사용 파이썬 모듈

Requests, BeautifulSoup, Selenium, Scrapy

# 윤리 및 법적 고려사항

## robots.txt 파일이란?

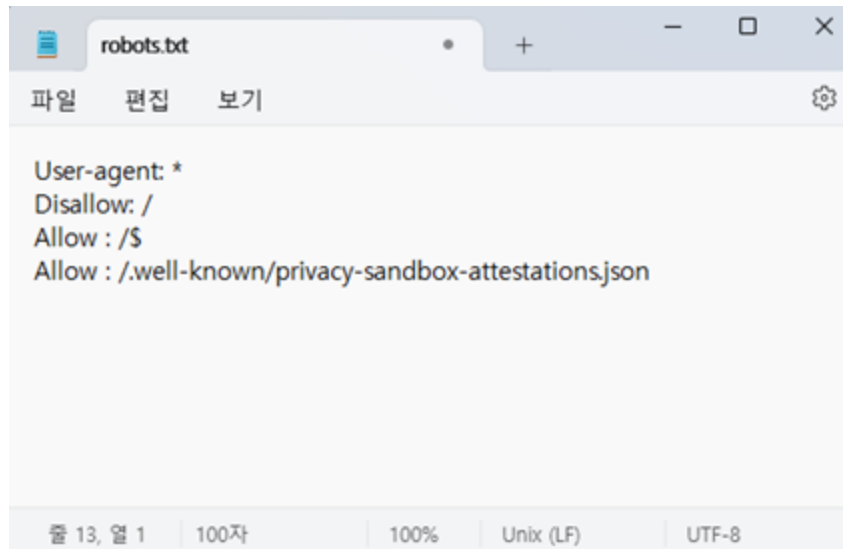
웹사이트에서 크롤러가 접근할 수 있는 페이지와 접근할 수 없는 페이지를 지정하는 파일

웹사이트의 robots.txt 파일을 확인하여 크롤링이 허용되는지 확인.  
또한 크롤링 시 서버에 과부하를 주지 않도록 요청 빈도와 타이밍 조절 필요

## robots.txt 파일을 확인하는 방법

웹사이트 주소 뒤에 /robots.txt를 추가하여 브라우저에서 확인

예를 들어, <http://www.naver.com/robots.txt>



```
User-agent: *
Disallow: /
Allow: /$
Allow: /.well-known/privacy-sandbox-attestations.json
```



# 윤리 및 법적 고려사항

## robots.txt 파일이란?

웹사이트에서 크롤러가 접근할 수 있는 페이지와 접근할 수 없는 페이지를 지정하는 파일

### ✓ User-agent : \*

- 모든 웹 크롤러(크롤링 봇)에게 지시

### ✓ Disallow: /

모든 페이지에 대한 접근 금지

모든 크롤러가 사이트의 모든 페이지를 크롤링하는 것을 금지

### ✓ Allow: /\$

루트 페이지(예: <https://example.com/>)는 크롤링 허용

\$는 URL의 끝을 의미 (사이트의 루트페이지를 의미)

A. 루트 페이지(root page)는 웹사이트의 기본 시작 페이지를 의미.

일반적으로 루트 페이지는 도메인 이름만으로 접근할 수 있는 페이지

B. 사이트 주소가 <https://www.example.com>일 때,

루트 페이지는 <https://www.example.com/>가 됨

### ✓ Allow: /.well-known/privacy-sandbox-attestations.json

특정 파일(.well-known/privacy-sandbox-attestations.json)은 크롤링 허용

# 윤리 및 법적 고려사항

## robots.txt 파일이란?

웹사이트에서 크롤러가 접근할 수 있는 페이지와 접근할 수 없는 페이지를 지정하는 파일



즉, 네이버는 모든 페이지에 대한 접근 및 크롤링을 금지하고 있지만, 루트페이지와 특정 인증 파일에 대해서는 예외적으로 크롤링 허용

# 하이퍼 텍스트 & 웹브라우저

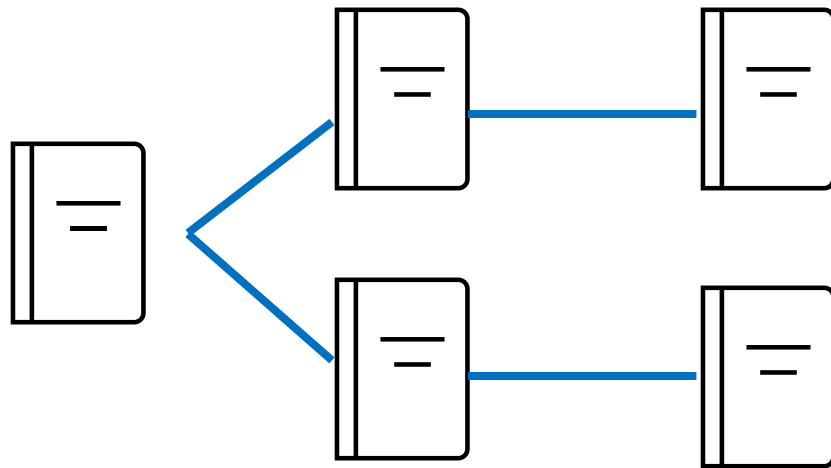
# 하이퍼텍스트란?

---



책을 읽을 때 우리가 모르는 정보가 있을 시  
참고 문헌을 따로 찾아 봐야하는 불편함이 있었음

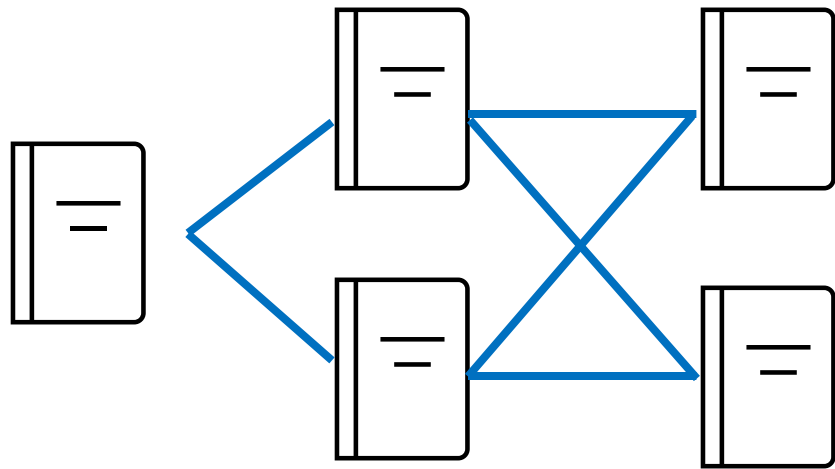
# 하이퍼텍스트란?



인터넷 발명 후

관련된 문서를 하나로 엮어내는 방법을 제안

## 하이퍼텍스트란?



여러 텍스트 페이지를 만든 후  
각 페이지 안에 다른 페이지에 대한 연결고리를 만듦

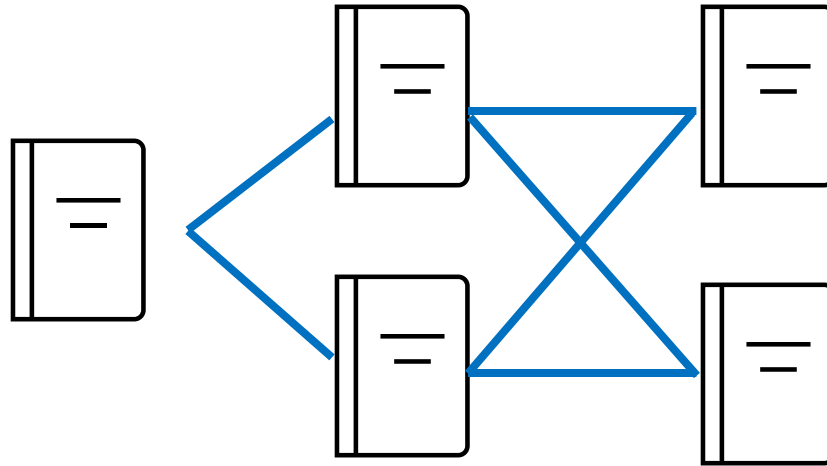
# 하이퍼텍스트란?

---

예시

[생성형 ai 위키피디아](#)

## 하이퍼텍스트란?



이러한 형태의 텍스트를 **하이퍼텍스트**라고 함



# 하이퍼텍스트란?

---

## Hypertext

참조를 통해 독자가 한 문서에서 다른 문서로 즉시 접근할 수 있는 텍스트

## 하이퍼링크란?

---

# Hyperlink

하이퍼텍스트안에서 직접 모든 형식의 자료를  
연결하고 가리킬 수 있는 참조 고리

# 하이퍼링크란?

---

## Hyperlink

하이퍼텍스트안에서 직접 모든 형식의 자료를  
연결하고 가리킬 수 있는 참조 고리

# 하이퍼텍스트란?

---

우리가 보는 위키피디아 같은 웹페이지들이  
하이퍼 텍스트로 구성돼있음을 알 수 있음

즉, 웹사이트는 모두 하이퍼텍스트의 일종

# 웹브라우저란?

---

하이퍼텍스트는 인터넷 상의 문서로써  
하이퍼텍스트를 보기 위해서는  
하이퍼텍스트를 보여주는 전용 소프트웨어가 필요

이 소프트웨어가 바로 **웹브라우저**

최초의 웹브라우저는 팀 버너스 리에 의해 1990년에 개발됨

# 웹브라우저란?

---



# 하이퍼텍스트 만들기

---

# HTML

**H**ypert**e**xt **M**arkup **L**anguage

하이퍼텍스트를 만드는 언어

# HTML로 간단한 Hypertext 만들어보기

---

예제파일

Web1\_1.html

Web1\_2.html



# HTML로 간단한 Hypertext 만들어보기

하이퍼링크(영어: `hyperlink` 또는 `link`, 문화어: 초연결, 하이퍼연결)는 `<a href="web1_2_학생용.html">하이퍼텍스트</a>` 문서 안에서 직접 모든 형식의 자료를 연결하고 가리킬 수 있는 참조 고리이다.

- ✓ `<>` : 태그(Tag)
  - HTML 구성 요소
  - `<a>` : a 태그
- ✓ **href :**
  - Hyperlink Reference
  - 어디로 연결시켜 줄지 적어주면 됨
  - 태그 사이의 '하이퍼텍스트' 글자를 누르면 href에 적어놓은 페이지로 이동하게 됨

# 웹의 확장

하이퍼링크(영어: `hyperlink` 또는 `link`, 문화어: 초연결, 하이퍼연결)는 `<a href="web1_2_학생용.html">하이퍼텍스트</a>` 문서 안에서 직접 모든 형식의 자료를 연결하고 가리킬 수 있는 참조 고리이다.

`<a href="web1_2_학생용.html">` 하이퍼텍스트 `</a>`

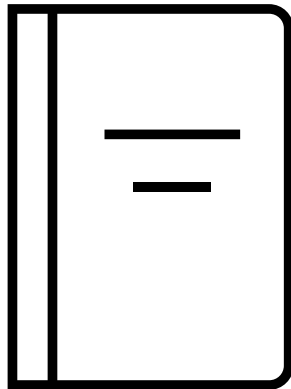


여기 href에 적힌 html은 내가 갖고 있는, 내가 작성한 하이퍼텍스트.

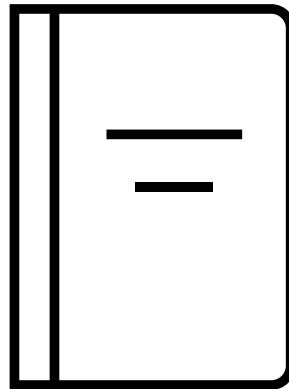
- a태그에서 href를 통해 내가 만든 html 파일을 연결할 수 있었음

# 웹의 확장

웹의 확장을 원함



내가 만든 하이퍼텍스트



남이 만든 하이퍼텍스트

# URL

---

`<a href="web1_2_학생용.html"> 하이퍼텍스트 </a>`



남이 만든 하이퍼텍스트라는 걸 명시해주기 위해  
남이 만든 하이퍼텍스트를 지칭할 명칭이 필요했음

이 명칭이 바로 **URL (Uniform Resource Locator)**

# URL

---

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

<https://www.youtube.com/>

<https://www.naver.com/>

<https://www.netflix.com/>

URL을 통해 남이 만든 웹페이지에 접근할 수 있게 됨

# 검색엔진

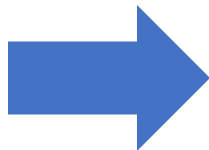
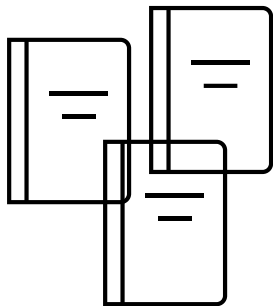
---

URL을 다른 사람들에게 알려주고, 다른 사람들의 URL을 찾을 수 있도록 함



# 검색 엔진 구동 방식

✓ 전 세계의 사람들이 만든 URL을 수집하는 방법?



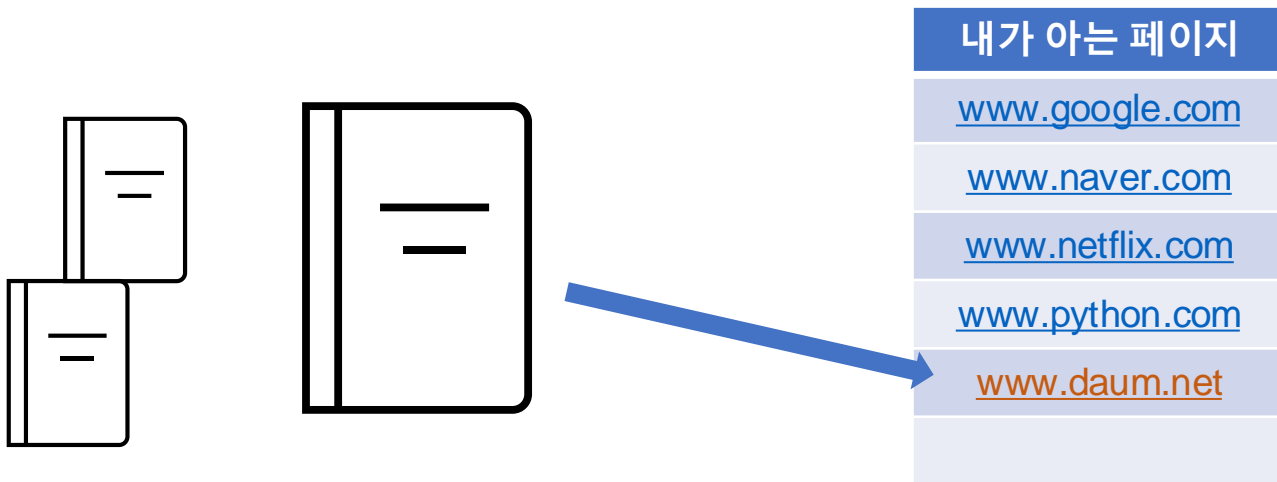
내가 아는 페이지
<a href="http://www.google.com">www.google.com</a>
<a href="http://www.naver.com">www.naver.com</a>
<a href="http://www.netflix.com">www.netflix.com</a>
<a href="http://www.python.com">www.python.com</a>

1. 내가 알고있는 페이지를 모음

2. 페이지 URL을 적어 놓음

# 검색 엔진 구동 방식

- ✓ 전 세계의 사람들이 만든 URL을 수집하는 방법



3. 내가 아는 URL에서 연결할 수 있는 url이 있으면  
목록에 추가



# 검색 엔진 구동 방식

내가 아는 페이지	키워드
<a href="http://www.google.com">www.google.com</a>	문서
<a href="http://www.naver.com">www.naver.com</a>	안에
<a href="http://www.netflix.com">www.netflix.com</a>	나온
<a href="http://www.python.com">www.python.com</a>	단어들, 이나
<a href="http://www.daum.net">www.daum.net</a>	키워드, 적어
	놓기

4. URL 페이지에 나온 단어 / 키워드 등을  
적어 놓기

# 검색 엔진 구동 방식

5. 검색 엔진을 통해 누군가가 키워드를 검색하면  
해당되는 URL을 소개시켜줌

내가 아는 페이지	키워드 (인덱싱)
<a href="http://www.google.com">www.google.com</a>	문서
<a href="http://www.naver.com">www.naver.com</a>	안에
<a href="http://www.netflix.com">www.netflix.com</a>	나온
<a href="http://www.python.com">www.python.com</a>	단어들, 이나
<a href="http://www.daum.net">www.daum.net</a>	키워드, 적어
	놓기



# 검색 엔진 구동 방식

---

현재는 더 발전된 검색 알고리즘과 추천시스템을 보유 중



# 웹 크롤링 vs 웹 스크래핑

# 웹 크롤링

---

# Web Crawl

웹

기어 다니다

체계적으로 웹 사이트를 돌아다니는 것

검색 엔진들이 웹사이트를 인덱싱 하기 위해 사용됨

# 웹 스크래핑

---

# Web Scrape

웹

긋다

웹사이트의 데이터를 긋어오는 행위

상품정보 수집, 리뷰 수집, 블로그 내용 수집 등

# 웹의 발전

## CSS

---

# Cascading Style Sheets

폭포처럼 쏟아지다/풍성하게 늘어나다

하이퍼텍스트에 디자인 요소 추가 가능



# CSS

## FinInsight<sup>+</sup>

본인의 강의 목표를 달성해 보세요.

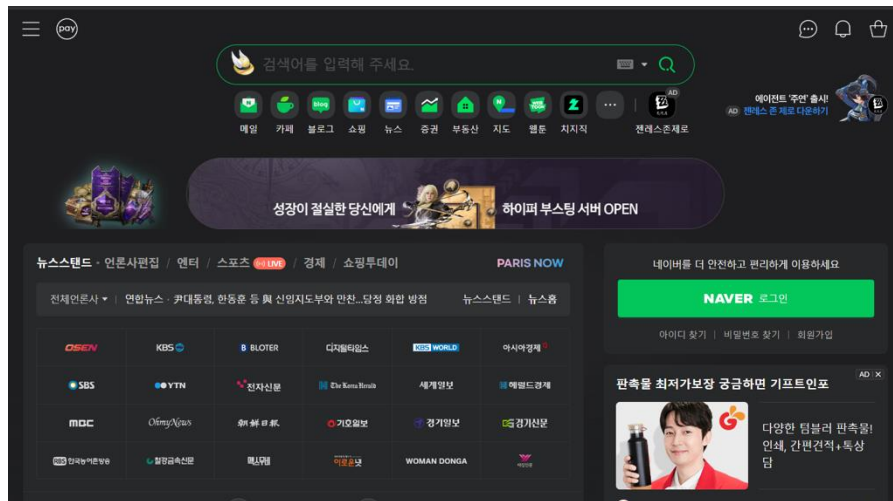
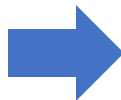
핀인사이트는 고객의 비즈니스 목표 달성을 위해 인공지능 기반 금융 데이터 분석 소프트웨어와 컨설팅 및 교육 서비스를 표를 빠르게 달성해보세요.

핀인사이트와 함께 궁금한 점을 해결해 봐요.

궁금한 게 생겼거나 함께 의견을 나누고 싶다면? 서로의 질문과 답변을 통해, 분석 역할을 더욱 향상시켜 보세요

새로운 데이터 분석 교육의 시작, 핀인사이트

- 파이썬 프로그래밍 기초 in Python
- 웹 크롤링
- 데이터 베이스
- 머신 러닝
- 딥러닝
- 생성형 AI



## Java Script

---

# Java Script

웹페이지에 다양한 기능을 추가할 수 있는 프로그래밍 언어

프로그래밍 언어를 통해 다양한 기능을 개발하여 웹페이지에 추가 가능

# Java Script

---

# Java Script

## Web Browser

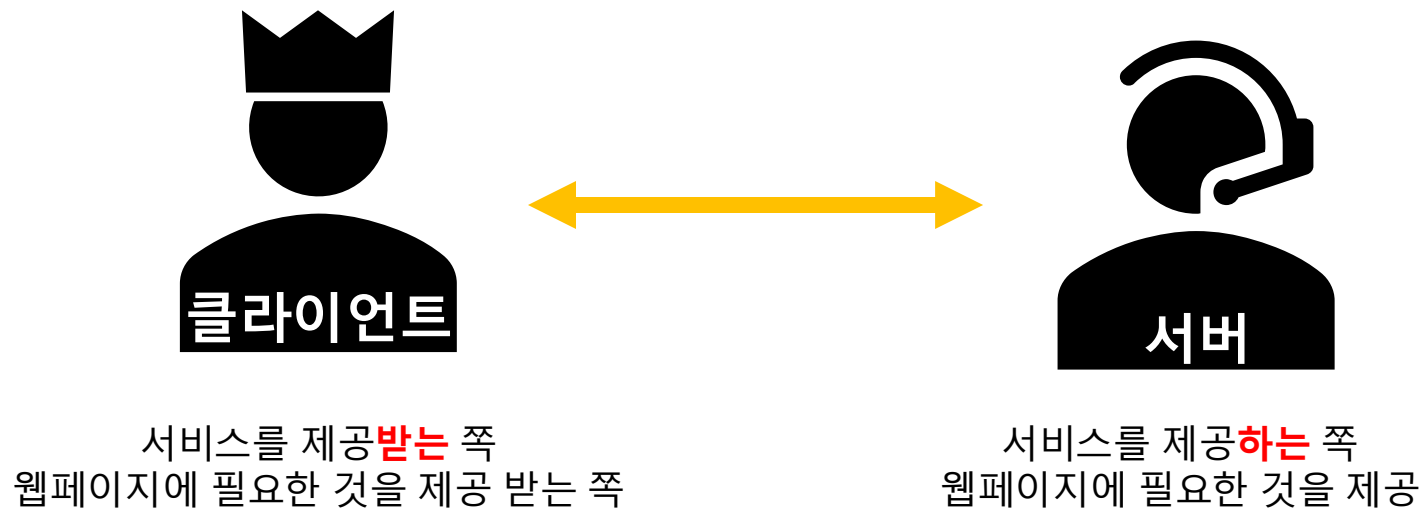
---

# Web Browser

단순 브라우저에서 사용자 편의 기능이 추가 됨  
뒤로 가기, 북마크, 등

# 클라이언트 & 서버

# 클라이언트 & 서버



# 클라이언트 & 서버

## 네이버 접속 시



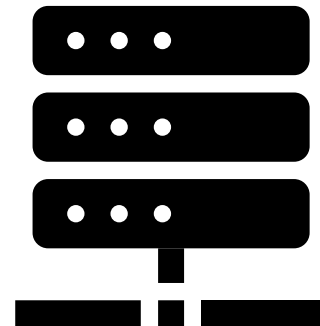
### 클라이언트

웹페이지를 제공받으려는 고객  
주소창에 [www.naver.com](http://www.naver.com) 입력

네이버 페이지 요청 (**request**)



네이버 페이지 필요 요소 제공  
(**response**)



### 서버

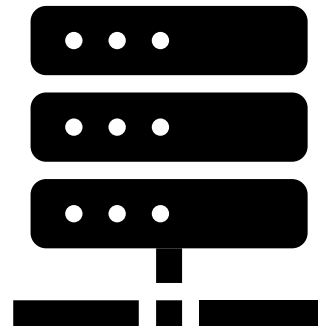
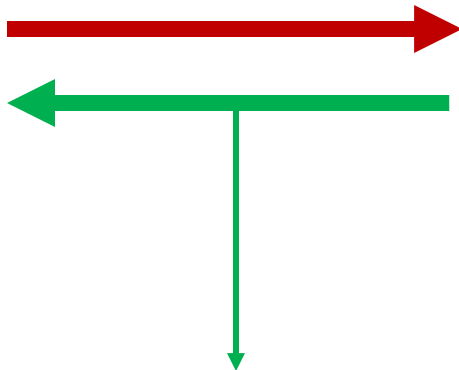
이 세상 어딘가에 있는 컴퓨터  
Request를 통해 뭐가 필요한지 파악 후  
필요한 것들을 **client**한테 제공

# 클라이언트 & 서버

네이버 접속 시



클라이언트



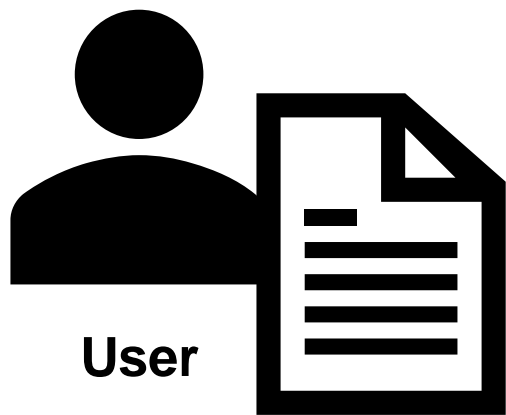
서버

**Response** 하는 것들 : html 코드, css, java script 코드 등



# 클라이언트 & 서버

네이버 접속 시



User

Naver페이지



클라이언트

우리는 브라우저를 통해 response받은 네이버 페이지를 볼 수 있음

# Anaconda 설치

# IDE 선택

---



VS Code



PyCharm

원하는 IDE 선택

# 아나콘다 가상환경 만들기

가상환경생성

```
conda create -n study python=3.10
```

가상환경이름

파이썬버전

```
(base) C:\Users\h[redacted]>conda create -n test python=3.10
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: C:\Users\h[redacted]\.conda\envs\test

added / updated specs:
- python=3.10

The following packages will be downloaded:
```

package	build	
bzip2-1.0.8	h2bbff1b_6	90 KB
ca-certificates-2024.3.11	haa95532_0	128 KB
libffi-3.4.4	hd77b12b_1	122 KB
openssl-3.0.13	h2bbff1b_1	7.5 MB
pip-24.0	py310haa95532_0	2.9 MB
python-3.10.14	he1021f5_1	15.9 MB
setuptools-69.5.1	py310haa95532_0	1013 KB
sqlite-3.45.3	h2bbff1b_0	973 KB
tk-8.6.14	h0416ee5_0	3.5 MB
tzdata-2024a	h04d1e81_0	116 KB

가상환경 리스트

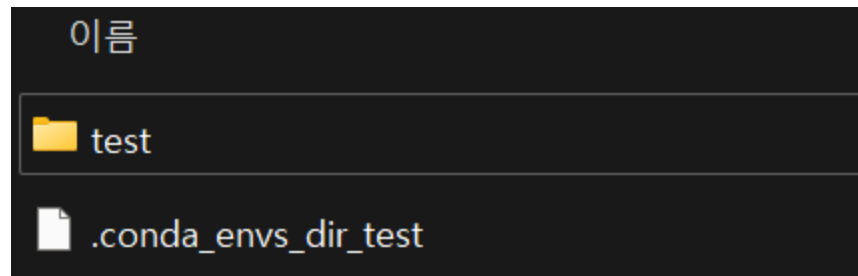
```
conda info --envs
```

```
(base) C:\Users\h[redacted]>conda info --envs
# conda environments:
#
base                * C:\ProgramData\anaconda3
test                C:\Users\h[redacted]\.conda\envs\test
```

# 아나콘다 가상환경 만들기

---

생성된 가상환경 경로로 가서 직접 확인



C:/Users/username/.conda/envs

# 아나콘다 가상환경 만들기

---

가상환경 활성화

conda activate study

```
(base) C:\Users\jinbeom>activate study  
(study) C:\Users\jinbeom>_
```

가상환경 비활성화

conda deactivate

```
(study) C:\Users\jinbeom>deactivate study  
deactivate does not accept arguments  
remainder_args: ('study',)
```

# 아나콘다 가상환경 만들기

---

가상환경 삭제

`conda env remove -name {가상환경 이름}`

```
(base) C:\>conda env remove --name webTest
```

가상환경 경로로 가서 삭제 됐는지도 확인 해보세요!

# Requests & Response

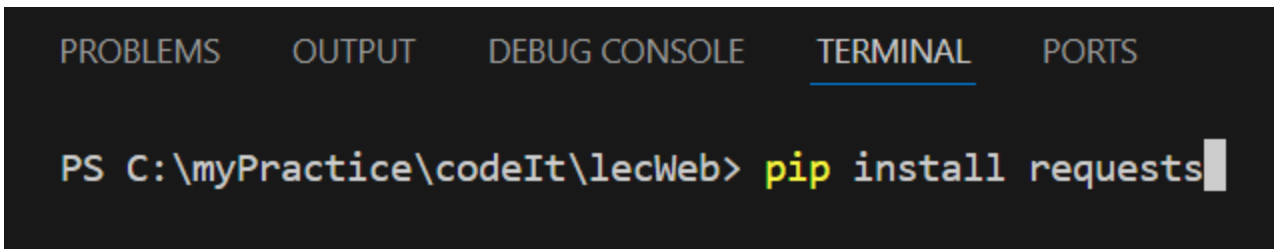


# requests

## ✓ requests 설치

1. ctrl + ` : 터미널 열기
2. pip install requests (가상환경 잘 확인하기)
3. proceed (y/n) 나오면 y입력 후 enter

Proceed (y/n)?



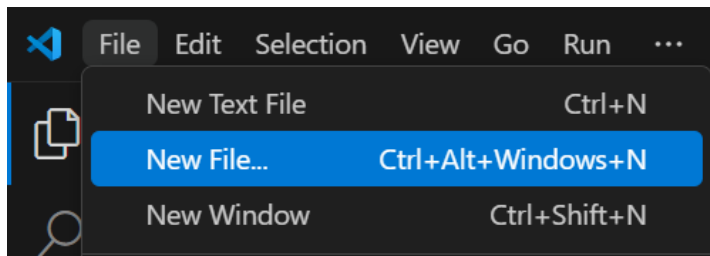
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\myPractice\codeIt\lecWeb> pip install requests
```

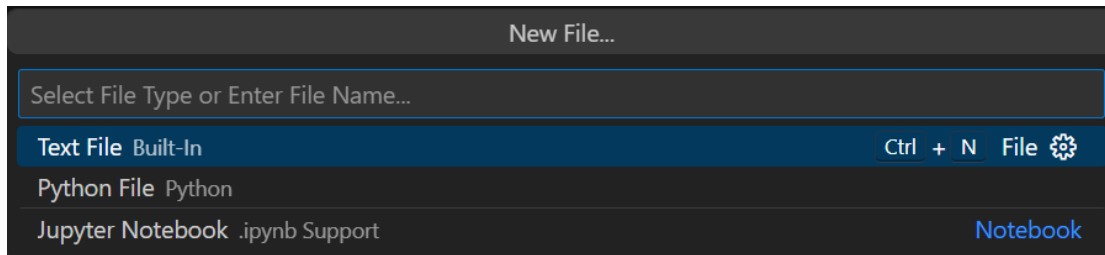
라이브러리 삭제 시 : pip uninstall requests 실행

# 파이썬 파일 생성

## 1. File -> New File



## 2. Python File 선택



## 3. Ctrl + s : 저장위치 선정 후 web1\_3.py로 저장

# requests

---

requests로 네이버 페이지 서버에 요청해서 response 받기

```
import requests

response = requests.get('https://www.naver.com')
print(response)
```

실행 결과 :

```
<Response [200]>
```

200 : 상태코드

# requests

---

requests로 네이버 페이지 서버에 요청해서 response 받기

```
import requests

response = requests.get('https://www.naver.com')
print(response.text)
```

response.text : response 내용을 text로 반환

# 실습

---

1. 네이버/구글/다음 뉴스 페이지 response 받아오기
2. 네이버/구글/다음 증권 response 받아오기

# 상태코드

# 상태코드란?

HTTP : Hypertext Transfer Protocol

(서버와 클라이언트 사이에 이뤄지는 요청/응답 규약)

클라이언트가 보낸 **HTTP** 요청에 대한 **서버의 응답 코드**  
상태 코드에 따라 **성공 / 실패 여부 판단**

# 상태코드

---

## ✓ 상태 코드 분류

1xx (information) : 조건부 응답

2xx (Successful) : 성공

3xx (Redirection) : 리다이렉션 완료

4xx ( Client Error) : 요청 오류

5xx (Server Error) : 서버 오류



# 상태코드

## ✓ 1xx (Information) : 조건부 응답

Request received, continuing process

요청이 수신되어 처리 중을 의미 (거의 볼 일 없는 코드)

상태코드	요약	설명
100	continue	<ul style="list-style-type: none"><li>• 서버로 보낸 요청에 문제가 없으니 다음 요청을 이어서 보내도 됨을 의미.</li><li>• 만약 클라이언트의 작업이 완료되었다면 이 응답은 무시해도 됨.</li></ul>

# 상태코드

## ✓ 2xx (Successful) : 성공

The action was successfully received, understood, and accepted.

요청을 정상적으로 처리했음을 의미

상태코드	요약	설명
200	ok	<ul style="list-style-type: none"> <li>요청이 성공적으로 수행 됐음을 의미.</li> <li>주로 Get 요청에 대한 응답.</li> </ul>
201	created	<ul style="list-style-type: none"> <li>요청이 성공적으로 수행 됐으며, 그 결과로 새로운 리소스가 생성됐음을 의미.</li> <li>주로 Post 요청에 대한 응답</li> </ul>
202	Accepted	<ul style="list-style-type: none"> <li>요청은 접수됐지만, 처리는 완료되지 않음.</li> <li>배치처리와 같이 요청 접수 후 일정 시간이 지난 후 요청을 처리하는 경우의 응답</li> </ul>

# 상태코드

---

상태코드	요약	설명
203	Non-Authoritative Information	<ul style="list-style-type: none"> <li>요청이 성공적으로 수행 됐으나, 요청에 대한 검증이 되지 않음</li> </ul>
204	No Content	<ul style="list-style-type: none"> <li>요청이 성공적으로 수행 되었고, 응답 payload에 보낼 데이터가 없음.</li> <li>주로 DELETE 요청에 대한 응답으로 사용 됨</li> </ul>
205	Rest Content	<ul style="list-style-type: none"> <li>서버가 요청을 성공적으로 처리했지만, 콘텐츠를 표시하지 않음</li> <li>클라이언트가 콘텐츠를 재설정할 것을 요구</li> </ul>
206	Partial Content	<ul style="list-style-type: none"> <li>서버가 Get 요청의 일부만 성공적으로 처리했음을 의미</li> </ul>

# 상태코드

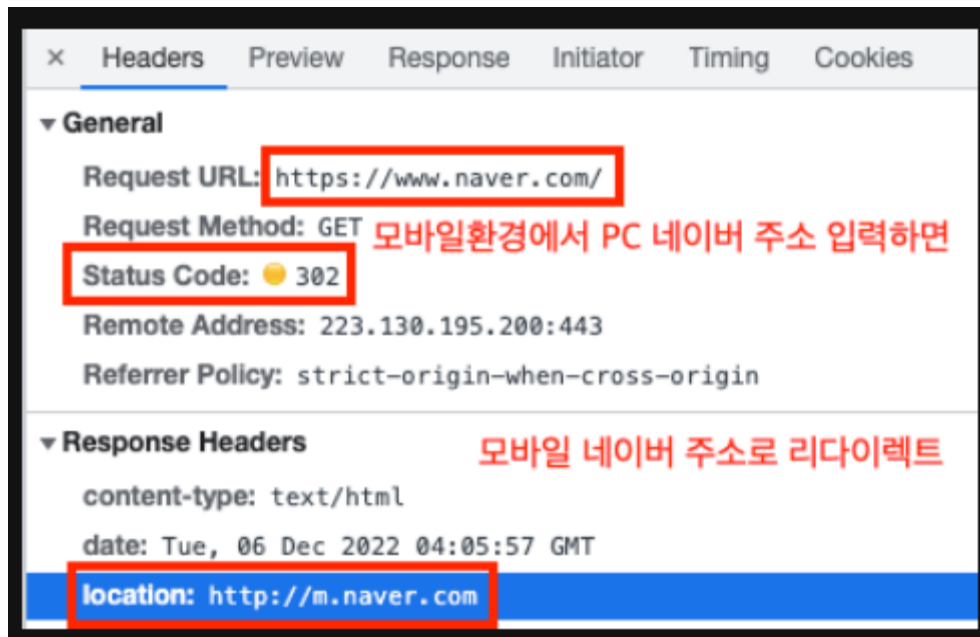
---

## ✓ 3xx (Redirection) : 리다이렉션 완료

- Further action must be taken in order to complete the request
- 요청 완료를 위해 추가 작업 조치가 필요함을 의미. (주로 리다이렉트를 할 때 많이 사용 됨)
- 해당 응답을 받으면, 브라우저는 HTTP 헤더에 들어있는 Location 필드를 찾아,  
해당 필드가 존재할 경우, Location에 담긴 URL로 자동으로 리다이렉트 함
- 리다이렉트
  - 서버가 브라우저(클라이언트)에게, 요청한 리소스가 다른 위치로 이동 됐음을 알려주는 방법
  - 클라이언트가 요청한 URL에 대해 서버가 응답을 통해 다른 URL로 이동하라고 지시

# 상태코드

- ✓ 3xx (Redirection) : 리다이렉션 완료



# 상태코드

## ✓ 3xx (Redirection) : 리다이렉션 완료

상태코드	요약	설명
300	Multiple Choices	<ul style="list-style-type: none"> <li>요청에 대해 하나 이상의 리소스가 존재함을 의미</li> </ul>
301	Moved Permetly	<ul style="list-style-type: none"> <li>요청한 리소스의 URI가 영구적으로 변경됨 (URL변경됨)</li> </ul>
303	Found	<ul style="list-style-type: none"> <li>요청한 리소스의 URI가 일시적으로 변경됨</li> </ul>
304	No Modified	<ul style="list-style-type: none"> <li>리소스가 수정되지 않음을 의미</li> <li>해당 응답을 받으면, 클라이언트는 서버로 부터 리소스를 재전송 받지 않고 캐싱된 리소스를 사용</li> </ul>

# 상태코드

## ✓ 3xx (Redirection) : 리다이렉션 완료

상태코드	요약	설명
307	Temporary Redirect	<ul style="list-style-type: none"><li>• 302와 유사</li><li>• 클라이언트는 HTTP 메서드를 유지한채, 요청을 재송신할 필요가 있음을 의미</li></ul>
308	Permanent Redirect	<ul style="list-style-type: none"><li>• 301과 유사</li><li>• 클라이언트는 HTTP 메서드를 유지한채, 요청을 재송신할 필요가 있음을 의미</li></ul>

# 상태코드

## ✓ 4xx (Client Error) : 요청 오류

The request contains bad syntax or cannot be fulfilled

클라이언트 오류(잘못된 문법 등)으로 인해 서버가 요청을 처리할 수 없음을 의미

상태코드	요약	설명
400	Bad Request	<ul style="list-style-type: none"> <li>잘못된 문법 등으로 인해 클라이언트가 올바르게 보내 서버가 요청을 이해할 수 없음을 의미</li> </ul>
401	Unauthorized	<ul style="list-style-type: none"> <li>인증되지 않은 사용자가 인증이 필요한 리소스를 요청하는 경우</li> <li>로그인이 필요한 API를 비로그인 사용자가 호출했을 때 사용 됨</li> </ul>
403	Forbidden	<ul style="list-style-type: none"> <li>클라이언트가 콘텐츠에 접근할 권한이 없음</li> <li>서버가 클라이언트가 누구인지 알고 있음</li> <li>특정 IP나 국가가 차단 되어있는 사이트에 접속을 시도한 경우</li> </ul>



# 상태코드

상태코드	요약	설명
404	Not Found	<ul style="list-style-type: none"> <li>요청한 리소스가 존재하지 않음을 의미</li> <li>인증되지 않은 클라이언트로부터 리소스를 숨기기 위해 403 대신 쓰이기도 함</li> </ul>
405	Method not allowed	<ul style="list-style-type: none"> <li>현재 리소스에 맞지 않는 메서드를 사용했음을 의미</li> <li>Get요청만 허용되는데 Post요청을 한 경우</li> </ul>
406	Not Acceptable	<ul style="list-style-type: none"> <li>알맞은 콘텐츠 타입이 없음을 의미</li> <li>서버의 리소스가 클라이언트의 HTTP 헤더에 들어있는 Accept 필드에 명시된 콘텐츠 타입이 아닌 경우의 응답</li> </ul>
408	Request Timeout	<ul style="list-style-type: none"> <li>응답하는 시간이 너무 오래 걸림</li> </ul>

# 상태코드

상태코드	요약	설명
409	conflict	<ul style="list-style-type: none"><li>요청이 현재 서버의 상태와 충돌될 때의 응답</li></ul>
412	Precondition Failed	<ul style="list-style-type: none"><li>서버가 요청자가 요청 시 부과한 사전조건을 만족하지 않을 때의 응답</li></ul>
413	Payload Too Large	<ul style="list-style-type: none"><li>요청이 너무 커서 서버가 처리할 수 없음을 의미</li></ul>
429	Too many Requests	<ul style="list-style-type: none"><li>클라이언트가 지정된 시간 내에 너무 많은 요청을 보낸 경우</li></ul>

# 상태코드

## ✓ 5xx (Server Error) : 서버 오류

The server failed to fulfill an apparently valid request

서버 오류로 인해 서버가 정상 요청을 처리하지 못함을 의미

상태코드	요약	설명
500	Internal Server Error	<ul style="list-style-type: none"> <li>• 서버에 오류가 발생하여 응답할 수 없음을 의미</li> <li>• 서버에 오류가 발생했으나 처리방법을 알 수 없을 경우</li> </ul>
501	Not Implemented	<ul style="list-style-type: none"> <li>• 클라이언트 요청에 대한 서버의 응답 수행 기능이 없음을 의미</li> </ul>
502	Bad Gateway	<ul style="list-style-type: none"> <li>• 서버가 게이트웨이로부터 잘못된 응답을 수신했음을 의미</li> <li>• 서버의 부모 서버에서 오류가 발생한 경우</li> <li>• 서버에 접속하는 사용자가 많아 과부하 될 때 발생</li> </ul>

# 상태코드

상태코드	요약	설명
503	Service Unavailable	<ul style="list-style-type: none"><li>• 서버가 요청을 처리할 준비가 되지 않음을 의미</li><li>• 일반적으로 유지보수를 위해 작동이 중단되거나 과부하가 걸린 경우의 응답</li></ul>
504	Gateway Timeout	<ul style="list-style-type: none"><li>• 서버가 게이트웨이의 역할을 하고있으며, 한 서버가 액세스 하고 있는 다른 서버에서 적시에 응답을 받지 못했음을 의미</li></ul>

# Post & Get

# Post요청 & Get 요청

---

웹에서 클라이언트(웹 브라우저)와 서버 간의 데이터를 주고받기 위한  
두 가지 주요 HTTP 메서드

# Get 요청

---

## ✓ 용도

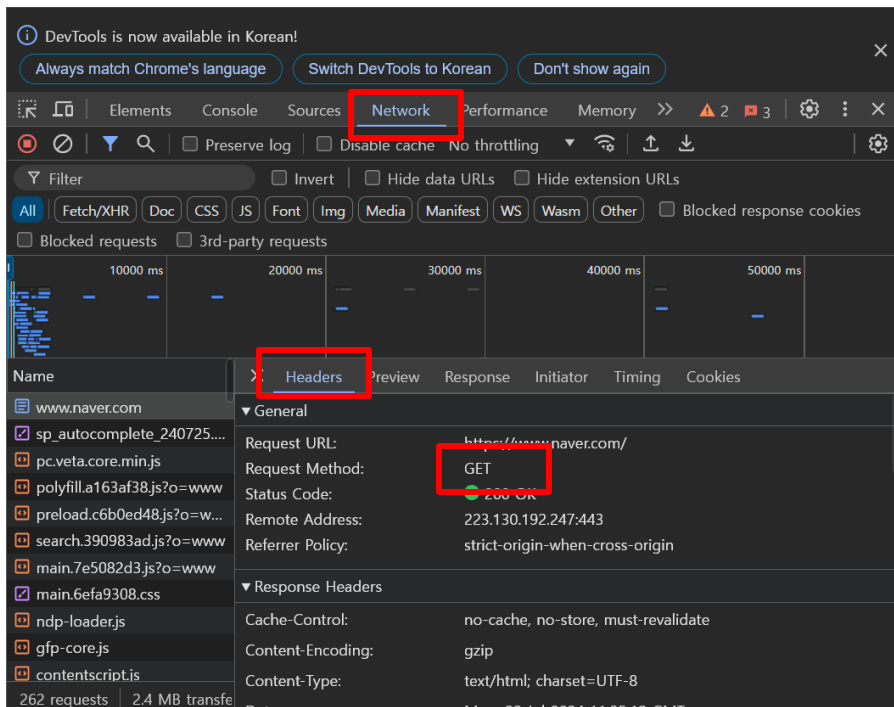
- 서버에게 **데이터를 요청**할 때 사용 됨
- (웹 페이지를 열거나 **데이터를 조회**할 때 사용 됨)

## ✓ 특징

- 데이터 전송 방식 : 데이터는 **URL의 쿼리 스트링**으로 전송 됨
  - \* ex) https://example.com/search?**query=example**
- 브라우저 캐시 / 서버 캐시에 의해 저장될 수 있음
- URL의 길이 제한으로 전송할 수 있는 데이터의 양이 제한 됨
- 데이터 조회 등 서버의 상태를 변경하지 않는 요청에 사용 됨
- 동일한 Get요청을 여러 번 반복해도 서버의 상태가 변하지 않음

# Get 요청

## 개발자 도구의 네트워크 확인





# Post 요청

---

## ✓ 용도

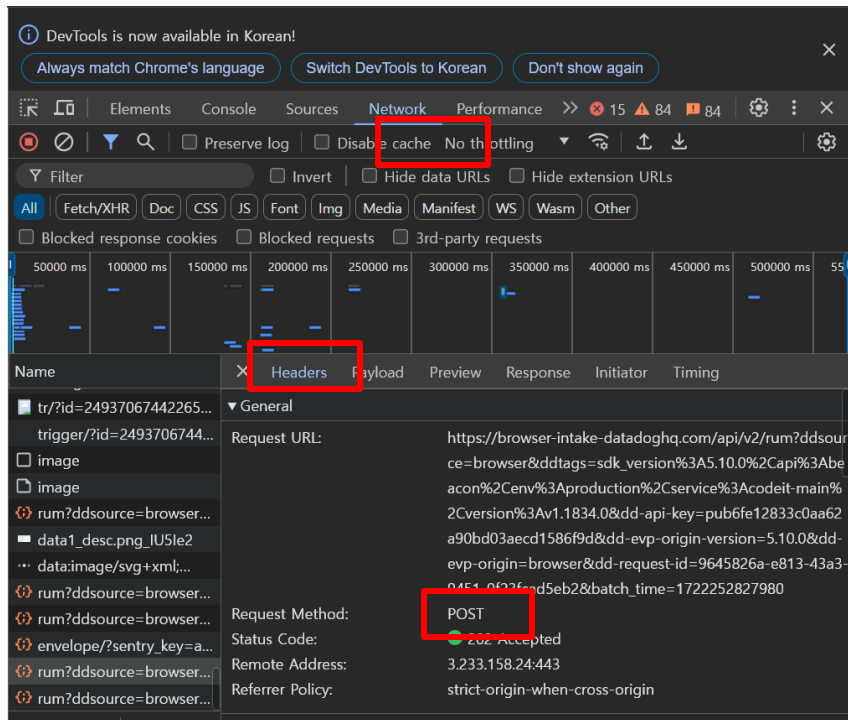
- 서버에 데이터를 전송하거나 서버의 상태를 변경할 때 사용 됨
- 게시물 작성, 폼 제출, 파일 업로드, DB에 데이터 삽입 등

## ✓ 특징

- 데이터는 HTTP의 본문(body)으로 전송 됨 -> 대용량 데이터 전송 가능
- 캐싱 불가
- 서버의 상태를 변경하는 작업에 사용되므로, 안전하지 않음
- 동일한 Post요청을 여러 번 반복하면 서버 상태가 변경될 수 있음  
(같은 데이터를 여러 번 삽입)

# Post 요청

## 개발자 도구의 네트워크 확인



# Post 요청

## 개발자 도구의 네트워크 확인

▼ General	
Request URL:	https://jsonplaceholder.typicode.com/cdn-cgi/rum?
Request Method:	POST
Status Code:	🟢 200
Remote Address:	104.21.4.48:443
Referrer Policy:	strict-origin-when-cross-origin
▶ Response Headers (12)	
▶ Request Headers (19)	
▼ Request Payload	<a href="#">view source</a>
<pre>{memory: {totalJSHeapSize: 5317271, usedJSHeapSize: 4179459, jsHeapSizeLimit: 4294705152},...}   documentWriteIntervention: false   errorCount: 0   eventType: 1   firstContentfulPaint: 388.7000000476837   firstPaint: 388.7000000476837   location: "https://jsonplaceholder.typicode.com/guide/"   ▶ memory: {totalJSHeapSize: 5317271, usedJSHeapSize: 4179459, jsHeapSizeLimit: 4294705152}   payloadId: "29984e57-0ec4-4dff-abff-a6f81bc9dd5a"   referrer: "https://jsonplaceholder.typicode.com/"   resources: []   si: 100   siteToken: "2f059cbf58a24e59854ee0884c97b166"   st: 2   startTime: 1636295072423   ▶ timingsV2: {unloadEventStart: 359.60000002384186, unloadEventEnd: 359.60000002384186,...}   ▶ versions: {f1: "2021.10.0", js: "2021.9.0", timings: 2}</pre>	

# 웹사이트 주소 이해하기

# http & https

---

네이버 쇼핑 : https://shopping.naver.com/market/emart/home

클라이언트가 웹 서버와 어떤 방식으로 소통하는지 나타냄

가장 기본적으로 **http** 사용

추가적인 보안이 사용될 시 **https** 사용

# 도메인 주소

---

네이버 쇼핑 : <https://shopping.naver.com/market/emart/home>

도메인 주소(서브도메인)

서브 도메인 사용 시 한 도메인에서 여러 개의 서비스를 구분할 수 있음  
서브 도메인은 별도의 웹 서버 / 어플리케이션을 나타낼 수도 있음

# 경로

---

네이버 쇼핑 : <https://shopping.naver.com/market/emart/home>

경로

슬래시(/)로 구분

웹 사이트에서 어떤 페이지를 보여줄지 나타냄

# 쿼리 스트링

---

네이버 쇼핑 : <https://search.shopping.naver.com/search/all?query=tv>

물음표 뒤 : 쿼리 스트링

함수의 파라미터 처럼 페이지의 옵션을 넘겨주는 역할  
검색 내용이 바뀌면 URL의 쿼리 스트링의 내용도 바뀜



## & 기호

---

**월마트 : <https://www.walmart.com/search?q=tv&page=2&affinityOverride=default>**

**& 기호 : 여러 파라미터를 한 번에 넘겨줄 수 있음**

검색어 쿼리(q=tv)와 페이지(page=2) 파라미터를 동시에 넘겨주게 됨  
& 기호로 구분되는 옵션들은 순서를 바꿔도 상관 없음

## 섹션

---

위키피디아 : <https://en.wikipedia.org/wiki/HTML#History>

# : 페이지 내의 특정 부분으로 이동

실제 페이지 내용이 바뀌지 않음

로딩 시 맨 위로 가지 않고 해당 섹션으로 감

# 실습

# 여러 페이지 갖고 오기

---

For문을 사용하여  
**URL의 page 옵션**을 사용하여 여러 개의 페이지 html을 갖고 오는  
실습을 해보자

조마다 컬리, 옥션, 네이버 쇼핑의 여러 페이지 html 갖고 오기

# HTTP 구조

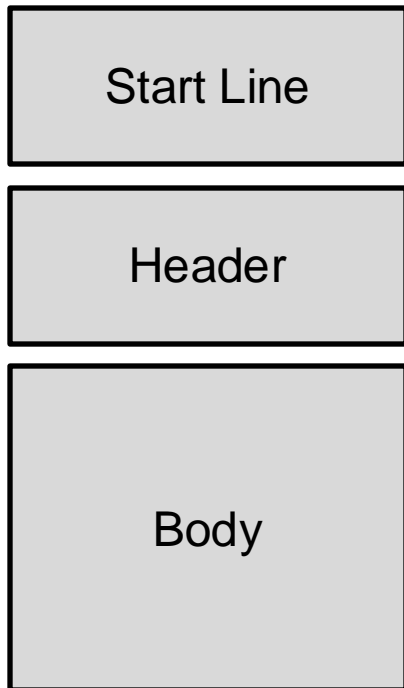
# HTTP 통신 방식

---

하이퍼텍스트 문서를 교환하기위해 만들어진 통신규약  
TCP/IP 프로토콜 기반

요청(Request) 응답(response) 구조  
HTTP는 어떠한 상태를 저장하지 않는다.

# HTTP 구조 - request



## Start Line

Request URL – 요청 URL

Request Method – 요청 방식 (Post, Get, Put, Delete)  
URL

## Header

accept : application / json

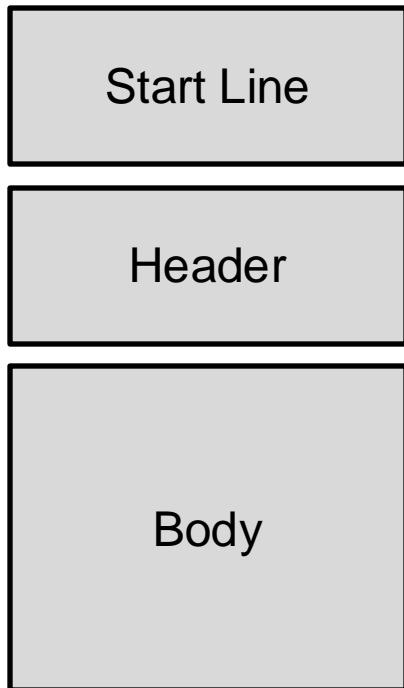
user-agent : Mozilla / 5.0

content-type : application / x-www-form-urlencoded; charset=UTF-8

## Body

전송하고 싶은 데이터

# HTTP 구조 - request



## Header

accept : application / json

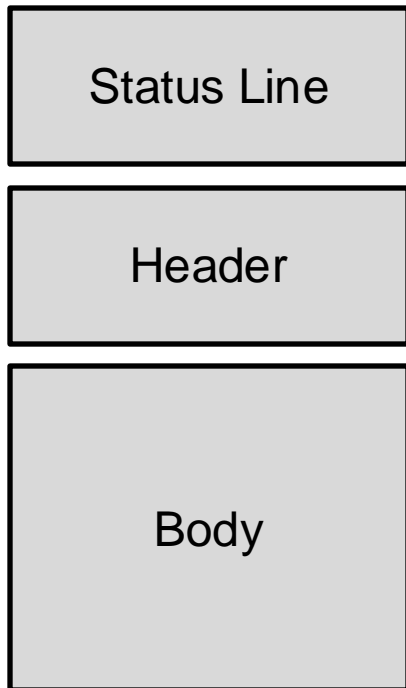
user-agent : Mozilla / 5.0

content-type : application / x-www-form-urlencoded; charset=UTF-8

- **요청 정보 전달** : 클라이언트 애플리케이션, 브라우저, 라이브러리에 대한 정보 전달 및 수신할 수 있는 타입 지정
- **인증 및 보안** : 서버에 접근할 때 필요한 인증정보 제공
- **컨텐츠 유형 지정** : 요청 본문에 포함된 데이터의 타입 지정
- **쿠키 및 세션 관리** : 서버가 클라이언트에게 설정한 쿠키 전달



# HTTP 구조 - response



## Status Line

### Status Code

- 200 : 오류 없이 성공
- 301 : URI가 중간에 다른 주소로 변경됨
- 400 : 요청이 잘못된 요청일 경우
- 401, 403 : 권한 없음
- 500 : 서버 오류

## Header

accept : application / json

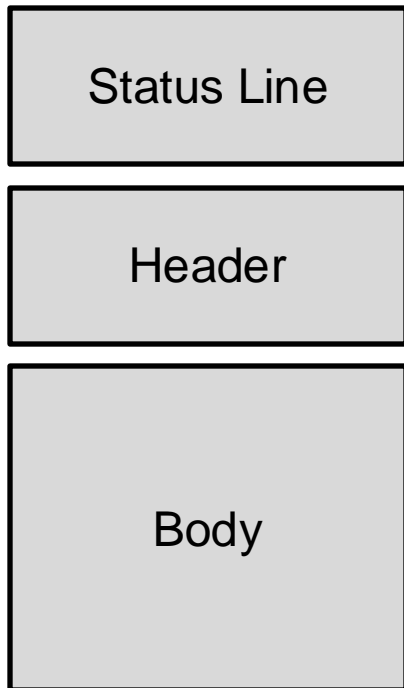
user-agent : Mozilla / 5.0

content-type : application / x-www-form-urlencoded; charset=UTF-8

## Body

전송하고 싶은 데이터

# HTTP 구조 - request



## Header

accept : application / json

user-agent : Mozilla / 5.0

content-type : application / x-www-form-urlencoded; charset=UTF-8

- **요청 정보 전달** : 클라이언트 애플리케이션, 브라우저, 라이브러리에 대한 정보 전달 및 수신할 수 있는 타입 지정
- **캐시 정보** : 클라이언트나 중간 프록시가 응답을 캐싱 할 방법 지정
- **쿠키 설정** : 클라이언트에 설정할 쿠키 정보 전달
- **다른 도메인의 웹 페이지에서 리소스에 접근할 수 있는 권한 지정**