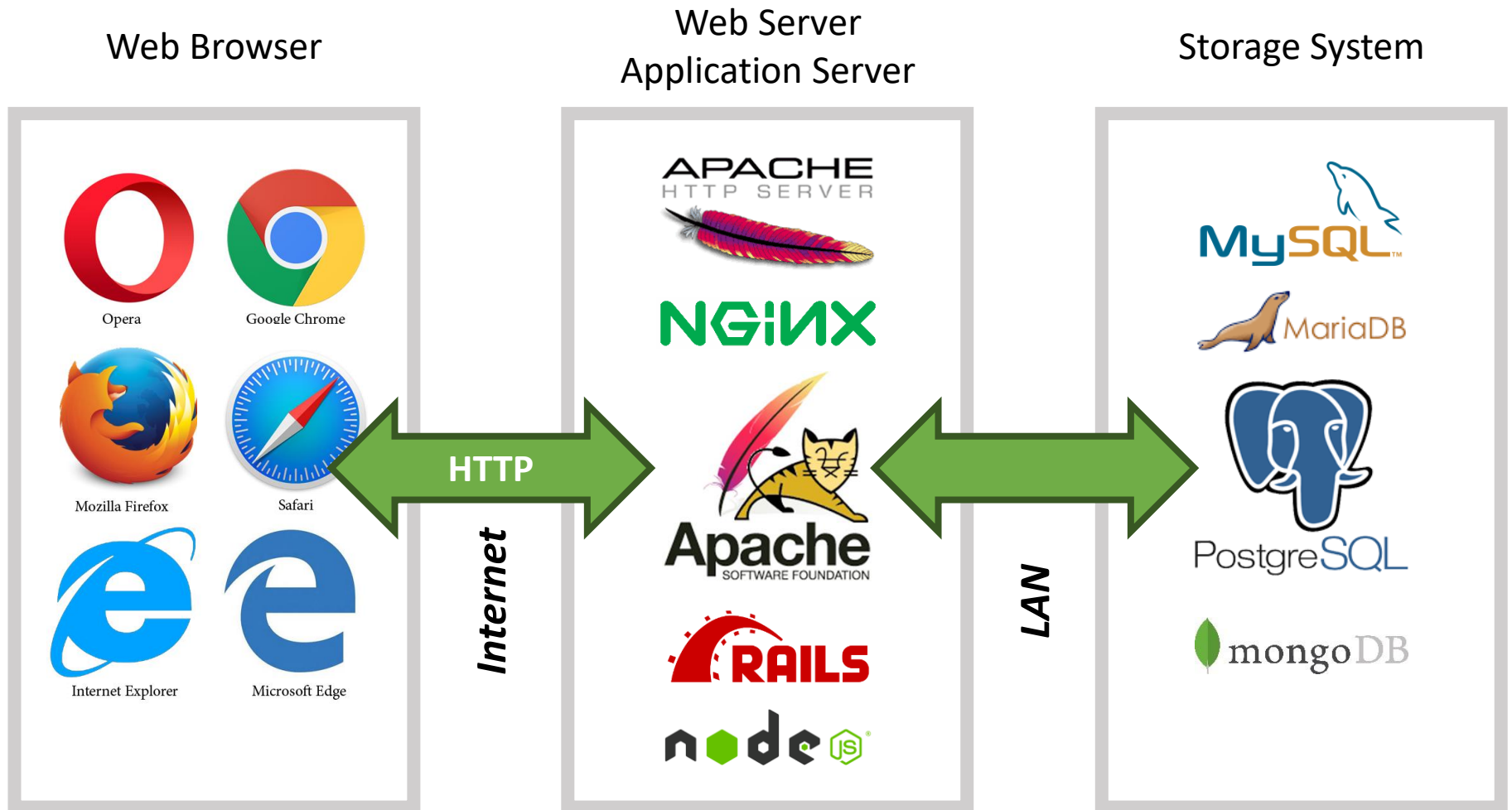


HTTP

# Web Application Architecture



*http://www.example.com:80/index.html*

<b>URL</b> Uniform Resource Locator	{	<b>Protocol</b>	<b>http</b>
		<b>Server</b>	www.example.com
		<b>Port</b>	80
		<b>File</b>	index.html

**HTTP**    **H**yper**T**ext **T**ransfer **P**rotocol

Simple ***request-response*** protocol layered on TCP/IP

**HTML**    **H**yper**T**ext **M**arkup **L**anguage

# request

**Header** {  
    **Get**        /index.html        HTTP/1.1  
    **Host**       www.example.com  
    **Accept**    text/html, \*/\*  
    **Accept-Language** en-us  
    **Accept-Charset**    ISO-8859-1,utf-8  
    **User-agent**        Mozilla/5.0  
    **Connection**       keep-alive

**Body**  
(Optional) {

# Method


<b>SAFE METHODS</b> NO ACTION ON SERVER	{	<b>GET</b>	HTTP/1.1 MUST IMPLEMENT THIS METHOD
		<b>HEAD</b>	<b>INSPECT</b> RESOURCE HEADERS
<b>MESSAGE WITH BODY</b> SEND DATA TO SERVER	{	<b>PUT</b>	<b>DEPOSIT</b> DATA ON SERVER — INVERSE OF GET
		<b>POST</b>	<b>SEND</b> INPUT DATA FOR PROCESSING
		<b>PATCH</b>	<b>PARTIALLY MODIFY</b> A RESOURCE
		<b>TRACE</b>	<b>ECHO</b> BACK RECEIVED MESSAGE
		<b>OPTIONS</b>	SERVER <b>CAPABILITIES</b>
		<b>DELETE</b>	DELETE A RESOURCE — NOT GUARANTEED

# REST

**REST**      Representational State Transfer

**CRUD**      Create, Read, Update, Delete

**RESTful**

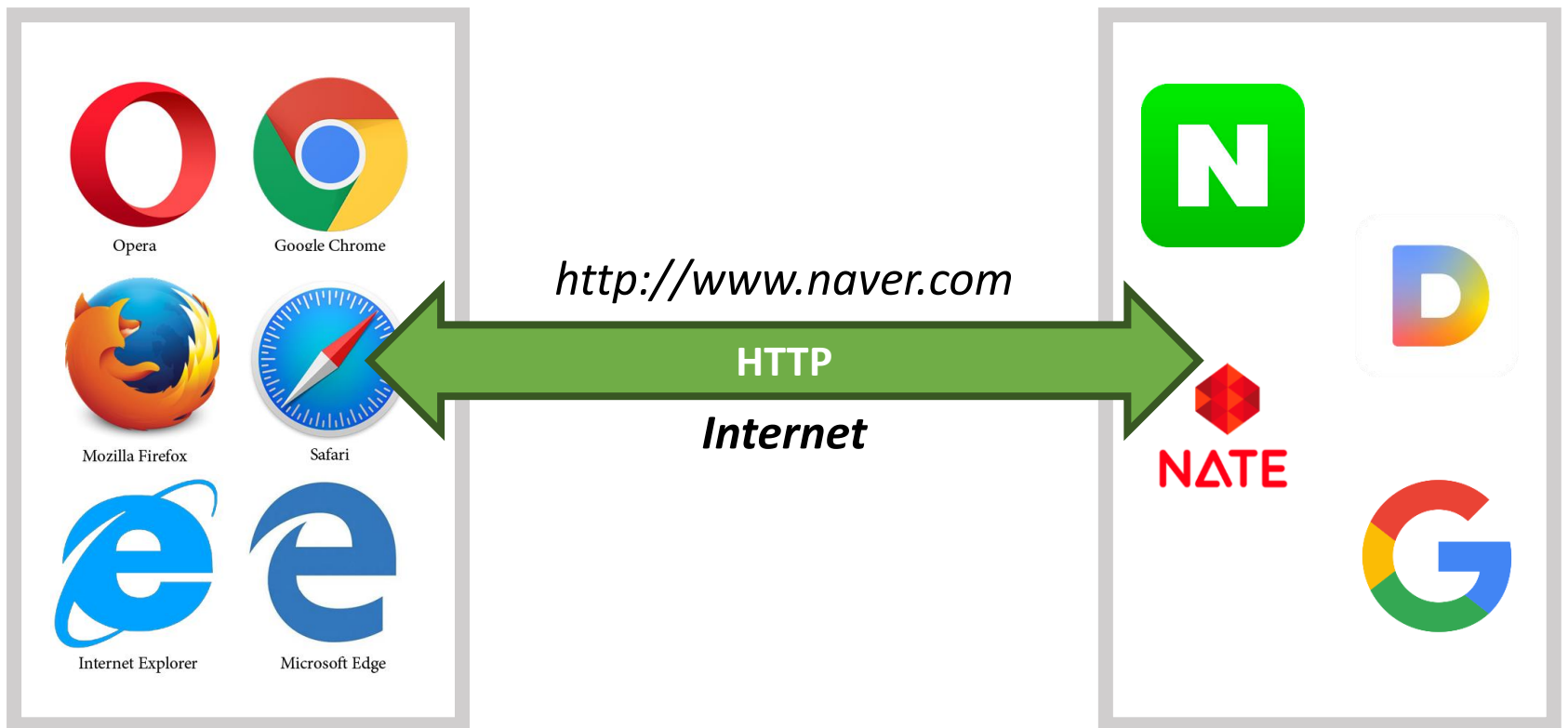


GET	/movies	Get list of movies
GET	/movies/:id	Find a movie by its ID
POST	/movies	Create a new movie
PUT	/movies	Update an existing movie
DELETE	/movies	Delete an existing movie

# HTTP Request-Response

Web Browser

Web Server



# *http://www.naver.com*



## ▼ Request Headers

**:authority:** www.naver.com

**:method:** GET

**:path:** /

**:scheme:** https

**accept:** text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8

**accept-encoding:** gzip, deflate, br

**accept-language:** ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7

**cache-control:** no-cache

**cookie:** npic=79bi60ql0lJkDEszX/3bql0z0I/RKFahZzkp84I+Dv45TGv/A1T4U11FknSwD764CA==; NNB=AFU2GPITRSavs; ASID=d27981e80000015e9927017e00045; PM\_CK\_loc=0f9e4037c8e30cb292966bc4bc7779e2f46f16fdb2644d91e01d56d4062e4e02; NID\_AUT=t10tLmaA90jrpJD13h/VhRfWVjB0kH5IDtkTJf05mGP6CJWY2202wodWu6R1+V0Kht8xCegKc0o6UnjvDMFjxy80zfz9zkCTeV5ojAKL2fiJbSLiv9digRUB8mk/5k7PpUErfZEBcPxw0TGk5XXw/n9UffDtf/McaCCFLjLTQkdiRJ7UpNnjKT+asagdbnW/g4kWSMgQJoMcGzr2XA91/YTu0lIKTjGp2PAAMLWAqAdeQPAAd4lkkeiGqiJxdg0SZCx8Y0GwKy2/rSD6PEFLoHljA4q09A2qBiUd2X2+AGZDsuW3ZJzYryyux/CvxBWjo2mV8C6LQvgszh9uxM3LKBoSll0ASMkjaI5hgplSHGWZhXGrddk10U+R89+idrZJFvQ78g3c6v98+jp0MfizWHS0Fmq250znYr0rq8dGSvqwgZfCtXadLk=

**pragma:** no-cache

**referer:** https://www.naver.com/

**upgrade-insecure-requests:** 1

**user-agent:** Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.119 Safari/537.36

**Protocol** https

**Server** www.naver.com

**Port** 80



# *http://www.naver.com*



## ▼ Response Headers

**cache-control:** no-cache, no-store, must-revalidate  
**content-encoding:** gzip  
**content-length:** 28175  
**content-type:** text/html; charset=UTF-8  
**date:** Thu, 28 Feb 2019 11:00:41 GMT  
**p3p:** CP="CA0 DSP CURa ADMa TAIa PSAa OUR LAW STP PHY ONL UNI PUR FIN COM NAV INT DEM STA PRE"  
**pragma:** no-cache  
**referrer-policy:** unsafe-url  
**server:** NWS  
**status:** 200  
**strict-transport-security:** max-age=63072000; includeSubdomains  
**vary:** Accept-Encoding  
**x-edgeconnect-midmile-rtt:** 1  
**x-edgeconnect-origin-mex-latency:** 15  
**x-frame-options:** DENY  
**x-xss-protection:** 1; mode=block

# Browser Rendering

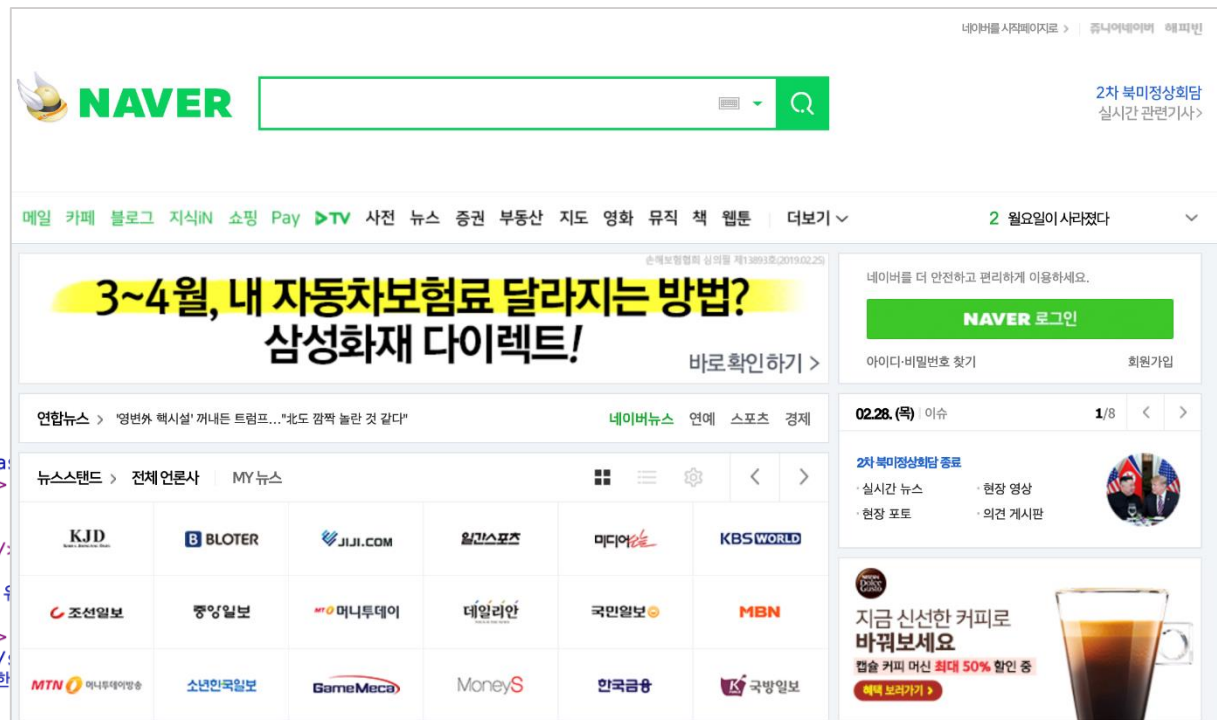


Google Chrome



```
<!doctype html>
```

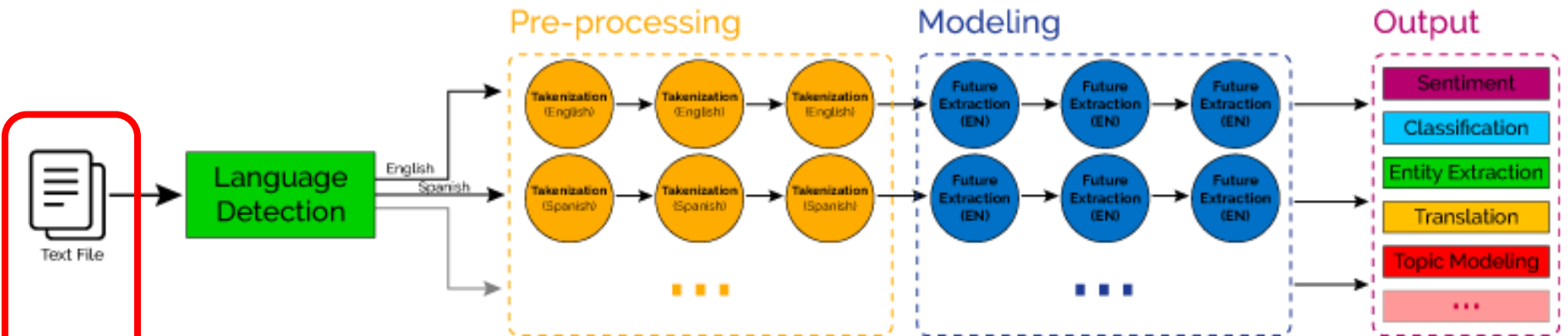
```
<html lang="ko">
<head>
<meta charset="utf-8">
<meta name="Referrer" content="origin">
<meta http-equiv="Content-Script-Type" content="text/javascript">
<meta http-equiv="Content-Style-Type" content="text/css">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=1100">
<meta name="apple-mobile-web-app-title" content="NAVER" />
<meta name="robots" content="index,nofollow"/>
<meta name="description" content="네이버 메인에서 다양한 정보와 유용한 콘텐츠를 만나 보세요">
<meta property="og:title" content="네이버">
<meta property="og:url" content="https://www.naver.com/">
<meta property="og:image" content="https://s.pstatic.net/static/www/mobile/edit/2016/0705/mobile_212852414260.png">
<meta property="og:description" content="네이버 메인에서 다양한 정보와 유용한 콘텐츠를 만나 보세요"/>
<meta name="twitter:card" content="summary">
<meta name="twitter:title" content="네이버">
<meta name="twitter:url" content="https://www.naver.com/">
<meta name="twitter:image" content="https://s.pstatic.net/static/www/mobile/edit/2016/0705/mobile_212852414260.png">
<meta name="twitter:description" content="네이버 메인에서 다양한 정보와 유용한 콘텐츠를 만나 보세요"/>
```



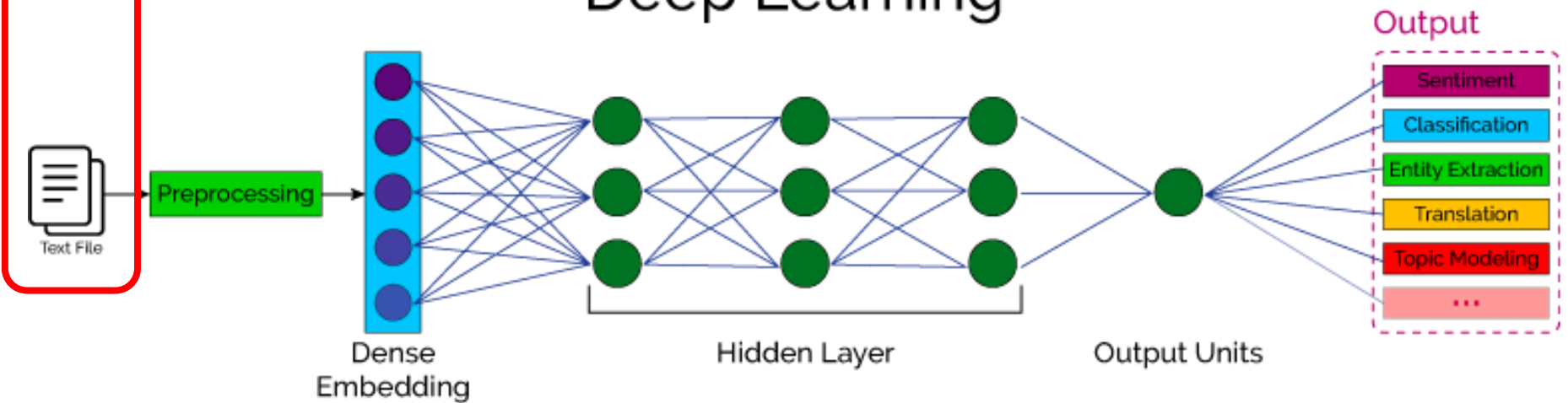
# How to prepare Dataset

Crawling / Scraping

# Classical NLP



# Deep Learning





*Crawler, Spider, Agent, Bot, ...*

# Legal Issues

# Opt-in vs Opt-out

**Opt-in** 정보수집을 명시적으로 동의할 때에만 정보수집 가능; *Whitelist*

**Opt-out** 정보수집을 명시적으로 거부할 때에만 정보수집 중단; *Blacklist*

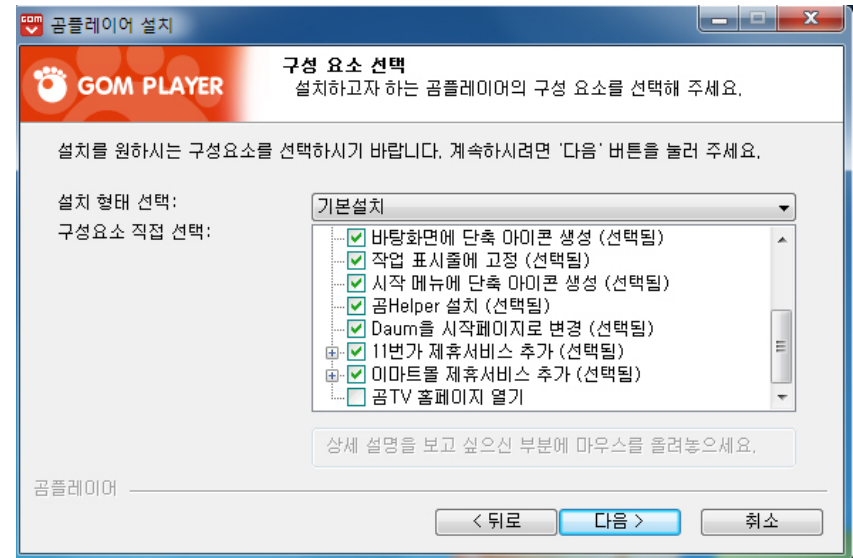


The screenshot shows Naver's consent form. At the top is the Naver logo. Below it, there are three sections, each with a checkbox and a checkmark icon:

- 이용약관, 개인정보 수집 및 이용, 위치정보 이용약관(선택)에 모두 동의합니다.**
- 네이버 이용약관 동의(필수)**
- 개인정보 수집 및 이용에 대한 안내(필수)**

Under the second section, there is a scrollable area with text explaining the terms of service. Under the third section, there is a scrollable area with text explaining the privacy policy.

**Opt-in 개인정보**



The screenshot shows GOM Player's installation window. At the top, it says "GOM PLAYER" and "구성 요소 선택" (Select components). Below that, it says "설치하고자 하는 공플레이어의 구성 요소를 선택해 주세요." (Please select the components of GOM Player you want to install.).

There are two sections for selection:

- 설치 형태 선택:** 기본설치 (Default installation)
- 구성요소 직접 선택:** A list of components with checkboxes:

- ☒ 바탕화면에 단축 아이콘 생성 (선택됨)
- ☒ 작업 표시줄에 고정 (선택됨)
- ☒ 시작 메뉴에 단축 아이콘 생성 (선택됨)
- ☒ GOMHelper 설치 (선택됨)
- ☒ Daum을 시작페이지로 변경 (선택됨)
- ☒ 11번가 제휴서비스 추가 (선택됨)
- ☒ 이마트를 제휴서비스 추가 (선택됨)
- ☐ 공TV 홈페이지 열기

At the bottom, there are buttons: "< 뒤로", "다음 >", and "취소".

**Opt-out 서비스 동의**



# 합법이다

**Opt-out** 검색엔진(bot, spider, crawler, etc.), 가격비교 등  
**위법?** 사이트 운영자의 의사에 반하지 않으면 합법

NAVER

검색결과

검색

통합검색

블로그

카페

지식IN

이미지

인간관계

모바일검색결과

내이름 검색

내이름검색

내이름검색

내이름검색

블로그

NAVER(095420) - 검색결과

검색결과

검색결과

검색결과

검색결과

Google

children's toy

검색결과 약 62,000,000개 (0.47초)

children's toy 관련 이미지

Children's Toys - IKEA

Children's Toys | Wooden Toys and Toy Kitchens |

Toys & Games - Amazon India

제주도 할인 항공권

좌석확보 항공권

항공사	김포 출발일시	제주 출발일시	기간	정상요금	할인율	할인요금
제주에어	11-02-17 (목) 06:55	11-02-18 (금) 21:05	1박 2일	126,400원	44%	70,784원
진에어	11-02-17 (목) 16:20	11-02-18 (금) 21:25	1박 2일	126,400원	55%	56,880원
아시아나항공	11-02-17 (목) 12:35	11-02-19 (토) 14:35	2박 3일	157,800원	30%	110,460원
제주에어	11-02-17 (목) 12:40	11-02-19 (토) 20:45	2박 3일	126,400원	46%	68,256원
제주에어	11-02-17 (목) 16:15	11-02-19 (토) 14:15	2박 3일	126,400원	42%	73,312원
대한항공	11-02-17 (목) 18:55	11-02-19 (토) 20:40	2박 3일	157,800원	35%	102,570원
제주에어	11-02-17 (목) 12:40	11-02-20 (일) 11:00	3박 4일	126,400원	34%	83,424원
아시아나항공	11-02-19 (토) 13:45	11-02-21 (월) 14:35	2박 3일	157,800원	30%	110,460원
대한항공	11-02-19 (토) 13:50	11-02-21 (월) 21:15	2박 3일	157,800원	25%	118,350원
제주에어	11-02-19 (토) 16:15	11-02-21 (월) 14:15	2박 3일	126,400원	42%	73,312원



# 불법이다



# 사례



## 대법원 "웹사이트 무단 크롤링은 불법"

김동훈 기자 99re@bizwatch.co.kr

2017.09.27(수) 17:37

잡코리아, 사람인 상대 9년 소송전서 승소  
크롤링(데이터 수집) 법적 기준 정립



## 야놀자 vs 여기어때 '진흙탕 싸움'..DB크롤링에서 비방 댓글까지 경찰 수사

여기어때, '야놀자 DB크롤링'..'업무방해'여부 두고 공방  
야놀자 임원 "여기어때 뉴스에 비방댓글 달아"..경찰 수사중

등록 2017-11-07 오후 2:52:03  
수정 2017-11-07 오후 2:55:15

가 가

# 이용방침

**이용방침 준수**      사이트 정보 이용에 대한 방침(안내, 약관 등) 미리 확인

## 이메일주소 무단수집 거부



본 웹사이트에 게시된 이메일 주소가 전자우편 수집 프로그램이나 그 밖의 기술적 장치를 이용하여 무단으로 수집되는 것을 거부하며 이를 위반시 정보통신망법에 의해 형사처벌됨을 유념하시기 바랍니다.

# robots.txt

## robots.txt

Crawler와 같은 (Ro)bot 접근을 제어하기 위한 규약  
대상 봇, 수집 여부, 수집 범위 등 기술

### Basic format:

User-agent: [user-agent name]

Disallow: [URL string not to be crawled]

이름	User-Agent
Google	Googlebot
Google image	Googlebot-image
Msn	MSNBot
Naver	Yeti <sup>[2]</sup>
Daum	Daumoa

```
""  
User-agent: *  
Allow: /  
""
```

```
""  
User-agent: *  
Disallow: /  
""
```

***어떤 사이트를 수집 하느냐 보다는, 어떤 데이터를 수집 하느냐가 문제!***

**1. Robots.txt**

접근 제약 규칙 준수

**2. Crawl delay**

사이트에 최대한 부담 지양

**3. Term of use**

사이트 이용방침(약관) 준수

**4. Public content**

지적재산권 침해 여부 주의

**5. Authentication-based sites**

민감한 정보 수집 주의

# Tips

## **builtwith**

Detect the technology used by a website

```
!pip install builtwith
```

```
from builtwith import builtwith  
builtwith('http://wordpress.com')  
builtwith('http://webscraping.com')  
builtwith('http://microsoft.com')  
builtwith('http://jquery.com')  
builtwith('http://joomla.com')
```

## **whois**

Retrieving WHOIS information of domains

```
!pip install python-whois
```

```
from whois import whois  
whois('http://wordpress.com')  
whois('http://webscraping.com')  
whois('http://microsoft.com')  
whois('http://jquery.com')  
whois('http://joomla.com')
```

urllib

# URL handling module

urllib	<b>urllib.request</b>	Opening and reading URLs
	<b>urllib.error</b>	Containing the exceptions raised by <i>urllib.request</i>
	<b>urllib.parse</b>	Parsing URLs
	<b>urllib.robotparser</b>	Parsing <i>robots.txt</i> files
	<b>urllib.response</b>	Used internally by the <i>urllib.request</i> module

```
from urllib import request
```



# robotparser

**RobotFileParser** This module provides a single class, **RobotFileParser**,  
**answers** questions about **whether or not** a particular user agent  
**can fetch a URL** on the Web site that published the robots.txt file

```
from urllib import robotparser

robot = robotparser.RobotFileParser()
robot.set_url('https://www.google.com/robots.txt')
robot.read()
robot.can_fetch('AgentName', 'https://www.google.com')
```

# request

**urlopen** Open the URL url, which can be either a **string** or a **Request object**

Returns a **http.client.HTTPResponse** object slightly modified

**geturl** return the **URL** of the resource retrieved

**info** return the **meta-information** of the page

**getcode** return the **HTTP status code** of the response

```
from urllib import request
```

```
resp = request.urlopen('https://www.google.com')
```

```
resp.geturl()
```

```
resp.reason
```

```
resp.getcode()
```

```
print(resp.info())
```

```
resp.getheaders()
```

# URL parameters

## parameters

Made of a **key** and a **value** separated by **=** and joined by **&**

First parameter always comes after **?** in a URL

```
https://www.google.com/search?newwindow=1&ei=vv94XLabGZLk_Aa9vomIAw
&q=%ED%8C%8C%EC%9D%B4%EC%8D%AC&oq=%ED%8C%8C%EC%9D%B4%EC%8
D%AC&gs_l=psy-ab.12...0.0..1777...0.0..0.0.0.....0.....gws-wiz.xlvFJMss_h8
```



```
{newwindow : 1}
{ei : vv94XLabGZLk_Aa9vomIAw}
{q : %ED%8C%8C%EC%9D%B4%EC%8D%AC} ← ASCII16 ← Hexadecimal ← Percent-Encoding
{oq : %ED%8C%8C%EC%9D%B4%EC%8D%AC}
{gs_l : psy-ab.12...0.0..1777...0.0..0.0.0.....0.....gws-wiz.xlvFJMss_h8}
```

# read

**read** Reads and returns the **response body**, or up to the next *amt* bytes

```
from urllib import request
```

```
resp = request.urlopen('https://www.google.com/search?q=%ED%8C%8C%EC%9D%B4%EC%8D%AC')  
resp.read()
```

```
[Out] b'<!doctype html> ...'
```

```
resp.read().decode('utf-8')
```

```
[Out] '<!doctype html> ...'
```

**HTTPError: HTTP Error 403: Forbidden**

# HTTP Status Codes

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

1xx: hold on

2xx: here you go

3xx: go away

4xx: you fucked up

5xx: I fucked up

# HTTPError

<b>HTTPError</b>	Handling exotic <b>HTTP errors</b> , such as requests for authentication
<b>code</b>	numeric value corresponds to an <b>HTTP status code</b>
<b>reason</b>	the <b>reason</b> for this error
<b>headers</b>	<b>HTTP response headers</b> for the HTTP request

```
from urllib import error

req = request.Request('https://www.google.com/search?q=%ED%8C%8C%EC%9D%B4%EC%8D%AC')

try:
    resp = request.urlopen(req)
except error.HTTPError as e:
    print(e.code)
    print(e.reason)
    print(e.headers)
```

**X-XSS-Protection** prevent some level of XSS (cross-site-scripting) attacks

# User-agent

robots.txt

```
User-agent: *
Disallow: /search
```

### ▼ Request Headers

**:authority:** www.google.com

```
:method: GET
```

**path:** /search?q=%ED%8C%8CEC%9D%B4%EC%8D%AC&oq=%ED%8C%8CEC%9D%B4%EC%8D%AC&qs=chrome..69i57j69i65l3j69i61l2.1195j0j4&sourceid=chrome&ie=UTF-

```
:scheme: https
```

**accept:** text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b2

**accept-encoding:** gzip, deflate, br

**accept-language:** ko-KR, ko;q=0.9,en-US;q=0.8,en;q=0.7

**cache-control:** no-cache

**cookie:** CGIC=InZ0ZXh0L2h0bWwsYXBwbGJjYXRpb24veGh0bWwreG1sLGFWcGxpY2F0aW9uL3htbDtxPTAu0SxpbWFnZS93ZWJwLGlTYWdlL2FwbmcsKi8qO3E9MC44LGFWcGxpY2F0\_nwt4oklmvv6VLcvlX-wm2Fl853Umbmt39ki18hZJQWs3gRMw.; HSID=AqvPMFavq-FVCBYMP; SSID=AqHWHmI-DnuBk56gF; APISID=jLYnZ9uhlXnawPzq/Ap9e3lwlnqxECfOTNylV26yU5XU20im2UXgXmd3u3PfAfCsA6Bis67RC3zmxAifyIGAWC5G7ypGDn0yP-CMZB9\_voXSjUxdPfK---r-ut3uXNLf0g4eor0RNwE4n0XyVfgcoDzu\_8rqob6ZVXRahmezbVa4x65K34IWiFndGga9Jg7NPwp8pwmHdBRk-1-t3CNE8gJBRCmkVKsLEd-okYEem-6IWIKBF64oYaY1h6VEjFuQHc60kZi93fUIJA56UH5itZIN8vMioafQcqPtTgKvg; 1P\_JAR=2019-03EyIHByb3ZlbmFuY2U6NiB0aw1lc3RhbmXA6MTU1MTQ20TYw0TY40TAwMCBsYXRsbmd7bGF0aXR1ZGVfZTC6Mzc10TCzMtYWIgxbmdpdHVkZV9lNzoXMjcwNDE1ODgwSfSbyYWrdpXMMTLq4SVhiy-8TBmjEVrr6hMLIDK

```
pragma: no-cache
```

**upgrade-insecure-requests: 1**

```
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.119 Safari/537.36
```

**x-client-data:** CIm2yQEIpLbJAQjBtskBCKmdygEIqKPKAQixp8oBCL+nygEI4qjKAQjqqcoBGPmlygE=

# urlparse

**urlparse** Break URL strings up in components (addressing scheme, network location, path etc.)

Combine the components back into a URL string

Convert a relative URL to an absolute URL given a base URL

```
from urllib import parse

parse.urlparse('https://www.google.com/search?q=%ED%8C%8C%EC%9D%B4%EC%8D%AC')
parse.urljoin('https://www.google.com/search?q=%ED%8C%8C%EC%9D%B4%EC%8D%AC','/search/about')
parse.urlencode({'q': '파이썬'})
parse.quote_plus('파이썬')
parse.unquote_plus('%ED%8C%8C%EC%9D%B4%EC%8D%AC')
```



requests

# URL handling module

The **Requests package** is recommended for a higher-level HTTP client interface.

requests {  
    requests.get  
    requests.post  
    requests.put  
    requests.delete  
    requests.head  
    requests.options  
    requests.response

*Making a request with Requests is **very simple***

```
import requests
```

exercises

**<http://httpbin.org>**