

# R 웹 크롤링 기초

연세대학교 대학원 응용통계학과 특강

# 웹 크롤링이란?

- 웹 크롤링(Crawling)은 웹 페이지에서 보이는 데이터를 필요한 부분만 선택하여 수집하는 행위를 말합니다. 스크레이핑(Scraping)이라고도 합니다. 웹 크롤링에 사용되는 프로그램을 크롤러라고 합니다.
- IE, 크롬과 같은 웹 브라우저 상에서 보이는 데이터는 크롤링이 가능하다고 할 수 있습니다. 그러므로 필요로 하는 데이터를 포함하고 있는 웹 사이트를 발견하는 것이 웹 크롤링의 시작이 됩니다.
- 웹 크롤링 방법은 웹 페이지에 따라 서로 다르게 적용해야 합니다. 본 강의를 통해 웹 크롤링에 필요한 다양한 방법을 익힐 수 있습니다.

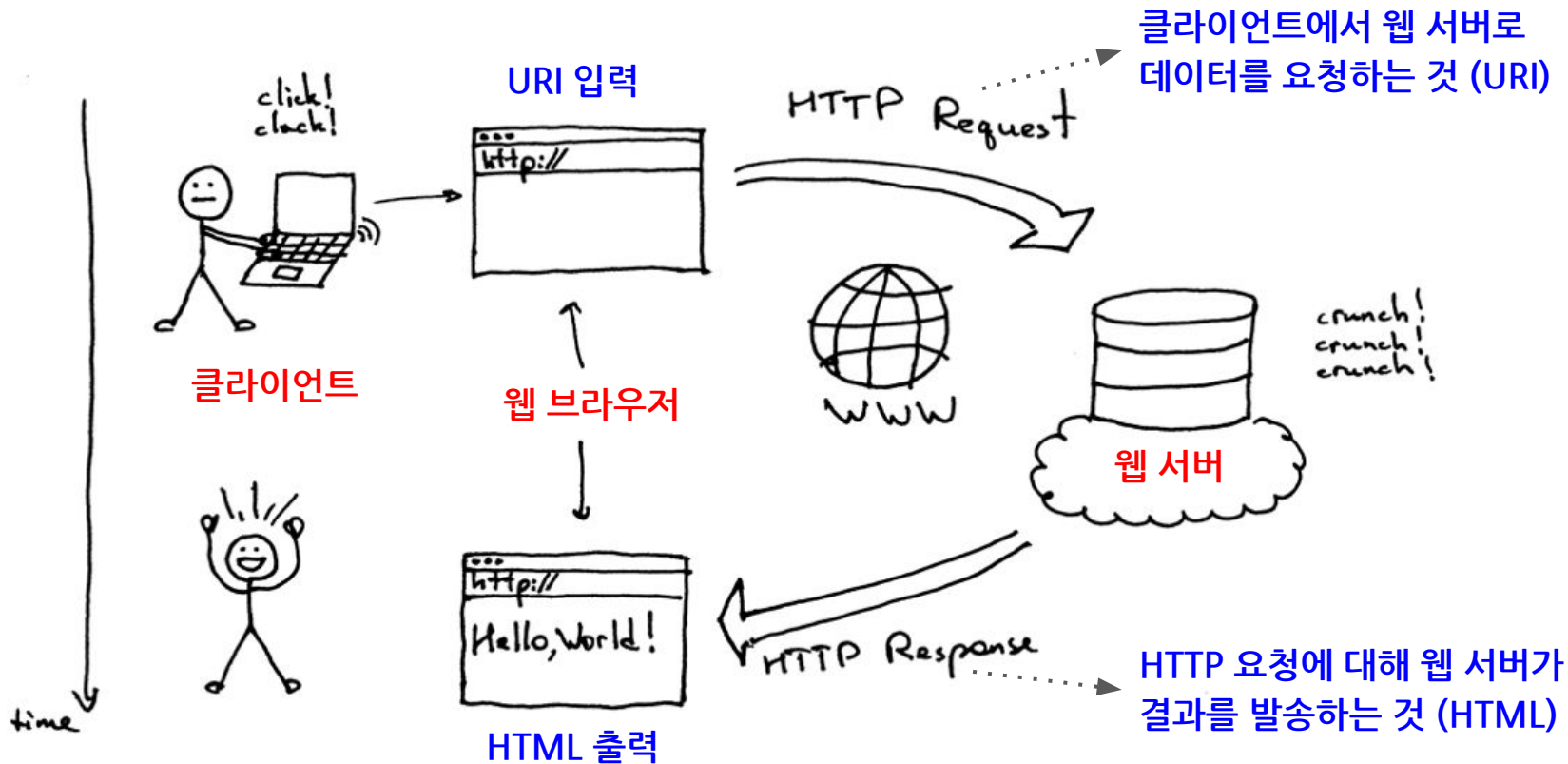
**[시연1] 네이버 실시간 검색어 수집**

# 이번 시간 강의 내용

1. 웹 크롤링 프로세스의 이해 및 주의사항
2. 반드시 알아두어야 할 기초 : HTTP 통신, HTML 요소, 인코딩 방식
3. 크롬 개발자도구 사용법 : Elements 탭에서 HTML 요소 찾기
4. 관련 R 패키지 및 주요 함수 소개
5. 웹 크롤링 시연 : 네이버 부동산 매물 리스트 수집 + 전처리
6. 혼자 공부할 때 참고자료

# 웹 크롤링 프로세스의 이해 및 주의사항

# 우리가 인터넷에서 정보를 검색하는 방법



# 웹 크롤링은 인터넷 검색과 유사

HTTP Request (요청)

- GET 방식과 POST 방식의 HTTP 통신
- JavaScript 및 RSelenium 이용

HTTP Response (응답)

- 응답 결과 확인 (상태코드, 인코딩 방식 등)
- 응답 받은 객체를 텍스트로 출력
- 응답 받은 객체에 찾는 HTML 포함 여부 확인

HTML에서 데이터 추출

- 응답 받은 객체를 HTML으로 변환
- CSS 또는 XPath로 HTML 요소 찾기
- HTML 요소로부터 데이터 추출

데이터 전처리 및 저장

- 텍스트 전처리 (결합, 분리, 추출, 대체)
- 다양한 형태로 저장 (RDS, Rdata, xlsx, csv 등)

httr  
urltools  
RSelenium

rvest  
jsonlite

stringr  
dplyr

## 추가로 알아야 할 사항

크롬 개발자도구

인코딩 및 로케일

다양한 에러 해결법

정규표현식 (Regex)



# 웹 크롤링 관련 주의사항

- 웹 페이지는 회사가 비즈니스를 영위할 목적으로 만든 것입니다.
- 웹 크롤링은 ‘영업권 및 지적재산권’을 침해하는 행위로 **민사소송**에 휘말릴 수 있습니다. (잡코리아, 사람인에 승소)
  - 관련 뉴스 : [http://it.chosun.com/site/data/html\\_dir/2017/09/27/2017092785016.html](http://it.chosun.com/site/data/html_dir/2017/09/27/2017092785016.html)
- 따라서 웹 크롤링 하려는 웹 사이트의 메인 페이지에서 사전에 **‘robots.txt’**를 **확인**해야 하며, 특히 수집한 데이터를 영업에 사용할 목적이라면 **반드시 법률 검토를 진행**하시기 바랍니다.

# HTTP 기초

HTTP 요청(Request) 및 응답(Response)

# HTTP (HyperText Transfer Protocol)

- HTTP는 ‘초본문 전송 규약’이라고 번역할 수 있는데, 인터넷(world wide web) 상에서 데이터를 주고 받을 때 사용되며, 주로 HTML을 주고 받습니다.
- 데이터를 주고 받는 당사자는 ‘클라이언트(Client)’와 ‘웹서버(Web Server)’입니다.
- 클라이언트가 웹서버에 데이터를 요청(Request)하고, 웹서버는 해당 요청에 대한 결과를 응답(Response)합니다.
- 클라이언트가 요청할 때 사용할 수 있는 방식(Method)에는 여러 가지가 있으며, 가장 많이 사용되는 것이 GET 방식과 POST 방식입니다.

# HTTP 요청 (Request)

- 클라이언트가 웹서버에 HTTP 요청을 할 때, 웹서버에 제공해야 하는 **‘요청메시지’**는 방식에 따라 다릅니다.
- GET 방식은 **‘요청 라인’**과 **‘요청 헤더’**를 보내야 하고, POST 방식은 위 2가지에 **‘메시지 바디’**를 추가해야 합니다.

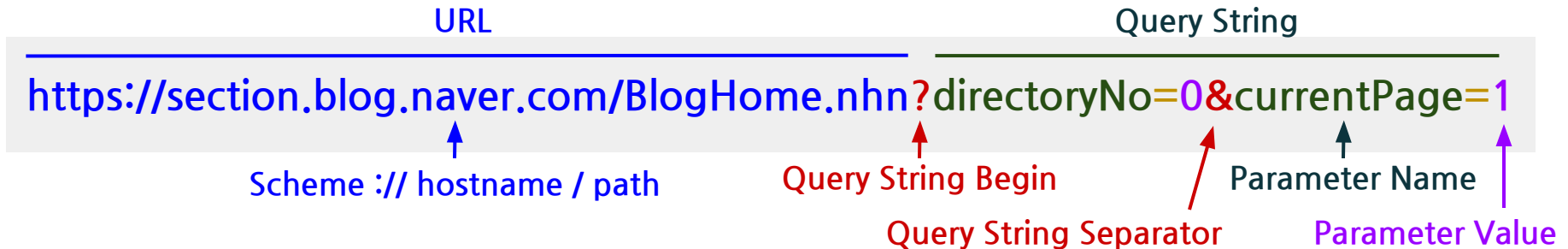
요청 라인	<ul style="list-style-type: none"><li>- 요청 방식 (GET or POST)</li><li>- <b>경로 (URI)</b></li><li>- HTTP 버전 등</li></ul>
요청 헤더	<ul style="list-style-type: none"><li>- 가능한 콘텐츠 형식 (Content-Type)</li><li>- 가능한 인코딩 방식 (Character set)</li><li>- 인증 스펙 (Authorization)</li><li>- Cookies</li><li>- <b>User-agent</b></li><li>- <b>Referer</b> 등</li></ul>
메시지 바디	<ul style="list-style-type: none"><li>- URI에 포함된 파라미터들을 할당 (길이 제한 없음)</li></ul>

# GET 방식과 POST 방식에 대한 이해

- 영어사전에서 'GET'으로 찾아보면 기본적으로 '받다, 가져오다' 등의 의미를 갖습니다.
- 웹 브라우저의 주소창에서 보이는 **URI**만 가지고 웹 서버에 요청할 수 있는 간단한 방법입니다.
- 영어사전에서 'POST'로 찾아보면 동사로는 '붙이다, 게시하다' 등의 의미를 갖습니다.
- 웹 브라우저의 주소창에서 보이는 URI로는 원하는 결과를 얻을 수 없으며, 크롬 개발자도구에서 관련 **URL**과 **Parameters**를 찾아야 하는 다소 복잡한 방법입니다.

# URL vs URI

- URI는 **Uniform Resource Indicator**의 머리글자, 리소스를 식별하는 문자열들을 차례대로 배열한 것입니다.
- URL은 **Uniform Resource Locator**의 머리글자, 리소스가 포함되어 있는 위치를 의미합니다. URL은 URI의 부분집합이라고 생각하면 됩니다.
- [예시] 네이버 블로그 메인화면 주소(URI) : URL과 Query String의 조합



# HTTP 응답 (Response)

- 웹 서버는 클라이언트의 요청에 대해 **‘응답메시지’**를 발송합니다.
- 응답메시지에는 **‘응답 헤더’**와 **‘바디’**로 구성되어 있습니다.
- 응답 헤더는 HTTP 버전, **상태코드**, 일시, **콘텐츠 형태**, **인코딩 방식**, 크기 등이 포함되며, 바디에는 **HTML**이 포함됩니다.

상태코드	내용
1XX	정보 교환
2XX	데이터 전송 성공 or 수락됨
3XX	방향 바꿈
4XX	클라이언트 오류 ( 주소 오류, 권한 없음, 접근 금지 등 )
5XX	서버 오류 ( 올바른 요청을 처리할 수 없음 등 )

# HTML 기초

HTML 요소(Element)의 구조



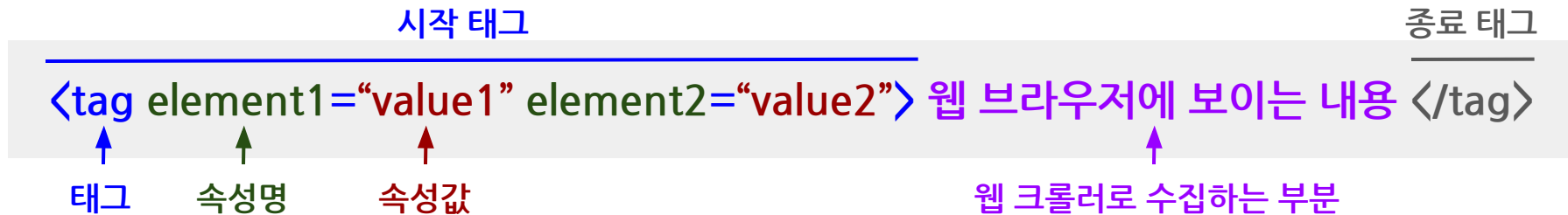
# HTML (HyperText Markup Language)

- HTML은 웹 페이지의 제목, 단락, 목록 등 **문서의 구조를 나타내는 마크업 언어**입니다.
- HTML은 꺾쇠 괄호 ‘< >’ 안에 태그로 되어 있는 HTML 요소 형태로 작성됩니다.
- HTML의 디자인을 담당하는 **CSS**와 웹 브라우저를 제어하는 **JavaScript**를 함께 사용함으로써 상호작용하는 웹 페이지를 구현할 수 있습니다.



# HTML element (요소)

- HTML 요소는 HTML 문서나 웹 페이지를 구성하는 개별 항목을 의미합니다.
- HTML 요소는 시작 태그와 종료 태그로 작성되며, 그 사이에 내용이 포함됩니다.
- 태그는 꺾쇠 괄호로 감쌉니다. 시작태그에 속성명과 속성값이 포함되고, 종료 태그에는 ‘/’가 추가됩니다.
- 웹 크롤링은 수집하려는 부분을 포함하는 HTML 요소를 찾는 것이 필수입니다.



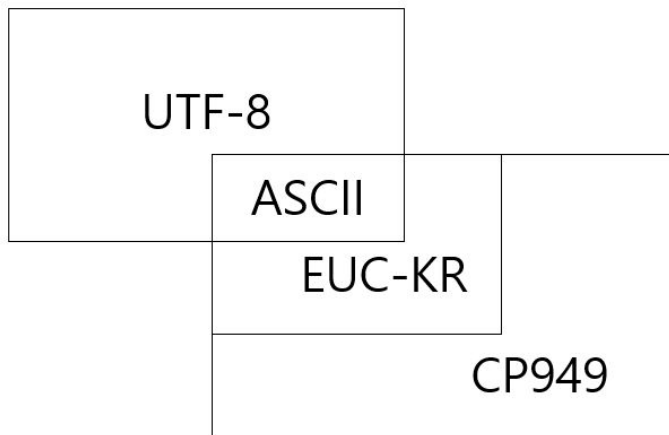
# 다양한 인코딩 방식의 이해

# 한글 인코딩이란?

- 한글 인코딩은 **‘한글을 컴퓨터에 표시하는 방식’**을 말합니다.
  - 사람들이 사용하는 문자(자연어)를 컴퓨터는 읽을 수 없습니다. 대신에 0과 1로 된 2진수를 사용하였고, 2진수는 자리수를 많이 차지하므로 나중에 8진수, 16진수 등으로 발전했습니다.
  - 반대로 사람은 16진수로 된 문자를 제대로 읽을 수 없습니다. 따라서 **사람들의 문자를 컴퓨터가 이해할 수 있도록 16진수로 표기한 것이 한글 인코딩**입니다.
- 한글 인코딩에 주로 사용되는 방식은 크게 **‘EUC-KR’**과 **‘UTF-8’**이 있습니다.

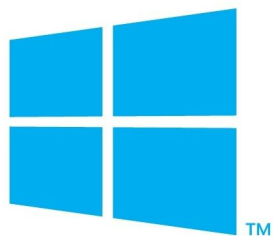
사람이 이해하는 문자	한글 인코딩 방식	컴퓨터가 이해하는 코드
가	EUC-KR	B0A1
	UTF-8	U+AC00

# 한글 인코딩 방식 관계도



구분	설명
ASCII	<ul style="list-style-type: none"> <li>● 미국에서 개발된 대표적인 영문 인코딩 방식</li> <li>● 알파벳, 숫자, 기호 등 128개가 지정되어 있음</li> </ul>
UTF-8	<ul style="list-style-type: none"> <li>● 유니코드에 기반한 인코딩 방식 (가변 길이)</li> <li>● <b>한글 완성형과 조합형이 모두 포함 (국제 표준)</b></li> </ul>
Unicode	<ul style="list-style-type: none"> <li>● 전 세계 모든 문자를 포함하는 표준 문자 방식</li> <li>● UTF-8, UTF-16, UTF-32 인코딩 방식 있음</li> </ul>
EUC-KR	<ul style="list-style-type: none"> <li>● 한글 초기 완성형 인코딩 방식 (2350자)</li> <li>● 조합형은 다른 문자 체계와 호환이 되지 않음</li> </ul>
CP949	<ul style="list-style-type: none"> <li>● MS가 8822자를 추가한 통합 완성형 인코딩 방식</li> <li>● <b>Windows 점유율이 높은 한국에서 '사실상 표준'</b></li> </ul>

# 컴퓨터 운영체제별 한글 인코딩 방식



운영체제

EUC-KR  
(CP949)

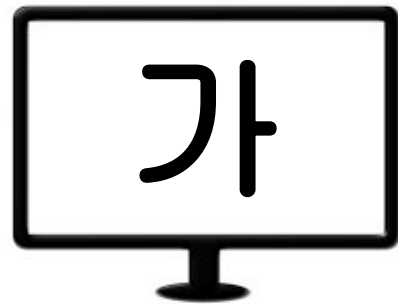
기본 한글  
인코딩 방식

B0A1

문자코드

Decoding

Encoding



화면에 보이는 문자



UTF-8

U+AC00

## 한글 인코딩 방식 지정 필요성 (외부 데이터)

인코딩 방식 지정 구분 (O : 필요함, X : 필요 없음)		컴퓨터 운영체제별 한글 인코딩 방식	
		Windows (CP949)	Mac (UTF-8)
HTML의 인코딩 방식	EUC-KR (CP949)	X	O
	UTF-8	O	X
	기타	O	O

# 로케일(Locale)이란?

- 국가마다 서로 다른 문화를 가지고 있다 보니, 시간을 표시하는 방법이라든가 숫자를 표시하는 방법 등에서 상당한 차이를 보입니다. 따라서 운영체제(OS)는 국가마다 서로 다른 로케일을 제공하고 있습니다.
- 로케일은 국가마다 다음과 같은 여러 가지 표시 형식을 설정하는 것입니다.
  - LC\_COLLATE(문자 정렬), LC\_CTYPE(문자 처리), LC\_MONETARY(통화), LC\_NUMERIC(숫자), LC\_TIME(날짜/시간), LC\_MESSAGES(언어/문화) 등



# 로케일을 왜 알아야 하나?

- 한글 인코딩 방식이 로케일에 영향을 받기 때문입니다.
- 아울러 EUC-KR 또는 CP949 인코딩 방식의 R 객체를 처리하지 못하는 함수가 있습니다. `html_table()` 함수가 한 예입니다.

## [참고] 국가별 로케일 이름 (locale 인자에 추가하는 이름)

로케일	Windows		Mac	
	로케일 이름	인코딩 방식	로케일 이름	인코딩 방식
대한민국	korean	CP949	ko_KR	UTF-8
미국	english	ISO8859-1	en_US	UTF-8
중국	chinese	CP936	zh_CN	UTF-8
일본	japanese	CP932	ja_JP	UTF-8
기본값	C	ASCII	C	ASCII

## 퍼센트 인코딩이란?

- 퍼센트 인코딩(percent-encoding)이란 URL에 사용되는 문자를 인코딩하는 방식이며 URL 인코딩이라고도 합니다. 아울러 한글 인코딩 방식에 따라 결과가 달라집니다. 예를 들어 ‘웹크롤링’을 퍼센트 인코딩한 결과는 다음과 같습니다.

UTF-8	%EC%9B%B9%ED%81%AC%EB%A1%A4%EB%A7%81
EUC-KR	%C0%A5%C5%A9%B7%D1%B8%B5

- 네이버 메인에서 웹크롤링으로 검색한 다음, 변경된 url를 복사하여 텍스트로 붙여넣기 해보시면 query 파라미터에 할당된 값이 아래와 같이 바뀝니다.
  - [https://search.naver.com/search.naver?sm=top\\_hyt&fbm=1&ie=utf8&query=%EC%9B%B9%ED%81%AC%EB%A1%A4%EB%A7%81](https://search.naver.com/search.naver?sm=top_hyt&fbm=1&ie=utf8&query=%EC%9B%B9%ED%81%AC%EB%A1%A4%EB%A7%81)

# 크롬 개발자도구 사용법

Elements 탭에서 HTML 요소 찾는 방법

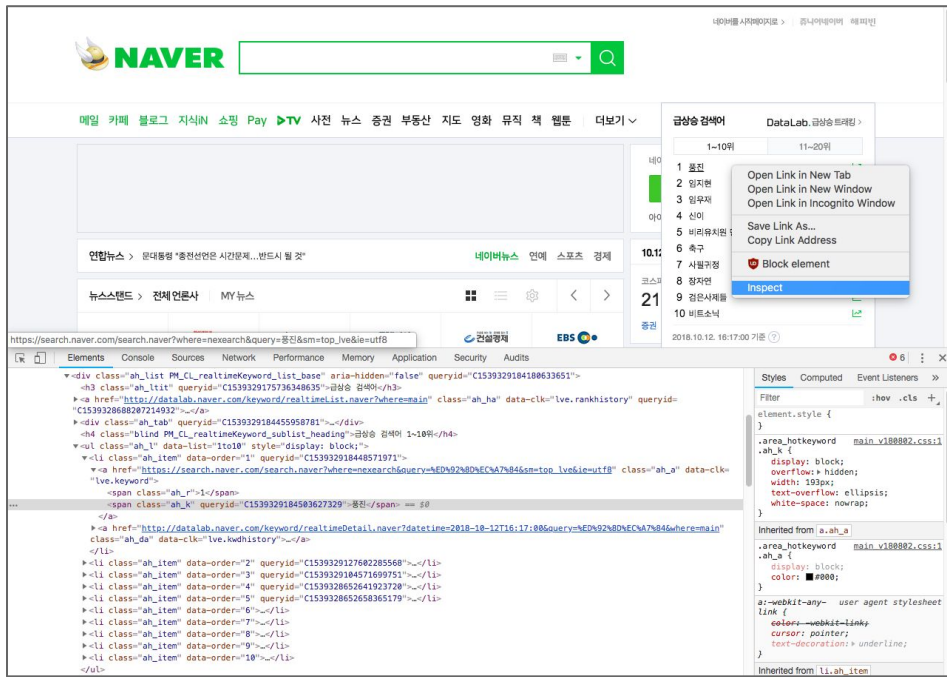
# 크롬 개발자도구를 사용하는 이유

- ‘크롬 개발자도구’는 웹 페이지에서 수집하려는 내용을 담고 있는 HTML 요소를 찾거나, HTTP 요청 과정에서 클라이언트와 웹 서버 간 주고 받은 (사용자에게 보이지 않는) 리소스를 찾고자 할 때 사용합니다.
- 크롬 메뉴에서 **‘도구 더보기(More Tools) -> 개발자도구(Developer Tools)’**를 선택하면 됩니다.
- 크롬 개발자도구에서 제공되는 탭은 Elements 외 8개이지만, 이번 강의에서는 **Elements**와 **Network**만 사용합니다.

# HTML 요소 찾기 (첫 번째 방법)

- 크롬 브라우저에서 웹 사이트로 이동한 다음, 수집하려는 내용 위에서 **마우스 오른쪽 버튼을 클릭**하면 메뉴가 뜹니다.

- '검사(Inspect)' 메뉴를 선택**하면 크롬 개발자도구가 열리며 **해당 HTML 요소가 파란색으로 표시**됩니다.



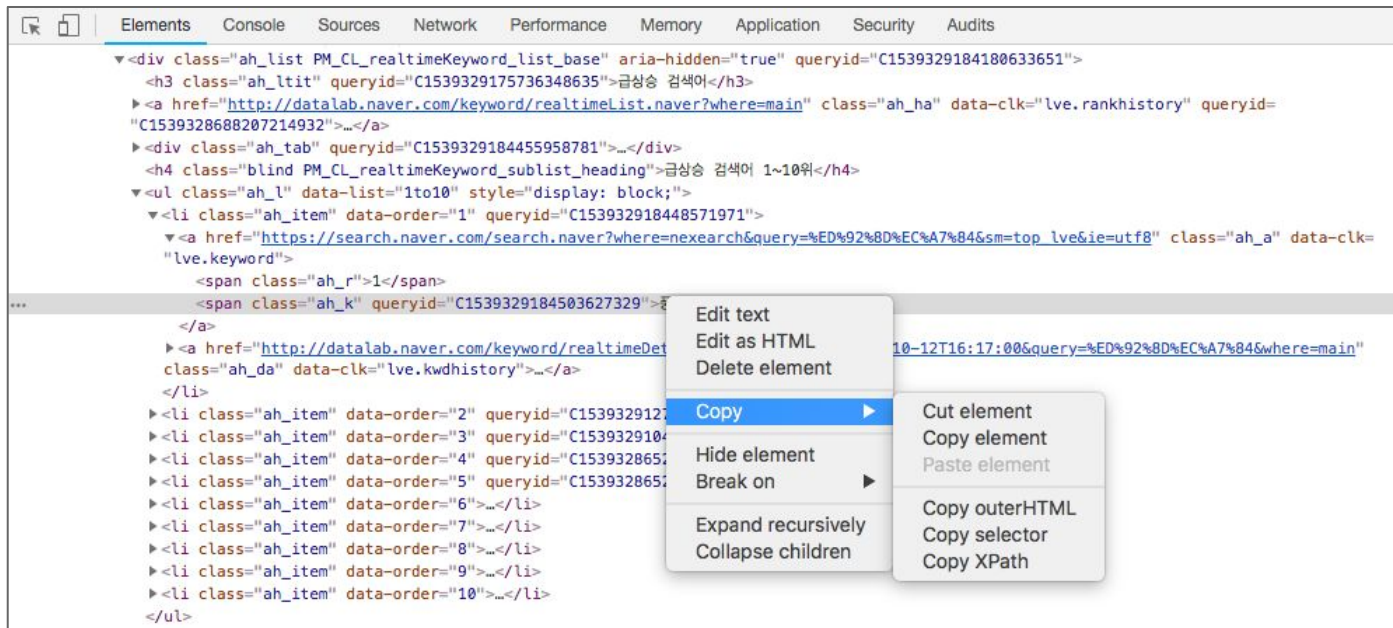
# HTML 요소 찾기 (두 번째 방법)

- 크롬 개발자도구 화면에서 (아래 빨간 점선으로 표시된) 버튼을 한 번 클릭한 다음 웹 페이지에서 원하는 정보를 선택하면, Elements 탭에서 해당 HTML 요소가 파란색으로 표시됩니다.



# HTML 요소 지정 : CSS Select 또는 XPath

- HTML 요소 위에 마우스 오른쪽 버튼을 클릭한 다음 'Copy'를 선택하면 하위 메뉴 창이 열립니다. 이 때, 'Copy selector' 또는 'Copy XPath'를 선택하면 됩니다.





# 네이버 실시간 검색어 1위 HTML 요소의 CSS와 Xpath 비교

- CSS Selector

```
#PM_ID_ct > div.header > div.section_navbar >  
div.area_hotkeyword.PM_CL_realtimeKeyword_base >  
div.ah_list.PM_CL_realtimeKeyword_list_base > ul:nth-child(5) >  
li:nth-child(1) > a.ah_a > span.ah_k
```

- Xpath

```
//*[@id="PM_ID_ct"]/div[1]/div[2]/div[2]/div[2]/ul[1]/li[1]/a[1]/span[2]
```

# CSS Selector과 XPath 비교

- **CSS는 HTML의 디자인을 담당**합니다. HTML 요소에 포함된 Selector를 참조하여 웹 브라우저에 출력되는 모습을 변경합니다. (폰트, 컬러, 크기, 굵기 등)
- **XPath는 XML Path Language**를 나타내며, 계층 구조를 갖는 XML 문서에서 노드(HTML의 태그)를 탐색하는 경로로 사용됩니다.
- 이 두 가지는 HTML 요소를 지정할 때 (우열 없이) 사용됩니다. 다만 XPath의 경우 Selenium에서 사용될 경우 좀 더 빠르다는 의견이 있습니다. 이번 강의에서는 일반적인 크롤러에서는 CSS Selector를 사용하고, RSelenium에서는 XPath를 사용하여 HTML 요소를 지정하도록 하겠습니다.

# CSS Selector와 XPath 표기법 [매우 중요!]

CSS Selector		XPath	
표현식	의미	표현식	의미
태그명	태그명이 같은 모든 태그를 선택	노드명	노드명이 같은 모든 노드를 선택
>	앞 태그의 직계 자손 태그에서 선택	/	루트노드부터 탐색. 부모노드/자식노드 경로
#	속성명이 'id'인 태그를 선택	//	위치와 상관없이 지정된 노드부터 탐색
.	속성명이 'class'인 태그를 선택	.	현재노드 선택 ('..'는 부모노드 선택)
[ ]	속성을 지정할 때 사용	@	속성노드 선택
:nth-child	n번째 태그를 선택할 때 사용	[ ]	속성 지정 및 n번째 노드를 선택할 때 사용

# CSS Selector와 XPath 표기법 [매우 중요!]

목표	CSS Selector	XPath
모든 요소	*	//*
p 태그를 포함하는 모든 요소	p	//p
p 태그의 모든 자식 요소	p > *	//p/*
id가 'foo'인 모든 요소	#foo	//*[@id='foo']
class가 'foo'인 모든 요소	.foo	//*[@class='foo']
title='header' 속성을 포함하는 모든 요소	*[title='header']	//*[@title='header']
li 태그 중 n번째인 요소	li:nth-child(n)	//li[n]
ul 자식 태그인 li 태그 중 n번째인 요소	ul > li:nth-child(n)	//ul/li[n]

# 관련 R 패키지 및 주요 함수 소개

# 웹 크롤링 관련 R 패키지 목록

- HTTP 통신 : **httr**, RSelenium
- HTML 요소 : **rvest**, jsonlite
- 인코딩 관련 : **urltools**, readr
- 파이프 연산자 : magrittr (**dplyr**)
- 텍스트 전처리 : **stringr**
  - tidyverse 패키지를 불러오면 8개의 패키지를 한 번에 불러올 수 있는데 그 중에 readr, dplyr 및 stringr 패키지가 포함되어 있습니다.

# HTTP 통신 관련 : http 패키지 소개

- http은 HTTP 요청 및 응답에 관한 작업에 사용되는 패키지입니다.
- 주요 함수는 다음과 같습니다.
  - HTTP 요청에 관한 함수들 : **GET()**, **POST()**, `user_agent()`, `add_headers()`, `set_cookies()`
  - HTTP 응답에 관한 함수들 : **status\_code()**, **content()**, `cookies()`, `headers()`
  - HTTP 응답에 성공하지 못했을 때 사용하는 함수들 : `warn_for_status()`, `stop_for_status()`

# httr 패키지 주요 함수 1 : GET 방식의 HTTP 요청

- `res <- GET(url = '요청할 웹 페이지 URL',  
            query = list(a = 'a에 할당된 값',  
                          b = 'b에 할당된 값'),  
            ... )`
  - GET 방식의 HTTP 통신이 사용된 경우, GET() 함수를 사용합니다.
  - url 인자에 웹 페이지의 URL 부분을 할당하고,
  - query 인자에 query string을 **list** 형태로 할당합니다.



## httr 패키지 주요 함수 2 : POST 방식의 HTTP 요청

- `res <- POST(url = '요청할 웹 페이지 URL',  
body = list('POST 방식 요청에 사용될 파라미터'),  
encode = c('multipart', 'form', 'json', 'raw'))`
  - POST 방식으로 HTTP 통신하는 경우에는 `POST()` 함수를 사용해야 합니다.
  - `GET()` 함수와 달리 `POST()` 함수는 `query` 인자 대신 `body`와 `encode` 인자를 추가합니다.
  - `body`와 `encode` 인자에 지정하는 값은 크롬 개발자도구의 네트워크 탭에서 찾습니다.
  - `encode`의 경우, 4가지 중 해당하는 한 가지를 선택하거나 생략할 수 있습니다.

# httr 패키지 주요 함수 3 : 상태코드 및 응답 결과 확인

- **print**(x = res)
  - HTTP 응답 결과를 한 번에 출력합니다.
- **status\_code**(x = res)
  - HTTP 응답 상태코드만 출력합니다. 사용자 정의 함수 등에서 유용하게 사용할 수 있습니다.
- **content**(x = res, as = 'text', type = 'text/html', encoding = 'EUC-KR')
  - HTTP 응답 바디(HTML)를 텍스트 형태로 출력하여 육안으로 확인합니다.
  - type과 encoding 인자는 추가하지 않아도 자동으로 설정됩니다.
  - 하지만 encoding 인자는 상황에 따라 반드시 추가해야 하는 경우가 있습니다.

# HTML 요소 관련 : rvest 패키지

- **rvest**는 웹 페이지로부터 데이터를 수집할 때 사용하는 패키지입니다.
- 주요 함수는 다음과 같습니다.
  - 응답 객체를 HTML로 변환하는 함수 : **read\_html()**
  - HTML 요소를 추출하는 함수 : **html\_node()**, **html\_nodes()**
  - HTML 속성에 관련된 함수 : **html\_attr()**, **html\_attrs()**, **html\_name()**
  - 데이터를 추출하는 함수 : **html\_text()**, **html\_table()**

# rvest 패키지 주요 함수 1 : 응답 객체에서 HTML 읽기

- `html <- read_html(x = res, encoding = 'UTF-8')`
  - HTTP 응답 객체인 `res`로부터 HTML을 읽은 다음 `html` 객체에 할당합니다. 이 때 주의해야 할 사항은 **res 객체의 한글 인코딩 방식을 지정해주어야 한다**는 것입니다.
  - `res` 객체의 한글 인코딩 방식을 확인하는 방법은 `print(x = res)`를 실행하면 됩니다.
  - Windows 컴퓨터는 한글 인코딩 방식으로 'EUC-KR'을 사용하므로, HTTP 응답 객체의 한글 인코딩 방식이 'EUC-KR'이 아닌, 예를 들어 'UTF-8'이면 반드시 `encoding` 인자를 추가해주어야 합니다.
  - 반대로 Mac 컴퓨터는 한글 인코딩 방식으로 'UTF-8'을 사용하므로, HTTP 응답 객체의 한글 인코딩 방식이 'UTF-8'이 아닌, 예를 들어 'EUC-KR'이면 반드시 `encoding` 인자를 추가해주어야 합니다.

## rvest 패키지 주요 함수 2 : HTML 요소 찾기

- `item <- html_node(x = html,`  
`css = '크롬 개발자도구에서 복사해온 CSS Selector',`  
`xpath = '크롬 개발자도구에서 복사해온 Xpath')`
  - `html_node()` 함수 인자 중 `css`와 `xpath` 둘 중 하나만 사용하면 됩니다.
  - `html_node()` 함수와 `html_nodes()` 함수는 사용법이 같습니다. 다른 점은 `html_node()` 함수는 찾고자 하는 HTML 요소가 여러 개 있을 때 맨 처음 하나만 가져오지만, `html_nodes()` 함수는 모든 HTML 요소를 가져옵니다. 따라서 일반적으로 `html_nodes()` 함수가 더 유용합니다.
  - `html_node()` 함수는 **찾는 HTML 요소가 없으면 NA를 반환**합니다.

## rvest 패키지 주요 함수 3 : HTML에서 텍스트 추출

- `text <- html_text(x = item, trim = FALSE)`
  - HTML 요소 중 시작 태그와 종료 태그 사이에 있는 '웹 브라우저에 보이는 내용'을 수집하려고 할 때 `html_text()` 파일을 사용하면 됩니다.
  - `html_node()` 함수 또는 `html_nodes()` 함수로 추출한 HTML 요소에 있는 모든 텍스트를 추출합니다.
  - 'trim' 인자에 FALSE가 기본값으로 설정되어 있으나, TRUE로 추가하면 문자열 양 옆의 불필요한 여백을 깔끔하게 제거할 수 있습니다.

## rvest 패키지 주요 함수 4 : 표에 있는 내용 일괄 수집

- `tbl <- html_table(x = item, fill = FALSE)`
  - HTML의 'table' 태그는 웹 브라우저에서 표 형태로 데이터를 출력합니다. 테이블 안에 포함된 모든 데이터를 데이터 프레임 형태로 수집하고자 할 때 `html_table()` 함수를 사용합니다.
  - Mac 사용자의 경우, 이 함수를 사용할 때 전혀 문제가 발생하지 않지만 Windows 사용자는 에러가 발생합니다. 그 이유는 한글 인코딩 방식 때문입니다.
  - 이러한 에러를 우회하는 방법으로 '로케일 변경'을 시도할 수 있습니다.

# 파이프 연산자 관련 : magrittr 패키지

- `text <- res %>%  
 read_html(encoding = 'UTF-8') %>%  
 html_node(css = 'CSS Selector', xpath = 'XPath') %>%  
 html_text(trim = TRUE)`
  - 이 강의에서는 **파이프 연산자(%>%)**를 사용합니다.
  - 파이프 연산자는 이전 함수 실행 결과를 다음 함수의 첫 번째 인자로 전달하기 때문에 코드의 흐름대로 이해할 수 있으며, 불필요한 객체를 여러 개 만들 필요가 없습니다.
  - 파이프 연산자를 사용하려면 **magrittr** 패키지를 로딩해야 합니다.



## 한글 인코딩 관련 : 기본 패키지 + readr 패키지

- **localeToCharset()** : 컴퓨터에 설정된 로케일의 문자 인코딩 방식을 확인합니다.
- **iconv**(x = '문자열', from = 'UTF-8', to = 'EUC-KR') : 인코딩 방식을 변경합니다.
- **readr::guess\_encoding**(file = '파일명') : 컴퓨터에 저장된 파일 또는 URL의 문자 인코딩 방식을 확인합니다.

# 로케일 관련 : 기본 패키지

- **Sys.getlocale()** : 현재 설정된 로케일을 확인합니다.
- **Sys.setlocale**(category = 'LC\_ALL', locale = 'locale name' ) : 로케일 설정을 변경합니다.
  - category 인자에는 'LC\_COLLATE', 'LC\_CTYPE', 'LC\_MONETARY', 'LC\_NUMERIC', 'LC\_TIME' 등 개별 카테고리를 지정할 수 있으나 편의상 'LC\_ALL'로 지정하도록 합니다.
  - locale 인자에 지정할 locale name은 운영체제에 따라 서로 다릅니다.
  - Sys.setlocale() 함수 안에 아무런 인자를 지정하지 않으면 운영체제의 기본값으로 설정됩니다.

# 퍼센트 인코딩 관련 : urltools 패키지

- 퍼센트 인코딩된 문자열은 **urltools** 패키지를 이용합니다.
  - **url\_decode**(urls = '문자열A') : 문자열A를 퍼센트 디코딩하여 사람이 읽을 수 있도록 합니다.
  - **url\_encode**(urls = '문자열B') : 문자열B를 퍼센트 인코딩하여 컴퓨터가 읽을 수 있도록 합니다.

# 텍스트 전처리 관련 : stringr 패키지

- stringr 패키지는 문자 데이터를 다루는 데 필요한 주요 함수를 담고 있습니다.
- R 기본 함수로도 충분히 가능하지만, 문제는 파이프 연산자를 사용할 수 없는 경우가 있다는 것입니다. 따라서 이번 강의에서는 stringr 패키지를 사용합니다.
- stringr 패키지의 주요 함수로는 다음과 같은 작업을 할 수 있습니다.
  - 패턴을 포함하고 있는지 확인 (str\_detect)
  - 패턴을 삭제(str\_remove)하거나 교체(replace) 또는 추출(extract) 및 인덱스로 자르기(sub)
  - 문자열을 하나로 묶음(str\_c) 또는 분리(str\_split)
  - 문자열의 양 옆에 있는 공백 제거(str\_trim)

## stringr 패키지 주요 함수 1 : 패턴 포함 여부 확인

- 문자열을 담고 있는 객체에서 찾고자 하는 패턴이 포함되어 있는지 확인할 수 있습니다.

```
> string <- '동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라 만세'
```

```
> string %>% str_detect(pattern = '우리나라')
```

```
## [1] TRUE
```

```
> string %>% str_detect(pattern = '하나님')
```

```
## [1] FALSE
```

## stringr 패키지 주요 함수 2 : 패턴을 한 번 또는 모두 삭제

- 문자열에서 맨 처음 나오는 패턴을 한 번 삭제할 수 있습니다.

```
> string %>% str_remove(pattern = '')
```

```
## [1] '동해물과백두산이 마르고 닳도록 하느님이 보우하사 만세'
```

- 지정한 패턴이 여러 번 나오는 경우, 모두 삭제할 수 있습니다.

```
> string %>% str_remove_all(pattern = '')
```

```
## [1] '동해물과백두산이마르고닳도록하느님이보우하사우리나라만세'
```

## stringr 패키지 주요 함수 3 : 패턴을 한 번 또는 모두 교체

- 문자열에서 맨 처음 나오는 패턴을 한 번 교체할 수 있습니다.

```
> string %>% str_replace(pattern = ‘’, replacement = ‘_’)
```

```
## [1] ‘동해물과_백두산이 마르고 닳도록 하느님이 보우하사 우리나라 만세’
```

- 지정한 패턴이 여러 번 나오는 경우, 모두 교체할 수 있습니다.

```
> string %>% str_replace_all(pattern = ‘’, replacement = ‘_’)
```

```
## [1] ‘동해물과_백두산이_마르고_닳도록_하느님이_보우하사_우리나라_만세’
```

## stringr 패키지 주요 함수 4 : 패턴을 한 번 또는 모두 추출

- 문자열에서 맨 처음 나오는 패턴을 한 번 추출할 수 있습니다.

```
> string %>% str_extract(pattern = '우')  
## [1] '우'
```

- 지정한 패턴이 여러 번일 때 모두 추출할 수 있습니다. 단, 결과는 리스트입니다. 함수 안에 '**simplify = TRUE**'를 추가하면 리스트 대신 행렬로 출력됩니다.

```
> string %>% str_extract_all(pattern = '우')  
## [[1]]  
## [1] '우' '우'
```



## stringr 패키지 주요 함수 5 : 문자열을 인덱스로 자르기

- 문자열의 인덱스를 이용하여 필요한 부분만 자를 수 있습니다.

```
> string %>% str_sub(start = 1, end = 2)
```

```
## [1] 동해
```

```
> string %>% str_sub(start = 3, end = 4)
```

```
## [1] 물과
```

```
> string %>% str_sub(start = 6, end = 7)
```

```
## [1] 백두
```

## stringr 패키지 주요 함수 6 : 문자열을 하나로 묶음

- 두 개 이상의 문자열을 하나의 커다란 문자열로 묶을 수 있습니다.

```
> str_c('우리나라', '만세')
```

```
## [1] 우리나라만세
```

```
> str_c('우리나라', '만세', sep = '')
```

```
## [1] 우리나라만세
```

## stringr 패키지 주요 함수 7 : 문자열을 구분자로 분리

- 하나의 문자열을 구분자 기준으로 여러 개의 문자열로 분리할 수 있습니다.  
단, 리스트 객체로 결과가 반환됩니다.

```
> string %>% str_split(pattern = ‘ ’)
```

```
## [[1]]
```

```
## [1] ‘동해물과’ ‘백두산이’ ‘마르고’ ‘닿도록’ ‘하느님이’ ‘보우하사’ ‘우리나라’ ‘만세’
```

## stringr 패키지 주요 함수 8 : 양 옆의 공백 제거

- 문자열 양 옆에 있는 불필요한 공백을 제거할 수 있습니다.

```
> string <- '\r\n\t\t\t\t 하느님이 보우하사 \r\n\t\t\t\t\t'
```

```
> string %>% str_trim()
```

```
## [1] 하느님이 보우하사
```

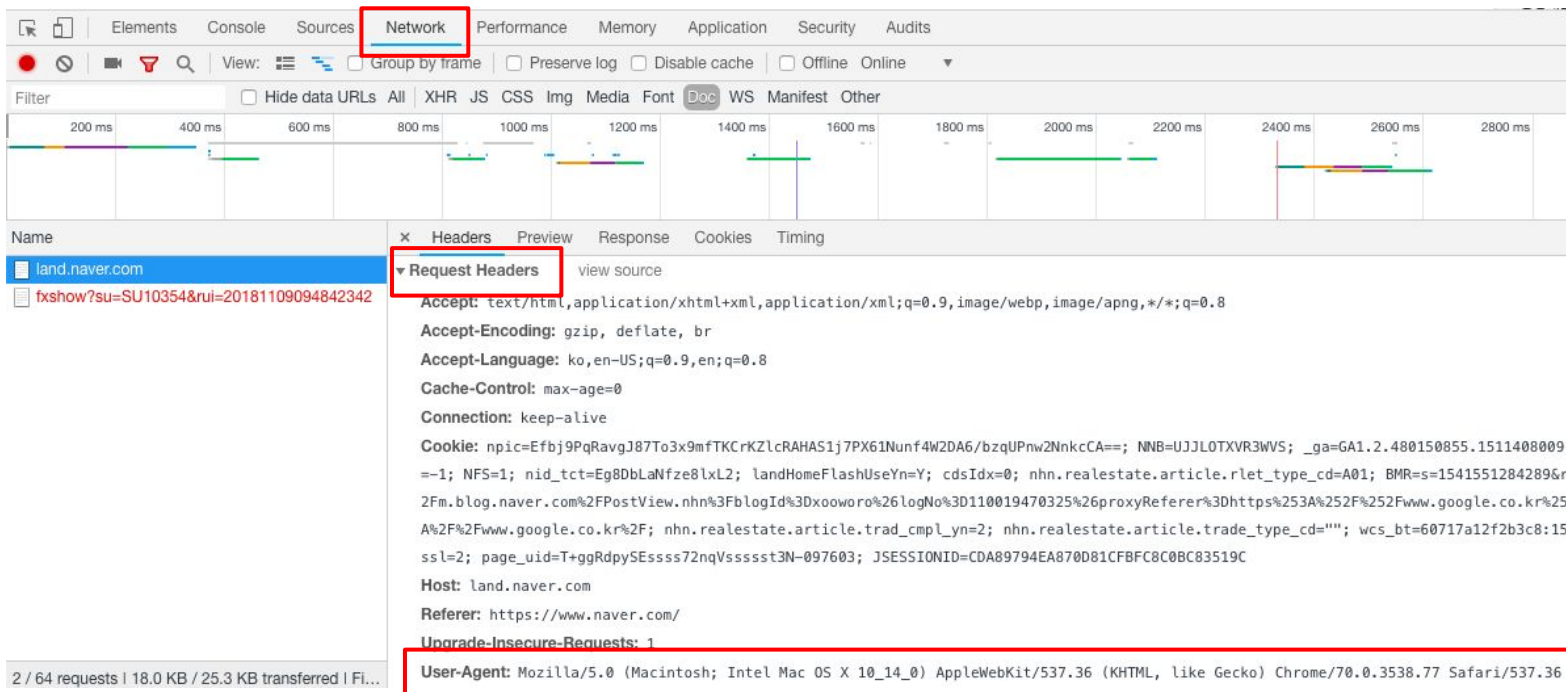
**[시연] 네이버 부동산 매물 리스트 수집**

# User-agent의 이해

- 인터넷 사용자가 웹 서버로 HTTP 요청을 하는 시점에서 클라이언트 웹 브라우저는 HTTP 요청 메시지 중 헤더에 **'사용자 에이전트(User-agent)'를 함께 전송**합니다.
  - GET() 또는 POST() 함수에 User-agent를 추가하지 않으면 기본값을 전송합니다.
  - 기본값 : libcurl/7.54.0 r-curl/3.2 htrr/1.3.1
- 사용자 에이전트는 애플리케이션 유형, 운영 체제, 웹 브라우저 소프트웨어 등 클라이언트에 관한 정보를 담은 문자열이며, 웹 서버는 이를 통해 클라이언트를 식별합니다. 네이버의 일부 서비스는 사용자 에이전트를 보고 4xx 응답을 하는 경우가 있습니다. (부동산, 블로그 등)

# User-agent를 확인하는 방법 1

- 크롬 개발자도구의 Network 탭에서 Request headers를 확인합니다.



The screenshot shows the Chrome DevTools Network tab. The 'Network' tab is selected and highlighted with a red box. Below the network timeline, a request to 'land.naver.com' is selected and highlighted in blue. The 'Request Headers' tab is open, also highlighted with a red box, showing the following headers:

```
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,en-US;q=0.9,en;q=0.8
Cache-Control: max-age=0
Connection: keep-alive
Cookie: npic=Efbj9PqRavgJ87To3x9mfTKrKZlCRAHAS1j7PX61Nunf4W2DA6/bzqUPnw2NnkcA==; NNB=UJJLOTXVR3WVS; _ga=GA1.2.480150855.1511408009;
=-1; NFS=1; nid_tct=Eg8DbLaNfze8lxL2; landHomeFlashUseYn=Y; cdsIdx=0; nhn.realestate.article.rlet_type_cd=A01; BMR=s=1541551284289&r
2Fm.blog.naver.com%2FPostView.nhn%3FblogId%3Dxooworo%26logNo%3D110019470325%26proxyReferer%3Dhttps%253A%252F%252Fwww.google.co.kr%25
A%2F%2Fwww.google.co.kr%2F; nhn.realestate.article.trad_cml_yn=2; nhn.realestate.article.trade_type_cd=""; wcs_bt=60717a12f2b3c8:15
ssl=2; page_uid=T+ggRdpySEsss72nqvsssst3N-097603; JSESSIONID=CDA89794EA870D81CFBFC8C08C83519C
Host: land.naver.com
Referer: https://www.naver.com/
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.77 Safari/537.36
```

At the bottom of the screenshot, the status bar shows '2 / 64 requests | 18.0 KB / 25.3 KB transferred | FI...'. The 'User-Agent' header value is highlighted with a red box.

# User-agent를 확인하는 방법 2

- 관련 웹 사이트 (WhoisHostingThis) 방문

- <https://www.whoishostingthis.com/tools/user-agent/>
- 사용자의 OS 버전
- 애플리케이션 정보
- 웹 브라우저 정보
- IP 주소 정보 등

- Google의 크롤러 사용자 에이전트 보기

- <https://support.google.com/webmasters/answer/1061943?hl=ko>



The screenshot shows the homepage of WhoIsHostingThis. The header includes the site logo, the text "11,799 Real User Reviews Of The World's Biggest Web Hosting Brands", and a notification "101 people on the site right now". A navigation bar contains links for "Best of 2018", "Host Reviews", "Hosting Comparison", "Coupons", "Resources", and "Blog". The main content area features a section titled "User Agent: Find Out Your Web Browser's UA Here". Below this, it displays "Your User Agent is:" followed by a text box containing the user agent string: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.77 Safari/537.36". Below that, it displays "Your IP Address is:" followed by a text box containing the IP address: "39.7.55.165".



## R에서 HTTP 응답 객체의 User-agent를 확인하는 방법

- HTTP 요청 결과를 `res`에 할당하면, `res`는 10개의 원소를 갖는 리스트가 됩니다.
- User-agent는 `request`라는 원소에 포함되어 있습니다. 따라서 다음과 같은 라인을 실행하면 HTTP 요청에 사용된 User-agent를 확인할 수 있습니다.

```
> print(x = res$request$options$useragent)
```

```
[1] "libcurl/7.54.0 r-curl/3.2 httr/1.3.1" [기본 User-agent]
```

```
[1] "사용자가 설정한 User-agent 문자열" [사용자가 설정한 경우]
```

## R에서 User-agent를 설정하는 방법

- 사용하려는 User-agent 문자열을 아래와 같이 `ua` 객체에 할당합니다.

```
> ua <- 'User-agent 문자열'
```

- `user_agent()` 함수의 `agent` 인자에 `ua`를 할당한 것을 `GET()` 또는 `POST()` 함수에 추가합니다.

```
> res <- GET(url = '네이버 부동산 URL',  
             query = list(query strings),  
             user_agent(agent = ua) )
```

# 네이버 부동산 웹 페이지 설정 1

- 네이버 부동산 메인페이지에서 '서울시', '강남구', '삼성동'을 선택합니다.

The screenshot shows the Naver Real Estate website interface. At the top, there is a green header with the 'NAVER 부동산' logo and a search bar. Below the header, there are navigation tabs for '부동산 홈', '매물 Beta', '매물', '분양', '뉴스', '커뮤니티', '경매', '테마부동산', and 'MY페이지'. A '부동산관리센터' button is located on the right side of the navigation bar.

The main content area features a map of a city district. Above the map, there are location selection filters: '서울시' (Seoul), '강남구' (Gangnam-gu), '삼성동' (Samsung-dong), and '단지' (Complex). These filters are highlighted with a red box. To the right of the map, there is a list of nearby real estate listings. The list includes the following items:

단지	현장확인매물	중개업소	매매
가나다순	인기순	세대수순	입주순
SM드림빌	(0건)		
강변삼부	(4건)		
그라나다	(0건)		
네오블로섬	(0건)		
노블테라스	(3건)		
다우림하우스	(0건)		
동구썬타빌	(1건)		
동일파크스위트	(14건)		
레이안삼성1차(101동)	(0건)		
레이안삼성1차(201동)	(0건)		

## 네이버 부동산 웹 페이지 설정 2

- 지도 아래에 있는 조건 설정 메뉴에서 설정을 바꿔봅니다. 이 때 웹 브라우저의 주소창에서 URI가 어떻게 바뀌는지 유심히 관찰하시기 바랍니다.

[삼성동]

매물	시세	학군정보	관리비
----	----	------	-----

거래종류	<input type="radio"/> 전체	<input checked="" type="radio"/> 매매	<input type="radio"/> 전세	<input type="radio"/> 월세	<input type="radio"/> 단기임대
매물종류	<input checked="" type="checkbox"/> 아파트	<input checked="" type="checkbox"/> 주상복합	<input checked="" type="checkbox"/> 재건축		

가격	전체	~	만원	면적	전체	~	m <sup>2</sup>	
역세권	-	이내	입주일	전체	이내	세대수	전체	이상
평균관리비	전체							

[초기화](#) [조회하기](#)

이 부분을 하나 하나 바꾸고 조회하기를 클릭하면서 주소창의 URI를 관찰합니다.

# 네이버 부동산 웹 페이지 설정 3

- 아파트 단지 목록에서 '아이파크삼성'을 선택하면 아래와 같이 매물 목록이 뜹니다.

확인매물  거래완료 매물 포함       랭킹순 | 확인매물일자순 ▾ | 면적순 ▾ | 가격순 ▾

거래	확인일자	매물명	면적(m <sup>2</sup> )	동	층	매물가(만원)	연락처
매매		 <a href="#">아이파크삼성</a>    정원조망 내부고급아파트럭셔리 최상급...	182/145	사우스윙 동	3/46	400,000 <a href="#">부동산뱅크</a>	아이파크공인중개... 02-518-8899
매매		 <a href="#">아이파크삼성</a>    한강과 도심뷰를 한번에, 로얄층으로 깨끗...	182/145	이스트윙 동	중/38	350,000 <a href="#">부동산뱅크</a>	라움부동산중개... 02-541-8898
매매		<a href="#">아이파크삼성</a>    입주가능한 실매물,전망좋은 집,내부깨끗	182/145	사우스윙 동	22/46	340,000 <a href="#">부동산뱅크</a>	애플공인중개사사... 02-515-2989
매매		<a href="#">아이파크삼성</a>    방과 거실 모두 트인 전망 좋은 집	195/156	웨스트윙 동	28/45	400,000 <a href="#">부동산뱅크</a>	애플공인중개사사... 02-515-2989
매매		<a href="#">아이파크삼성</a>    로얄층, 한강전망, 내부깨끗, 입주가능	216/175	웨스트윙 동	29/45	540,000 <a href="#">부동산뱅크</a>	애플공인중개사사... 02-515-2989

이번 예제에서 수집할 데이터

- 거래
- 확인일자
- 매물명
- 면적
- 동
- 층
- 매물가(만원)
- 연락처 (부동산, 전화번호)
- 매물 정보 텍스트

혼자 공부할 때 참고자료

# 온라인에 공개된 사이트

- 개인 블로그 (무료)
  - <https://stat-and-news-by-daragon9.tistory.com/category/R/R%ED%81%AC%EB%A1%A4%EB%A7%81>
  - <https://kuduz.tistory.com/1041>
- 개인 유튜브 (무료)
  - [https://www.youtube.com/results?search\\_query=r%EC%BD%94%ED%99%80%EB%A6%AD](https://www.youtube.com/results?search_query=r%EC%BD%94%ED%99%80%EB%A6%AD)
- 온라인 유료 강의
  - <https://mindscale.kr/course/r-crawling>

# 도전 과제

1. 다음 포털에서 실시간 검색어 수집
  - a. URL : <https://www.daum.net/>
2. 네이버 증권 국내증시 KOSPI 페이지에서 상단에 있는 표 수집
  - a. URL : [https://finance.naver.com/sise/sise\\_index.nhn?code=KOSPI](https://finance.naver.com/sise/sise_index.nhn?code=KOSPI)
  - b. <table> 태그를 찾고, html\_table() 함수를 이용하면 해결 가능
  - c. Windows 사용자는 locale을 'C'로 변경해야 함



**End of Document**