

데이터 수집

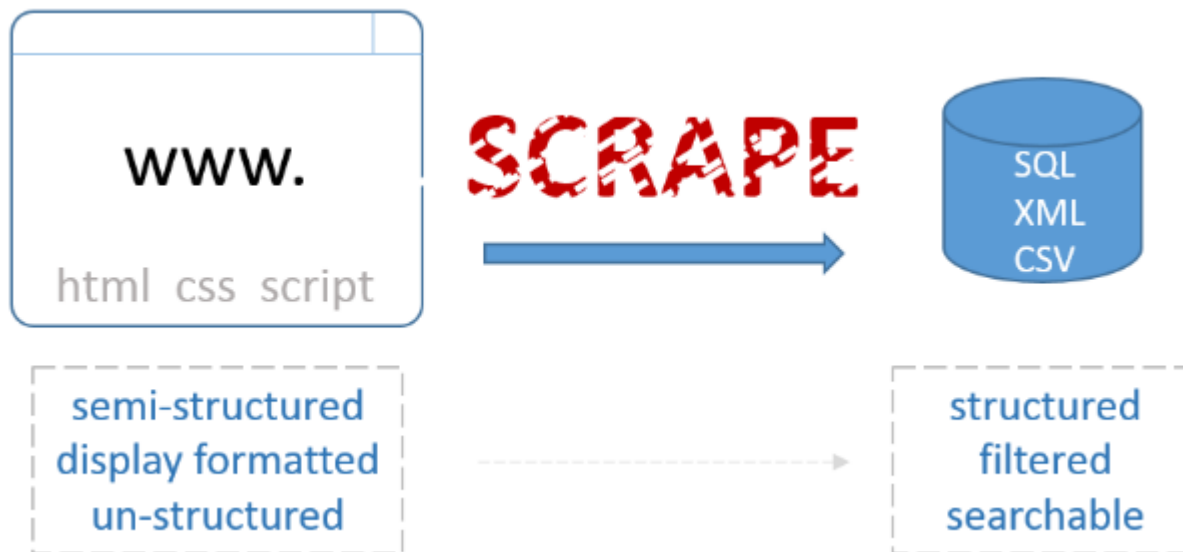
1 정적 스크래핑(크롤링)

[웹 스크래핑(web scraping)]

웹 사이트 상에서 원하는 부분에 위치한 정보를 컴퓨터로 하여금 자동으로 추출하여 수집하는 기술

[웹 크롤링(web crawling)]

자동화 봇(bot)인 웹 크롤러가 정해진 규칙에 따라 복수 개의 웹 페이지를 브라우징 하는 행위



데이터 수집

1 정적 스크래핑(크롤링)

Selectors

Basics

#id
element
.class,
.class.class
*
selector1,
selector2

Hierarchy

ancestor
descendant
parent > child
prev + next
prev ~ siblings

Basic Filters

:first
:last
:not(selector)
:even
:odd
:eq(index)
:gt(index)
:lt(index)

Content Filters

:contains(text)
:empty
:has(selector)
:parent

Visibility Filters

:hidden
:visible

Child Filters

:nth-child(expr)
:first-child
:last-child
:only-child

Attribute Filters

[attribute]
[attribute=value]
[attribute!=value]
[attribute^=value]
[attribute\$=value]
[attribute*=value]
[attribute|=value]
[attribute~=value]
[attribute]
[attribute2]

Forms

:input
:text
:password
:radio
:checkbox
:submit
:image
:reset
:button
:file

Form Filters

:enabled
:disabled
:checked
:selected

데이터 수집

1 정적 스크래핑(크롤링)

(1) 네이버 영화 사이트 댓글정보 스크래핑

네이버 영화 사이트의 데이터 중 영화제목, 평점, 리뷰만을 추출하여 CSV 파일의 정형화된 형식으로 저장한다.

(1) 스크래핑하려는 웹페이지의 URL 구조와 문서 구조를 파악해야 한다.

- URL 구조 : <http://movie.naver.com/movie/point/af/list.nhn?page=1>

The screenshot shows the Naver movie review page for the movie '고지전' (Gojijeon). The URL in the browser's address bar is <http://movie.naver.com/movie/point/af/list.nhn?page=1>, with the page number '1' highlighted in a red box. The page displays a list of reviews, including the title, rating, and a brief review snippet. The DOM structure is visible on the right, showing the HTML elements for the review list. A red box highlights the specific review for '고지전', showing the rating '10' and the title '고지전'.

번호	평점	140자평
14911148	★★★★★ 10	고지전 박평식 평론가의 성을 차는 영화는 어디 있는가? 내가 가졌으면, 평론가가 된다면, 박평식 평론가부터 지적할 이다 신고
14911147	★★★★★ 10	국가부도의 날 IMF는 현재진행형 뉴배우들의 연기가 뛰어나서당시를 대로 옮겨놓은 느낌이었다. 신고
14911146	★★★★★ 9	후드 액션씬 화려하고 좋음. 여주가 넘 이쁨. 신선하고 재밌음. 2편도 기대된다. 신고
14911145	★★★★★ 8	신비한 동물들과 그린델왈드의 범죄 조니덱의 비중이 1편보다 확실히 커졌다. 신고
14911144	★★★★★ 7	1987

1 정적 스크래핑(크롤링)

- 문서 구조

영화 제목 class="movie"

영화 평점 class="point"

영화 리뷰 class="title"

[rvest 패키지의 주요 함수]

html_nodes(x, css, xpath), html_node(x, css, xpath)

html_text(x, trim=FALSE)

html_attrs(x)

html_attr(x, name, default = "")

[1페이지 스크래핑]

```
install.packages("rvest");  
library(rvest)  
url <- "http://movie.naver.com/movie/point/af/list.nhn?page=1"  
text <- read_html(url, encoding="CP949")  
# 영화제목  
nodes <- html_nodes(text, ".movie")  
title <- html_text(nodes)  
# 영화평점  
nodes <- html_nodes(text, ".point")  
point <- html_text(nodes)  
# 영화리뷰  
nodes <- html_nodes(text, ".title")  
review <- html_text(nodes, trim=TRUE); review  
review <- gsub("\t", "", review)  
review <- gsub("\r\n", "", review)  
review <- gsub("신고", "", review); review  
page <- cbind(title, point)  
page <- cbind(page, review)  
write.csv(page, "movie_reviews.csv")
```

1 정적 스크래핑(크롤링)

[여러 페이지 스크래핑]

```
site<- "http://movie.naver.com/movie/point/af/list.nhn?page="
movie.review <- NULL
for(i in 1: 100) {
  url <- paste(site, i, sep="")
  text <- read_html(url, encoding="CP949")
  nodes <- html_nodes(text, ".movie")
  title <- html_text(nodes)
  nodes <- html_nodes(text, ".point")
  point <- html_text(nodes)
  nodes <- html_nodes(text, ".title")
  review <- html_text(nodes, trim=TRUE)
  review <- gsub("http", "", review); review <- gsub("http", "", review)
  review <- gsub("신고", "", review)
  page <- cbind(title, point)
  page <- cbind(page, review)
  movie.review <- rbind(movie.review, page)
}
write.csv(movie.review, "movie_reviews2.csv")
```

데이터 수집

1 정적 스크래핑(크롤링)

2. 한국일보 헤드라인 기사 스크래핑

→ ↻ ⓘ 주의 요함 | www.hankookilbo.com

고 보는 동영상 PRAN

정치 경제 사회 국제 문화 연예 라이프 스포츠 피플 지역 | 오피니언 기획·특집 디지털스페셜 멀티미디어

"문 대통령, 링컨이나 물태우냐" 정동영, 선거제 개편 촉구

바른미래당 · 민주평화당 · 정의당 등 3당이 연동형 비례대표제 관철을 위해 문재인 대통령을 거론하며 더불어민주당에 대한 압박 수위를 높였다. 더보기

간판 없는 비밀의 공간 "취향을 팝니다"
요즘 뜨는 명소들은 손맛이나 목 좋은 곳이 아니다. 찾아오는 방법부터 공간이 가진 매력을 느낄 수 있어야 비로소 명소가 된다.

겨울 "신지도 않는 에어조던을 왜 수집하느냐고?"
대중문화부터 상품까지, 레트로에 열광하는 이유는
미국서도 멀레니얼 세대 겨냥 레트로 마케팅 활발

"그때 그 아이 맞습니다" 아역출신 배우들이 떴다
"여기 착하는 20대 배우" 기괴한 아역 출신 배우들의 활약이 반세기

단독 이영렬, 돈봉투 만찬 '무혐의' 받았지만 명예는...
형사상 혐의 벗었지만 정권초 과도한 망신주기 조치 비판도

뒤끝뉴스 '4조원'에 막힌 470조 슈퍼 예산심의
문화상 의장, 중부세 인상 등 예산부수법안 28건 지정

조명래 "미세먼지 중국 탓 하기 전 우리가 줄여야"
30일 또 스모그 밀려온다... "中 환경규제 늦춘 탓"
흡입하면 몸에 축적되는 미세먼지, 이렇게 대처해야

속보 부산 폐수처리업체 황화수소 누출...4명 의식불명
대처, 아이스크림 개발자로 었지페 모델 후보에

이번엔 '분빠이'... 이은재 의원의 일본어 사랑
맞춤법은 틀리면서... 이은재 의원, 또 일어 사용

줌인뉴스 배달음식 하나 당 딸려오는 일회용품 7개
유치원생이 아파트 2채... '금수저' 미성년자들 조사

"대기업 임원들, 2년 차에 옷 가장 많이 벗는다"
친일했는데 독립운동가? "가짜 100명 색출한다"

음주 뺑소니로 50대 사망 이르게 한 20대
"광개이بل 화재 처음 본다" KT화재 원인 미스터리

합계출산율 2.3분기 연속 '0명대'... 역대 최저
수면유도제 졸피뎀 처방, 4주 넘길 수 없다

김관영 "여당 스스로 대통령 레임덕 부추겨"

데이터 수집

1 정적 스크래핑(크롤링)

[XML 패키지의 주요 함수]

htmlParse (file, encoding="...")

xpathSApply(doc, path, fun)

fun : **xmlValue**, **xmlGetAttr**, **xmlAttrs**

library(XML)

imsi <- read_html("http://hankookilbo.com")

t <- htmlParse(imsi)

content <- xpathSApply(t, "//p[@class='title']", xmlValue);

content

content <- gsub("[[:punct:]][:cntrl:]", "", content)

content

content <- trimws(content)

content

```
<div class="splash" /.../div>
▼<ul class="headline-related">
  ▼<li>
    ▶<div class="frame"> </div>
    ▼<p class="title">
      <a href="/News/Read/201811262370046811?
      did=PA&dtype=3&dtypecode=571" target>

      간판 없는 비밀의 공간 “취향을 팝니다”
    </a>
    </p>
    ▶<p class="preview">...</p>
  </li>
  ▶<li>...</li>
  ▶<li>...</li>
```

1 정적 스크래핑(크롤링)

그 외의 웹 스크래핑시 알고 있으면 도움되는 내용들

[R에서 GET으로 사이트 내용 가져오기 : httr 패키지 사용]

```
library(httr)
```

```
http.standard <- GET('http://www.w3.org/Protocols/rfc2616/rfc2616.html')
```

```
title2 = html_nodes(read_html(http.standard), 'div.toc h2')
```

```
title2 = html_text(title2)
```

```
title2
```

[R에서 POST로 사이트 내용 가져오기 : httr 패키지 사용]

```
library(httr)
```

```
# POST 함수를 이용해 모바일 게임 랭킹 10월 29일 주 모바일 게임 랭킹을 찾는다
```

```
 #(http://www.gevolution.co.kr/score/gamescore.asp?t=3&m=0&d=week)
```

```
game = POST('http://www.gevolution.co.kr/score/gamescore.asp?t=3&m=0&d=week',
```

```
          encode = 'form', body=list(txtPeriodW = '2019-04-10'))
```

```
title2 = html_nodes(read_html(game), 'a.tracktitle')
```

```
title2 = html_text(title2)
```

```
title2[1:10]
```


1 정적 스크래핑(크롤링)

[뉴스, 게시판 등 글 목록에서 글의 URL만 뽑아내기]

```
res = GET('https://news.naver.com/main/list.nhn?mode=LSD&mid=sec&sid1=001')
htxt = read_html(res)
link = html_nodes(htxt, 'div.list_body a')
article.href = unique(html_attr(link, 'href'))
```

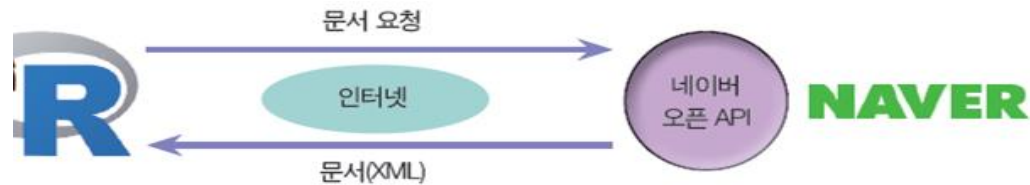
[이미지, 첨부파일 다운 받기]

```
# pdf
res = GET('http://cran.r-project.org/web/packages/htr/htr.pdf')
writeBin(content(res, 'raw'), 'c:/Temp/htr.pdf')

# jpg
h = read_html('http://unico201dothome.co.kr/productlog.html')
imgs = html_nodes(h, 'img')
img.src = html_attr(imgs, 'src')
for(i in 1:length(img.src)){
  res = GET(paste('http://unico201dothome.co.kr/',img.src[i], sep=""))
  writeBin(content(res, 'raw'), paste('c:/Temp/', img.src[i], sep=""))
}
```

데이터 수집

2 네이버의 뉴스와 블로그 글 읽어오기



<https://developers.naver.com/docs/search/blog/> 에서 내용 검토

API	요청	출력 포맷
뉴스	https://openapi.naver.com/v1/search/news.xml	XML
블로그	https://openapi.naver.com/v1/search/blog.xml	XML

요청 변수	값	설명
query	문자(필수)	검색을 원하는 질의, UTF-8 인코딩
display	정수: 기본값 10, 최대 100	검색 결과의 출력 건수(최대 100까지 가능)
start	정수: 기본값 1, 최대 1000	검색의 시작 위치(최대 1000까지 가능)
sort	문자: date(기본값), sim	정렬 옵션 • date : 날짜순(기본값) • sim : 유사도순

2 네이버의 뉴스와 블로그 글 읽어오기

네이버 블로그에서 "여름추천요리"를 검색하여 블로그에 올려진 데이터를 수집해본다.

[네이버 블로그 연동 : Rcurl 패키지 사용]

```
install.packages("RCurl")
```

```
library(RCurl)
```

```
library(XML)
```

```
searchUrl<- "https://openapi.naver.com/v1/search/blog.xml"
```

```
Client_ID <- "....."
```

```
Client_Secret <- "....."
```

```
query <- URLencode(iconv("여름추천요리","euc-kr","UTF-8"))
```

```
url<- paste(searchUrl, "?query=", query, "&display=20", sep="")
```

```
doc<- getURL(url, httpheader = c('Content-Type' = "application/xml",  
                                'X-Naver-Client-Id' = Client_ID,'X-Naver-Client-Secret' = Client_Secret))
```

```
# 블로그 내용에 대한 리스트 만들기
```

```
doc2 <- htmlParse(doc, encoding="UTF-8")
```

```
text<- xpathSApply(doc2, "//item/description", xmlValue);
```

```
text
```

2 네이버의 뉴스와 블로그 글 읽어오기

네이버 뉴스에서 "미세먼지"로 검색하여 뉴스에 올려진 데이터를 수집해본다.

[네이버 블로그 연동 : Rcurl 패키지 사용]

```
install.packages("RCurl")
```

```
library(RCurl)
```

```
library(XML)
```

```
searchUrl<- "https://openapi.naver.com/v1/search/news.xml"
```

```
Client_ID <- "izGsQP2exeThwwEUUVU3x"
```

```
Client_Secret <- "WrwbQ1l6ZI"
```

```
query <- URLencode(iconv("미세먼지","euc-kr","UTF-8"))
```

```
url<- paste(searchUrl, "?query=", query, "&display=20", sep="")
```

```
doc<- getURL(url, httpheader = c('Content-Type' = "application/xml",  
                                'X-Naver-Client-Id' = Client_ID,'X-Naver-Client-Secret' = Client_Secret))
```

```
# 블로그 내용에 대한 리스트 만들기
```

```
doc2 <- htmlParse(doc, encoding="UTF-8")
```

```
text<- xpathSApply(doc2, "//item/description", xmlValue);
```

```
text
```

3 트위터 글 읽어오기

트위터에서는 twitterR 이라는 패키지를 제공하여 트위터에 올려진 글을 수집하는데 도움을 준다.

```
install.packages("twitterR")
library(twitterR)
api_key <- "....."
api_secret <- "....."
access_token <- "....."
access_token_secret <- "....."
setup_twitter_oauth(api_key,api_secret, access_token,access_token_secret)
key <- "취업"
key <- enc2utf8(key)
result <- searchTwitter(key, n=100)
DF <- twListToDF(result)
content <- DF$text
content <- gsub("[[:lower:][:upper:][:digit:][:punct:][:cntrl:]]", "", content);
content
```

setup_twitter_oauth(api_key,api_secret , access_token,access_token_secret)	현재의 R세션에 인증키를 내려받는 기능
result<- searchTwitter(key, n=100)	key 에 해당되는 트위터 글 읽어 오기
DF <- twListToDF(result)	응답 내용을 데이터 프레임으로 변환

4 동적 스크래핑(크롤링)

다음의 경우에는 웹 페이지의 내용이 동적으로 생성되는 경우에는 지금까지의 방법으로 스크래핑 할 수 없다.

- 사용자의 선택과 같은 이벤트에 의해서 자바스크립트의 수행 결과로 콘텐츠를 생성한다.
- 페이지의 렌더링을 끝낸 후에 Ajax 기술을 이용하여 서버로 부터 콘텐츠의 일부를 전송받아 동적으로 구성한다.

이러한 경우에는 Selenium 을 사용하면 제어되는 브라우저에 페이지를 렌더링 해놓고 렌더링된 결과에서 콘텐츠를 읽어올 수 있다. 뿐만 아니라 콘텐츠내에서 클릭이벤트를 발생할 수도 있으며 로그인과 같은 데이터를 입력하는 것도 가능하다.

[Selenium 서버 기동 과정]

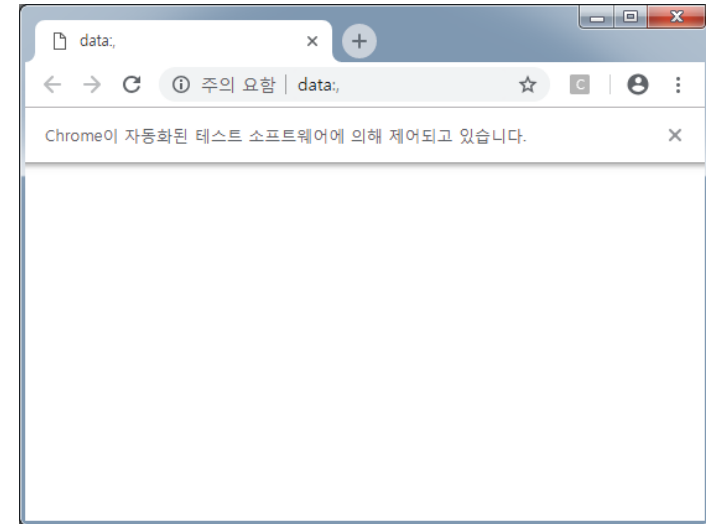
- (1) selenium-server-standalone-master.zip, chromedriver.exe 를 복사한다.
- (2) 적당한 디렉토리에 selenium-server-standalone-master.zip 파일의 압축을 푼다
- (3) bin 디렉토리 안에 chromedriver.exe 를 복사한다.
- (4) Selenium 을 기동시킨다. (박스속의 명령을 CMD 창에서 실행시켜야 한다.)

```
java -jar selenium-server-standalone.jar -port 4445
```

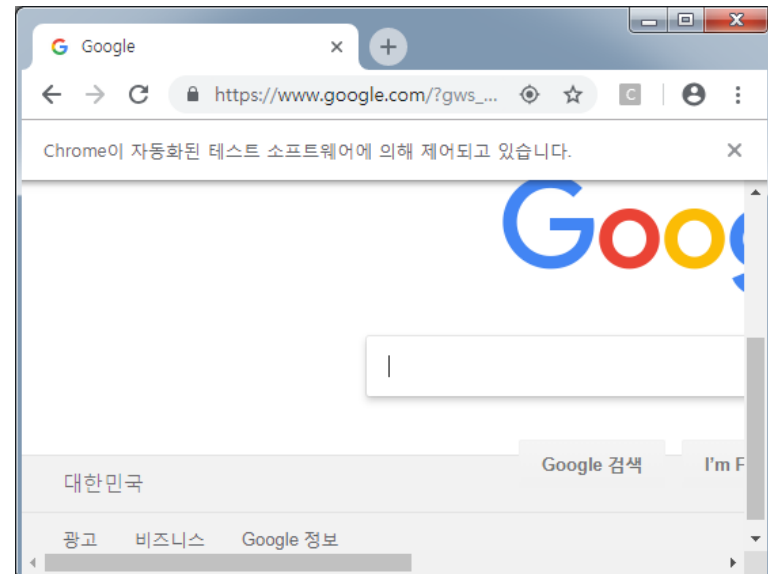
데이터 수집

4 동적 스크래핑(크롤링)

```
C:\WRstudy\Wsssm\bin>java -jar selenium-server-standalone.jar -port 4445
17:30:43.319 INFO - Selenium build info: version: '3.4.0', revision: 'unknown'
17:30:43.320 INFO - Launching a standalone Selenium Server
2018-11-28 17:30:43.386:INFO::main: Logging initialized @394ms to org.selenium
.jetty9.util.log.StdErrLog
17:30:43.449 INFO - Driver class not found: com.opera.core.systems.OperaDrive
```



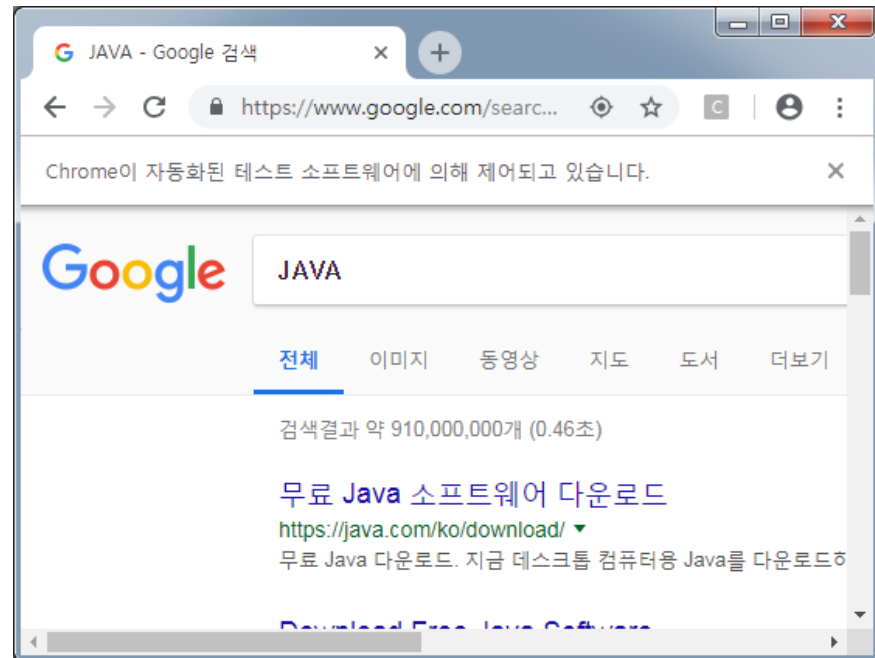
```
install.packages("RSelenium")
library(RSelenium)
remDr <- remoteDriver(remoteServerAddr = "localhost" ,
                      port = 4445, browserName = "chrome")
remDr$open()
remDr$navigate("http://www.google.com/")
```



데이터 수집

4 동적 스크래핑(크롤링)

```
webElem <- remDr$findElement(using = "css", "[name = 'q']")  
webElem$sendKeysToElement(list("JAVA", key = "enter"))
```



4 동적 스크래핑(크롤링)

[API 소개]

remDr <- remoteDriver(remoteServerAddr="localhost",port=4445,browserName="chrome")	Selenium 서버에 접속하고 remoteDriver 객체 리턴
remDr\$open()	크롬브라우저 오픈
remDr\$navigate(url)	페이지 렌더링
doms <- remDr\$findElements(using ="css selector","컨텐츠를추출하려는태그의선택자")	태그를 찾자
sapply(doms,function(x){x\$getText()})	찾아진 태그들의 컨텐츠 추출
more <- remDr\$findElements(using='css selector', 클릭이벤트를강제발생시키려는태그의선택자')	태그를 찾자
sapply(more,function(x){x\$clickElement()})	찾아진 태그에 클릭 이벤트 발생
webElem <- remDr\$findElement("css", "body") remDr\$executeScript("scrollTo(0, document.body.scrollHeight)", args = list(webElem))	페이지를 아래로 내리는(스크롤) 효과

4 동적 스크래핑(크롤링)

1. 자바스크립트가 자동 생성하는 댓글 읽어 오기 : 신라스테이 호텔

```
remDr$navigate("https://www.agoda.com/ko-kr/shilla-stay-seocho/hotel/seoul-kr.html?cid=-204")
doms <- remDr$findElements(using = "css selector", ".Review-comment-bodyText")
sapply(doms, function (x) {x$getElementText()})
```

2. 링크 클릭으로 AJAX 로 처리되는 네이버 웹툰 댓글 읽어 오기

```
repl_v = NULL;
url<-'http://comic.naver.com/comment/comment.nhn?titleId=570503&no=135'
remDr$navigate(url)
doms1<-remDr$findElements(using ="css selector","ul.u_cbox_list span.u_cbox_contents")
p_repl <- sapply(doms1,function(x){x$getElementText()})
p_repl_v <- unlist(p_repl)
more<-remDr$findElements(using='css','span.u_cbox_in_view_comment')
sapply(more,function(x){x$clickElement()})
doms2<-remDr$findElements(using ="css selector","ul.u_cbox_list span.u_cbox_contents")
repl <-sapply(doms2,function(x){x$getElementText()})
repl_v <- c(repl_v, unlist(repl))
```

4 동적 스크래핑(크롤링)

```
repeat {  
  for (i in 4:12) {  
    nextCss <- paste("#cbox_module>div>div.u_cbox_paginate>div> a:nth-child(",i,") > span", sep="")  
    try(nextPage<-remDr$findElements(using='css',nextCss))  
    if(length(nextPage) == 0) break;  
    sapply(nextPage,function(x){x$clickElement()})  
    Sys.sleep(1)  
    doms3<-remDr$findElements(using ="css selector","ul.u_cbox_list span.u_cbox_contents")  
    repl <-sapply(doms3,function(x){x$getText()})  
    repl_v <- c(repl_v, unlist(repl))  
  }  
}
```

4 동적 스크래핑(크롤링)

```
try(nextPage<-remDr$findElements(using='css',
    "#cbox_module > div > div.u_cbox_paginate > div > a:nth-child(13) > span.u_cbox_cnt_page"))
if(length(nextPage) == 0) break;
sapply(nextPage,function(x){x$clickElement()})
Sys.sleep(1)
doms2<-remDr$findElements(using ="css selector","ul.u_cbox_list span.u_cbox_contents")
repl <-sapply(doms2,function(x){x$getText()})
repl_v <- c(repl_v, unlist(repl))
}
print(repl_v)
write(repl_v, "webtoon.txt")
```

4 동적 스크래핑(크롤링)

```
library(rvest)
url <-
"http://www.saramin.co.kr/zf_user/search?search_area=main&search_done=y&search_optional_item=n&searchType
=recently&searchword=java"
remDr$navigate(url)
nextten<-remDr$findElements(using='css',' #recruit_info_list > div > a.next.page_move')
page <- 1
flag <- FALSE
fullcontent<-NULL
repeat {
  for(index in 3 : 11){
    fullContent<-remDr$findElements(using='css', '#recruit_info_list p.keywordline')
    content <- sapply(fullContent,function(x){x$getElementText()})
    content <- gsub("키워드","",content)
    fullcontent <- c(fullcontent,unlist(content))
  }
}
```

4 동적 스크래핑(크롤링)

```
if(page==1){
  index <- index-1
}
nextx <- paste('#recruit_info_list > div > a:nth-child(',index,')', sep='')
nextlink <- remDr$findElements(using='css',nextx)
if(length(nextlink)==0){
  flag <- TRUE
}
sapply(nextlink,function(x){x$clickElement()})
Sys.sleep(3)
}
nextten <- remDr$findElements(using='css',' #recruit_info_list > div > a.next.page_move')
if(length(nextten)==0 || flag) break
sapply(nextten,function(x){x$clickElement()})
Sys.sleep(3)
page <- page+10
}
write(fullcontent,"examall.txt")
```

데이터 수집

[정규표현식 활용]

```
word <- "JAVA javascript Aa 가나다 AAaAaA123 %^&*"

```

```
gsub("A", "", word)

```

```
gsub("a", "", word)

```

```
gsub("Aa", "", word)

```

```
gsub("(Aa){2}", "", word)

```

```
gsub("[Aa]", "", word)

```

```
gsub("[가-힣]", "", word)

```

```
gsub("[^가-힣]", "", word)

```

```
gsub("[&^%*]", "", word)

```

```
gsub("[[:punct:]]", "", word)

```

```
gsub("[[:alnum:]]", "", word)

```

```
gsub("[1234567890]", "", word)

```

```
gsub("[[:digit:]]", "", word)

```

```
gsub("[^[:alnum:]]", "", word)

```

```
gsub("[[:space:]]", "", word)

```

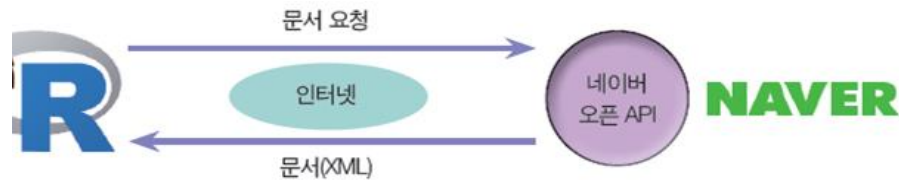
.	Any character except \n
	Or, e.g. (a b)
[...]	List permitted characters, e.g. [abc]
[a-z]	Specify character ranges
[^...]	List excluded characters
(...)	Grouping, enables back referencing using \N where N is an integer

*	Matches at least 0 times
+	Matches at least 1 time
?	Matches at most 1 time; optional string
{n}	Matches exactly n times
{n,}	Matches at least n times
{,n}	Matches at most n times
{n,m}	Matches between n and m times

[[[:digit:]] or \d	Digits; [0-9]
\\D	Non-digits; [^0-9]
[[[:lower:]]	Lower-case letters; [a-z]
[[[:upper:]]	Upper-case letters; [A-Z]
[[[:alpha:]]	Alphabetic characters; [A-z]
[[[:alnum:]]	Alphanumeric characters [A-z0-9]
\\w	Word characters; [A-z0-9_]
\\W	Non-word characters
[[[:xdigit:]] or \x	Hexadec. digits; [0-9A-Fa-f]
[[[:blank:]]	Space and tab
[[[:space:]] or \s	Space, tab, vertical tab, newline, form feed, carriage return
\\S	Not space; [^[:space:]]
[[[:punct:]]	Punctuation characters; !"#\$%&'()*+,-./:;<=>?@[]^_`{ }~
[[[:graph:]]	Graphical char.; [[[:alnum:]][[:punct:]]
[[[:print:]]	Printable characters; [[[:alnum:]][[:punct:]]\s]
[[[:cntrl:]] or \c	Control characters; \n, \r etc.

데이터 수집

SNS 수집 : 네이버의 뉴스와 블로그 글 읽어오기



<https://developers.naver.com/docs/search/blog/> 에서 내용 검토

API	요청	출력 포맷
뉴스	https://openapi.naver.com/v1/search/news.xml	XML
블로그	https://openapi.naver.com/v1/search/blog.xml	XML

요청 변수	값	설명
query	문자(필수)	검색을 원하는 질의, UTF-8 인코딩
display	정수: 기본값 10, 최대 100	검색 결과의 출력 건수(최대 100까지 가능)
start	정수: 기본값 1, 최대 1000	검색의 시작 위치(최대 1000까지 가능)
sort	문자: date(기본값), sim	정렬 옵션 <ul style="list-style-type: none">• date : 날짜순(기본값)• sim : 유사도순

데이터 수집

네이버의 뉴스와 블로그 글 읽어오기

네이버 블로그에서 "여름추천요리"를 검색하여 블로그에 올려진 데이터를 수집해본다.

[네이버 블로그 연동 : Rcurl 패키지 사용]

```
install.packages("RCurl")
```

```
library(RCurl)
```

```
library(XML)
```

```
searchUrl<- "https://openapi.naver.com/v1/search/blog.xml"
```

```
Client_ID <- "....."
```

```
Client_Secret <- "....."
```

```
query <- URLencode(iconv("여름추천요리","euc-kr","UTF-8"))
```

```
url<- paste(searchUrl, "?query=", query, "&display=20", sep="")
```

```
doc<- getURL(url, httpheader = c('Content-Type' = "application/xml",  
                                'X-Naver-Client-Id' = Client_ID,'X-Naver-Client-Secret' = Client_Secret))
```

```
# 블로그 내용에 대한 리스트 만들기
```

```
doc2 <- htmlParse(doc, encoding="UTF-8")
```

```
text<- xpathSApply(doc2, "//item/description", xmlValue);
```

```
text
```

데이터 수집

네이버의 뉴스와 블로그 글 읽어오기

네이버 뉴스에서 "미세먼지"로 검색하여 뉴스에 올려진 데이터를 수집해본다.

[네이버 뉴스 연동 : Rcurl 패키지 사용]

```
install.packages("RCurl")
```

```
library(RCurl)
```

```
library(XML)
```

```
searchUrl<- "https://openapi.naver.com/v1/search/news.xml"
```

```
Client_ID <- "izGsQP2exeThwwEUVU3x"
```

```
Client_Secret <- "WrwbQ1l6ZI"
```

```
query <- URLEncode(iconv("미세먼지","euc-kr","UTF-8"))
```

```
url<- paste(searchUrl, "?query=", query, "&display=20", sep="")
```

```
doc<- getURL(url, httpheader = c('Content-Type' = "application/xml",  
                                'X-Naver-Client-Id' = Client_ID,'X-Naver-Client-Secret' = Client_Secret))
```

```
# 블로그 내용에 대한 리스트 만들기
```

```
doc2 <- htmlParse(doc, encoding="UTF-8")
```

```
text<- xpathSApply(doc2, "//item/description", xmlValue);
```

```
text
```

데이터 수집

SNS 수집 : 트위터 글 읽어오기

트위터에서는 twitterR 이라는 패키지를 제공하여 트위터에 올려진 글을 수집하는데 도움을 준다.

```
install.packages("twitterR")
library(twitterR)
api_key <- "....."
api_secret <- "....."
access_token <- "....."
access_token_secret <- "....."
setup_twitter_oauth(api_key,api_secret, access_token,access_token_secret)
key <- "취업"
key <- enc2utf8(key)
result <- searchTwitter(key, n=100)
DF <- twListToDF(result)
content <- DF$text
content <- gsub("[[:lower:][:upper:][:digit:][:punct:][:cntrl:]]", "", content);
content
```

setup_twitter_oauth(api_key,api_secret, access_token,access_token_secret)	현재의 R세션에 인증키를 내려받는 기능
result<- searchTwitter(key, n=100)	key 에 해당되는 트위터 글 읽어 오기
DF <- twListToDF(result)	응답 내용을 데이터 프레임으로 변환