

# 1.1. HTTP

## ■ HTTP Architecture



### ■ 예제



**Client**

*HTTP*

<http://www.naver.com>



**Server**

## ○ Request



request



## ▼ General

**Request URL:** https://www.naver.com/  
**Request Method:** GET  
**Status Code:** ● 200  
**Remote Address:** 23.35.221.113:443  
**Referrer Policy:** origin

## ▼ 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  
**cookie:** npic=79bi60ql0lJkDEszX/3bql0z0I/RKFahZzkp84I+Dv45TGv/A1T4U11FknSwd764CA==; NNB=AFU2GPITRSavs; \_\_date\_rename\_layer=201782; AS\_hfaJm5VzyHz7K6/x4+d; PM\_CK\_rcode=09290610; NID\_SES=AAABc24B/kcGVtdY0U0nVCWLtob2wMX39p9CjP3ylWPUBvgBdRgkKE4I9EpmjpIkXjSsSmINGzVImQt9As7SKEC5lKY2d7Fmia48EwC70L8dp/RUZGZ26tq0gnzYde9y0Jo0kPLlozxD+108XyDjxeyij2//L9HIpJhVtUEYnAF+ByXoSeGoCc6+6e4ZDquxlzfHvgLM5RHwu\lLJ/BUmil/Ao0M32AakjNMPV0n10PESUwvFQNC/u1QW5VjRHHe6VXMdFWU2aRmcmHivdkwCLlaDB/XKbLYERp4sIDM99mkUwCvsCzfRfys2BJyQ==  
**upgrade-insecure-requests:** 1  
**user-agent:** Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_13\_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36

## ○ Response



response



## ▼ Response Headers

```
cache-control: no-cache, no-store, must-revalidate
content-encoding: gzip
content-length: 27896
content-type: text/html; charset=UTF-8
date: Sun, 08 Jul 2018 05:36:18 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=31536000; preload
vary: Accept-Encoding
x-frame-options: SAMEORIGIN
```

## ○ Content



response



```

<div class="special_bg">
<div class="area_flex">
<div class="area_logo">
<h1>
<a data-clk="top.logo" href="/"><span class="naver_logo">네이버</span></a>
</h1>
</div>
<div class="area_links">
<a data-clk="top.mkhome" href="http://help.naver.com/support/alias/contents2/naverhome/naverhome_1.naver" class="al_favorite">네이버를 시작페이지로</a>
<span class="al_bar"></span>
<a data-clk="top.jrnaver" href="http://jr.naver.com" class="al_jr"><span class="blind">주니어네이버</span><span class="al_ico"></span></a>
<a data-clk="top.happybean" href="http://happybean.naver.com/main/SectionMain.nhn" class="al_happybean"><span class="blind">해피빈</span><span class="al_ico"></span></a>
</div>
<div id="search" class="search">
<!--자동완성 입력창-->

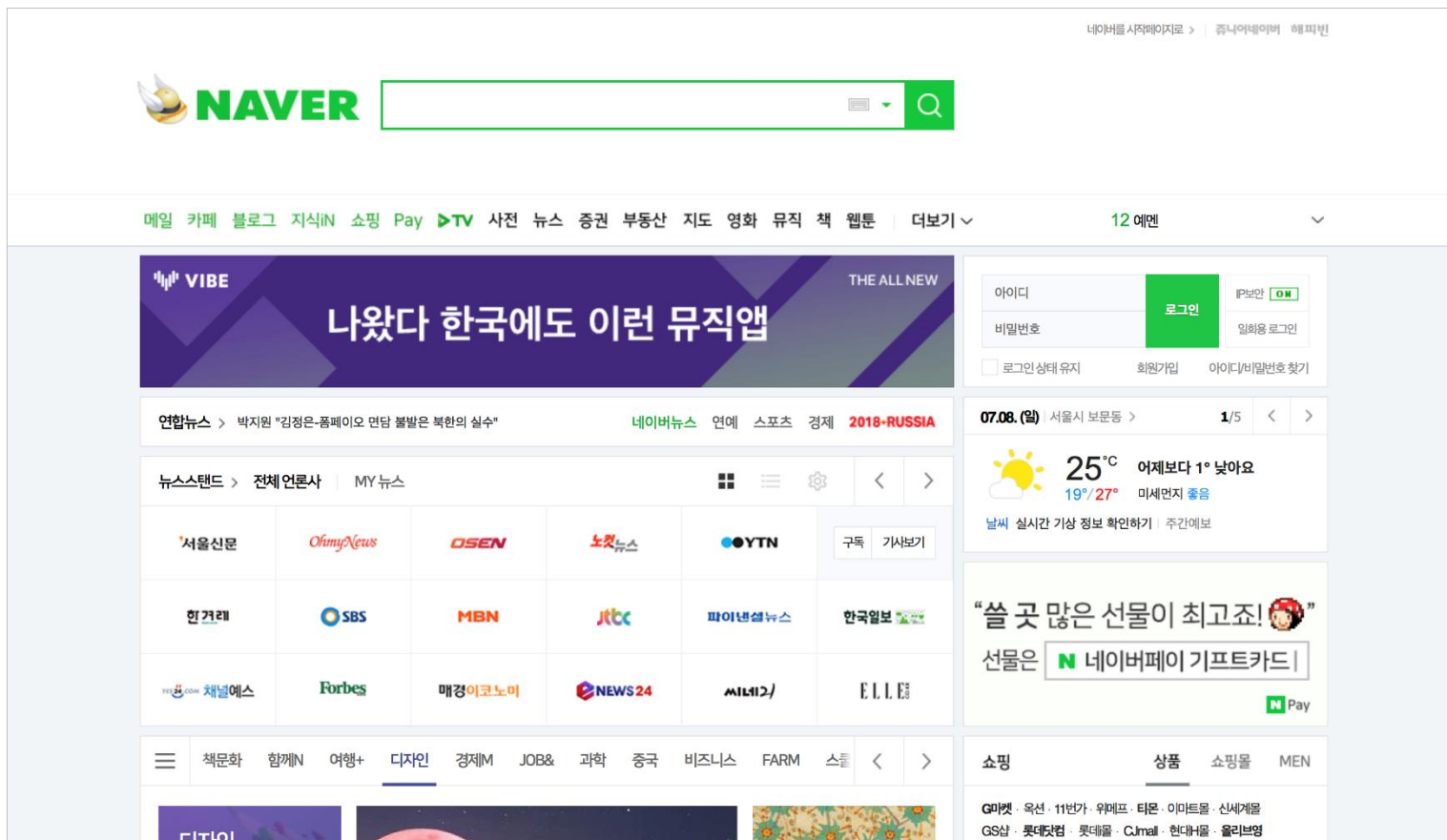
<form id="sform" name="sform" action="https://search.naver.com/search.naver" method="get">

  <fieldset>
    <legend class="blind">검색</legend>
    <select id="where" name="where" title="검색 범위 선택" class="blind">
      <option value="nsearch" selected">통합검색</option><option value="post">블로그</option><option value="cafeblog">카페</option><option value="cafe"> 카페명</option><option
value="article">- 카페글</option><option value="kin"> 지식iN</option><option value="news">뉴스</option><option value="web">사이트</option><option value="category">- 카테고리</option><option value="site">- 사이트
</option><option value="movie">영화</option><option value="webkr">웹문서</option><option value="dic">사전</option><option value="100">- 백과사전</option><option value="endic">- 영어사전</option><option
value="eedic">- 영영사전</option><option value="krdic">- 국어사전</option><option value="jpdic">- 일본어사전</option><option value="hanja">- 한자사전</option><option value="terms">- 용어사전</option><option
value="book">책</option><option value="music">음악</option><option value="doc">전문자료</option><option value="shop">쇼핑</option><option value="local">지역</option><option value="video">동영상</option><option
value="image">이미지</option><option value="mypc">내PC</option><optgroup label="스마트 파인더"><option value="movie">영화</option><option value="auto">자동차</option><option value="game">게임</option><option
value="health">건강</option><option value="people">인물</option></optgroup></select>

    <input type="hidden" id="sm" name="sm" value="top_hy" />
    <input type="hidden" id="fbm" name="fbm" value="0" />
    <input type="hidden" id="acr" name="acr" value="" disabled="disabled" />
    <input type="hidden" id="acq" name="acq" value="" disabled="disabled" />
    <input type="hidden" id="qdt" name="qdt" value="" disabled="disabled" />
    <input type="hidden" id="ie" name="ie" value="utf8" />
    <input type="hidden" id="acir" name="acir" value="" disabled="disabled" />
    <input type="hidden" id="os" name="os" value="" disabled="disabled" />
    <input type="hidden" id="bid" name="bid" value="" disabled="disabled" />
    <input type="hidden" id="pkid" name="pkid" value="" disabled="disabled" />
    <input type="hidden" id="eid" name="eid" value="" disabled="disabled" />
    <input type="hidden" id="mra" name="mra" value="" disabled="disabled" />
    <span class="green_window">
      <input id="query" name="query" type="text" title="검색어 입력" maxlength="255" class="input_text" tabindex="1" accesskey="s" style="ime-mode:active;" autocomplete="off"
onclick="document.getElementById('fbm').value=1; value="" />
    </span>
    <div id="nautocomplete" class="autocomplete">
      <!-- 자동완성 열린 경우 fold 클래스 추가, 닫힌 경우 dim 추가 -->
      <a href="javascript;" role="button" tabindex="2" class="btn_arw_btn_arw fold"><span class="blind_text">자동완성 필치기</span><span class="ico_arw"></span></a>
    </div>
    <button id="search_btn" type="submit" title="검색" tabindex="3" class="sch_smit" onmouseover="this.className='sch_smit over'" onmousedown="this.className='sch_smit down'"
onmouseout="this.className='sch_smit'" onclick="clicker(this,'sch.action','',event);"><span class="blind">검색</span><span class="ico_search_submit"></span></button>
  </fieldset>
</form>
<!--자동완성 입력창-->
<!--한글입력기-->
<a href="javascript;" id="ke_kbd_btn" role="button" class="btn_keyboard" onclick="nx_ime_load(this)" data-clk="sch.ime"><span class="blind">한글 입력기</span><span class="ico_keyboard"></span></a>
<style type="text/css" id="nx_kbd_style"></style>
<div id="nx_kbd" style="display:none;"></div>
<!--한글입력기-->

```

## ○ Rendering



#### ■ 불법인가? 합법인가?





**잡았다 요놈!**




## ■ 합법이다!

제주도 렌트카 가격 비교와 예약을 한번에! [로그인](#) [예약확인](#) [공지사항](#)

 예약가능 차량: 2,945개  최저가: ₩3,900

[최저가순](#) | [인기순](#)

최저가 차량 리스트	회사	차량가	보험가	기타
 쉐보레 뉴모닝(후) 경형 4 워털유 1000CC 보험선택대안함 ₩74,000 / 24시간 <b>₩ 3,900 / 24시간</b> 실시간 예약 가능 차량: 112 <a href="#">실시간 견제</a>	광	₩ 3,900 / 24시간	보험마선택 / 2일	유선
	타시오	₩ 4,200 / 24시간	보험마선택 / 2일	유선
	제주e	₩ 5,000 / 24시간	보험마선택 / 2일	유선
	대만	₩ 5,400 / 24시간	보험마선택 / 2일	유선
	용두임	₩ 5,900 / 24시간	보험마선택 / 2일	유선
	탐라	₩ 6,000 / 24시간	보험마선택 / 2일	유선
	금강	₩ 6,800 / 24시간	보험마선택 / 2일	유선
	제주아산	₩ 6,800 / 24시간	보험마선택 / 2일	유선
	유명	₩ 6,800 / 24시간	보험마선택 / 2일	유선
	한성	₩ 7,600 / 24시간	보험마선택 / 2일	유선
	제주사랑	₩ 7,600 / 24시간	보험마선택 / 2일	유선
	메이저	₩ 8,000 / 24시간	보험마선택 / 2일	유선
	원리산	₩ 8,500 / 24시간	보험마선택 / 2일	유선
	하이제주	₩ 8,500 / 24시간	보험마선택 / 2일	유선
	피사리	₩ 10,300 / 24시간	보험마선택 / 2일	유선
기자제주	₩ 16,000 / 24시간	보험마선택 / 2일	유선	
엑스	₩ 16,800 / 24시간	보험마선택 / 2일	유선	

## ■ 불법이다!



## ■ 불법 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**

리그베다위키 저작권 침해 사건서 DB제작자 권리 인정받아 최종 승소

2017-12-15

법무법인 민후는 서브컬처 위키백과 '리그베다위키(옛 엔하위키)' 운영자를 대리해 '엔하위키 미래' 운영자를 상대로 저작권 침해 및 부정경쟁방지법 위반 등에 기인한 형사고소 및 손해배상청구 소송을 제기하고 최종 승소했습니다.

**'야놀자 DB 무단 크롤링' 심명섭 전 여기어때 대표, 1심서 집행유예 2년(종합)**

2020.02.11 15:25

■ 불법은 아니지만 조심하자

1. 스크랩하는 콘텐츠에 지적재산권이 있는지
2. 크롤링 하는 행동이 사이트에 큰 부담을 주지 않는지
3. 크롤러가 사이트의 이용방침을 위반하지 않는지
4. 크롤러가 사용자의 민감한 정보를 가져오지 않는지
5. 가져온 콘텐츠를 적합한 사용 표준하에 사용하는지

## ■ robots.txt

- 웹사이트에 Crawler같은 (ro)bot들의 접근을 제어하기 위한 규약
- 아직 권고안이라 꼭 지킬 의무는 없음

### 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: /  
""
```

## ■ 데이터 3법

### 데이터 3법 개정안 주요 내용

개인정보 보호법	개인 식별 불가능한 '가명정보'를 제품·서비스 개발에 활용 허용. 개인정보 관리·감독 권한 개인정보보호위원회로 일원화
정보 통신망법	산재된 개인정보 관련 법체계 개인정보보호법으로 이관
신용 정보법	'가명정보' 금융분야 빅데이터 분석에 활용 허용. 통계 작성·연구 등에 개인신용정보 활용 허용

### 개인정보·가명정보·익명정보는

예시	특징
<b>개인정보</b>  홍길동 990909-1234567 010-1234-5678 서울 도봉구 방학로4길 google@gmail.com 통신요금 7만6500원 기관지염	정보주체의 이익, 안전조치(암호화) 여부 등에 따라 동의 없이 활용 가능
<b>가명정보</b>  홍** 99년생, 남자 010-****-**** 서울 도봉구 *****@gmail.com 통신요금 7만6500원 기관지염	통계작성, 과학적 연구, 공익적 기록보존 등에 동의 없이 활용 가능
<b>익명정보</b>  - 20대, 남자 서울 통신요금 7만6500원 기관지염	언제든지 동의 없이 활용 가능

자료: 정부부처 합동『개인정보 비식별 조치 가이드라인』,  
한국인터넷진흥원(KISA) 참고

## ■ 사용된 웹 기술 확인

### ○ builtwith

```
#!pip install builtwith

import builtwith

builtwith.parse('http://www.google.com')
builtwith.parse('http://www.facebook.com')
builtwith.parse('http://www.wordpress.com')
builtwith.parse('http://example.webscrapping.com')
```

## ■ 웹사이트 소유자 확인

### ○ whois

```
#!pip install python-whois

import whois

print(whois.whois("naver.com"))
print(whois.whois("appspot.com"))
```

## ■ 웹사이트 다운로드

### ○ urllib

```
import urllib
```

`urllib` is a package that collects several modules for working with URLs:

- `urllib.request` for opening and reading URLs
- `urllib.error` containing the exceptions raised by `urllib.request`
- `urllib.parse` for parsing URLs
- `urllib.robotparser` for parsing `robots.txt` files

### ○ request

The `urllib.request` module defines functions and classes which help in opening URLs (mostly HTTP) in a complex world — basic and digest authentication, redirections, cookies and more.

```
from urllib.request import urlopen
```



➤ urlopen

`urllib.request.urlopen(url, data=None, [timeout, ], cafile=None, capath=None, cadefault=False, context=None)`

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

```
resp = urlopen("http://python.org")  
html = resp.read()
```

○ response

The `urllib.response` module defines functions and classes which define a minimal file like interface, including `read()` and `readline()`. The typical response object is an `addinfourl` instance, which defines an `info()` method and that returns headers and a `geturl()` method that returns the url. Functions defined by this module are used internally by the `urllib.request` module.

➤ read

➤ info

## ○ error

The `urllib.error` module defines the exception classes for exceptions raised by `urllib.request`. The base exception class is `URLError`.

**code**

An HTTP status code as defined in [RFC 2616](#). This numeric value corresponds to a value found in the dictionary of codes as found in `http.server.BaseHTTPRequestHandler.responses`.

**reason**

This is usually a string explaining the reason for this error.

**headers**

The HTTP response headers for the HTTP request that caused the `HTTPError`.

```
from urllib.error import HTTPError

try:
    resp = urlopen("http://www.google.com/search?q=korean")
except HTTPError as e:
    print("error:", e.code, e.reason, e.headers)
```

## ■ HTTP Errors

### ○ 4XX Client Errors

- This class of status code is intended for situations in which the error seems to have been caused by the client

### ○ 5XX Server Errors

- The server failed to fulfil a request

```
def download(url, num_retries=2):
    try:
        html = urlopen(url).read().decode("utf-8")
    except HTTPError as e:
        html = None

        print("error:", e.code, e.reason)

        if 500 <= e.code < 600 and num_retries > 0:
            return download(url, num_retries-1)

    return html

#download("http://www.google.com/search?q=korean")
download("http://httpstat.us/500")
```

## ■ Header

### Forbidden

You don't have permission to access / on this server.

#### ○ Request

*`class urllib.request.Request(url, data=None, headers={}, origin_req_host=None, unverifiable=False, method=None)`*

This class is an abstraction of a URL request.

*headers* should be a dictionary, and will be treated as if `add_header()` was called with each key and value as arguments. This is often used to “spoof” the `User-Agent` header value, which is used by a browser to identify itself – some HTTP servers only allow requests coming from common browsers as opposed to scripts. For example, Mozilla Firefox may identify itself as `"Mozilla/5.0 (X11; U; Linux i686) Gecko/20071127 Firefox/2.0.0.11"`, while `urllib`'s default user agent string is `"Python-urllib/2.6"` (on Python 2.6).

## ○ 예제

```
from urllib.request import Request

def download(url, agent="python ", num_retries=2):
    headers = {'User-agent':agent}
    req = Request(url, headers=headers)

    try:
        html = urlopen(req).read().decode("utf-8")
    except HTTPError as e:
        html = None

        print("error:", e.code, e.reason)

        if 500 <= e.code < 600 and num_retries>0:
            return download(url, num_retries-1)

    return html

download("http://www.google.com/search?q=korean")
```

## ■ urllib vs Requests

See also: The [Requests package](#) is recommended for a higher-level HTTP client interface.

`requests.request(method, url, **kwargs)`

Constructs and sends a **Request**.

- Parameters:**
- **method** – method for the new **Request** object.
  - **url** – URL for the new **Request** object.
  - **params** – (optional) Dictionary or bytes to be sent in the query string for the **Request**.
  - **data** – (optional) Dictionary or list of tuples [(key, value)] (will be form-encoded), bytes, or file-like object to send in the body of the **Request**.
  - **json** – (optional) A JSON serializable Python object to send in the body of the **Request**.
  - **headers** – (optional) Dictionary of HTTP Headers to send with the **Request**.
  - **cookies** – (optional) Dict or CookieJar object to send with the **Request**.
  - **files** – (optional) Dictionary of 'name': file-like-objects (or {'name': file-tuple}) for multipart encoding upload. file-tuple can be a 2-tuple ('filename', fileobj), 3-tuple ('filename', fileobj, 'content\_type') or a 4-tuple ('filename', fileobj, 'content\_type', custom\_headers), where 'content-type' is a string defining the content type of the given file and custom\_headers a dict-like object containing additional headers to add for the file.
  - **auth** – (optional) Auth tuple to enable Basic/Digest/Custom HTTP Auth.
  - **timeout** (float or tuple) – (optional) How many seconds to wait for the server to send data before giving up, as a float, or a (connect timeout, read timeout) tuple.
  - **allow\_redirects** (bool) – (optional) Boolean. Enable/disable GET/OPTIONS/POST/PUT/PATCH/DELETE/HEAD redirection. Defaults to **True**.
  - **proxies** – (optional) Dictionary mapping protocol to the URL of the proxy.
  - **verify** – (optional) Either a boolean, in which case it controls whether we verify the server's TLS certificate, or a string, in which case it must be a path to a CA bundle to use. Defaults to **True**.
  - **stream** – (optional) if **False**, the response content will be immediately downloaded.
  - **cert** – (optional) if String, path to ssl client cert file (.pem). If Tuple, ('cert', 'key') pair.

**Returns:**

**Response** object

```
import requests

def download(url, agent="python", num_retries=2):
    headers = {'User-agent': agent}

    req = requests.request("get", url, headers=headers)

    if 500 <= req.status_code < 600 and num_retries > 0:
        print("error:", req.status_code, req.reason)
        return download(url, num_retries=(num_retries-1))

    return req

#html = download("http://www.google.com/search?q=korean")
html = download("http://httpstat.us/500")
print(html)
```

## ■ urllib 한글

### ○ parse

```
from urllib import parse
```

This module defines a standard interface to break Uniform Resource Locator (URL) strings up in components (addressing scheme, network location, path etc.), to combine the components back into a URL string, and to convert a “relative URL” to an absolute URL given a “base URL.”

#### ➤ parsing

`urllib.parse.urlparse(urlstring, scheme="", allow_fragments=True)`

Parse a URL into six components, returning a 6-tuple. This corresponds to the general structure of a URL: `scheme://netloc/path;parameters?query#fragment`. Each tuple item is a string, possibly empty. The components are not broken up in smaller parts (for example, the network location is a single string), and % escapes are not expanded. The delimiters as shown above are not part of the result, except for a leading slash in the *path* component, which is retained if present. For example:

#### ➤ quoting

`urllib.parse.quote(string, safe='/', encoding=None, errors=None)`

Replace special characters in *string* using the `%xx` escape. Letters, digits, and the characters `'_.-~'` are never quoted. By default, this function is intended for quoting the path section of URL. The optional *safe* parameter specifies additional ASCII characters that should not be quoted — its default value is `'/'`.

## ○ 예제

```
parse.quote("한글")
```

```
from urllib import parse
from urllib.request import Request

def download(url, agent="python ", num_retries=2):
    print(url)
    headers = {'User-agent':agent}
    req = Request(url, headers=headers)

    try:
        html = urlopen(req).read().decode("utf-8")
    except HTTPError as e:
        html = None

        print("error:", e.code, e.reason)

        if 500 <= e.code < 600 and num_retries>0:
            return download(url, num_retries-1)

    return html

html = download("http://www.google.com/search?q="+parse.quote("한글")+"&ie=utf-8&oe=utf-8")
#html = download("http://search.naver.com/search.naver?query="+parse.quote("한글"))
html
```



## ○ 예제

```
import requests

def download(url, agent="python", num_retries=2):
    headers = {'User-agent':agent}

    req = requests.request("get", url, headers=headers)

    if 500 <= req.status_code < 600 and num_retries>0:
        print("error:", req.status_code, req.reason)
        return download(url, num_retries=(num_retries-1))

    return req

html = download("http://www.google.com/search?q=한글&ie=utf-8&oe=utf-8")

print(html.encoding)

html.encoding = "utf-8"

print(type(html.text))
print(type(html.content))

html.text
```

○ get / post

`requests.get(url, params=None, **kwargs)`

[source]

Sends a GET request.

**Parameters:**

- **url** – URL for the new **Request** object.
- **params** – (optional) Dictionary or bytes to be sent in the query string for the **Request**.
- **\*\*kwargs** – Optional arguments that `request` takes.

**Returns:** **Response** object

**Return type:** `requests.Response`

`requests.post(url, data=None, json=None, **kwargs)`

[source]

Sends a POST request.

**Parameters:**

- **url** – URL for the new **Request** object.
- **data** – (optional) Dictionary (will be form-encoded), bytes, or file-like object to send in the body of the **Request**.
- **json** – (optional) json data to send in the body of the **Request**.
- **\*\*kwargs** – Optional arguments that `request` takes.

**Returns:** **Response** object

**Return type:** `requests.Response`

## ○ 예제

➤ get

```
import requests

def download(url, agent="python", num_retries=2):
    headers = {'User-agent':agent}

    req = requests.get(url, params={"q":"한글", "ie":"utf-8", "oe":"utf-8"}, headers=headers)

    if 500 <= req.status_code < 600 and num_retries>0:
        print("error:", req.status_code, req.reason)
        return download(url, num_retries=(num_retries-1))

    return req

html = download("http://www.google.com/search")

html.url
```

## ○ 예제

➤ post - dict

```
import requests

def download(url, agent="python", num_retries=2):
    headers = {'User-agent':agent}

    req = requests.post(url, data={"q":"한글", "ie":"utf-8", "oe":"utf-8"}, headers=headers)

    if 500 <= req.status_code < 600 and num_retries>0:
        print("error:", req.status_code, req.reason)
        return download(url, num_retries=(num_retries-1))

    return req

html = download("http://httpbin.org/post")

print(html.url)
print(html.text)
```

## ○ 예제

➤ post - json

```
import json
import requests

def download(url, agent="python", num_retries=2):
    headers = {'User-agent':agent}

    req = requests.post(url, json=json.dumps({"q":"한글", "ie":"utf-8", "oe":"utf-8"}), headers=headers)

    if 500 <= req.status_code < 600 and num_retries>0:
        print("error:", req.status_code, req.reason)
        return download(url, num_retries=(num_retries-1))

    return req

html = download("http://httpbin.org/post")

print(html.url)
print(html.text)
```