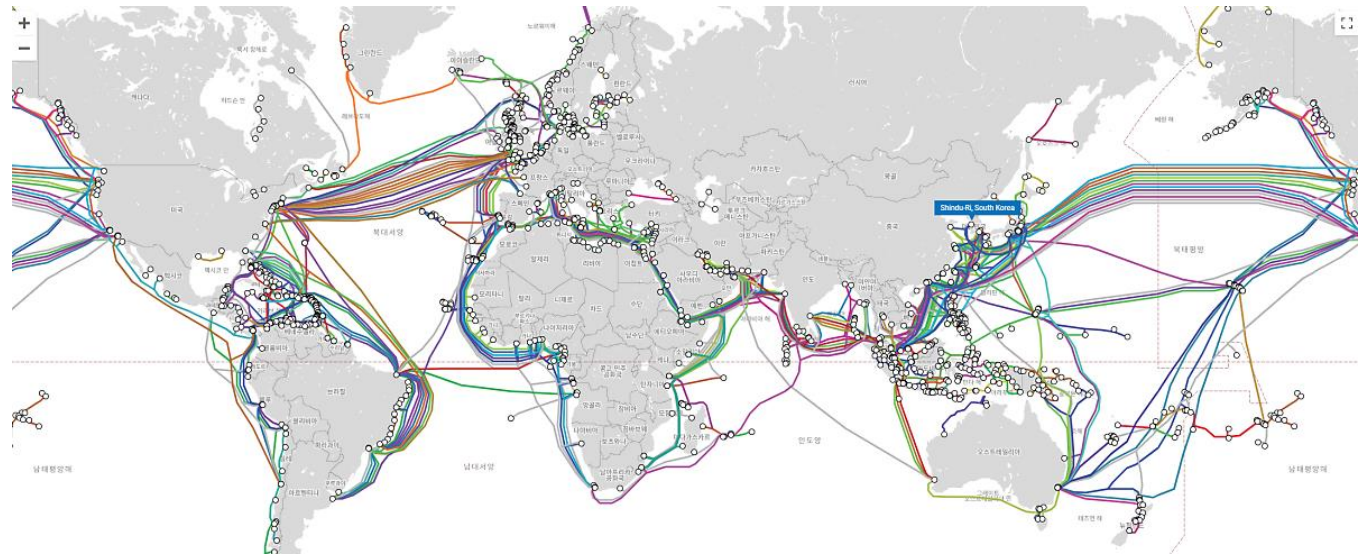


# 01\_웹 크롤링 시작하기

## • 인터넷이란?

- > 인터넷은 컴퓨터로 연결하여 TCP/IP 라는 통신 프로토콜을 이용해 정보를 주고받는 컴퓨터 네트워크
- > 미국 국방부의 군사용 통신을 위해 개발된 것이 발전하여 현재의 인터넷이 됨
- > 해저케이블을 이용하여 전세계 서버에 접속하여 통신하고 있음
- > 한국은 미국에 이어 세계 2번째로 인터넷 연결에 성공했다.



## • 웹 사이트의 동작 과정

### > 클라이언트

- 사용자가 웹 사이트에 접근할 때 사용하는 기기
- 웹 브라우저

### > 서버

- 인터넷에 연결된 컴퓨터
- 웹 요소와 여러 정보가 저장됨



- URL

- Uniform Resource Locator

> 일반적으로 특정 사이트에 접속하기 위한 주소

> `http://news.naver.com:80/main/read.nhn?mode=LSD&mid=shm&sid1=105&oid=001&aid=0009847211#da_727145`

- `https://` - Protocol
- `news` - sub Domain
- `naver.com` - Domain
- `80` - Port
- `main` - path
- `read.nhn` - filename
- `mode=LSD&mid=shm&sid1=105&oid=001&aid=0009847211` - query
- `#da_727145` - fragment

## • Get & Post

- 데이터를 주고 받는 방식

### > Get

- Url에 데이터가 포함된다.
- 따라서 데이터가 노출됨
- 길이에 제한이 있음

### > Post

- Body에 데이터가 포함된다.
- 따라서 데이터 노출이 없음
- 길이에 제한이 없다



• HTTP Status Code

- > 서버와 클라이언트가 데이터를 주고 받은 결과로 상태 코드를 확인할 수 있다.
  - 2xx - success
  - 3xx - redirection
  - 4xx - request error
  - 5xx - server error
- > 주요 상태코드

상태	코드	메시지	설 명
정상	200	OK	클라이언트의 요청은 정상적으로 처리됨
리 다이렉트	304	Not Modified	요청된 문서는 변경되지 않음
클라이언트 에러	400	Bad Request	비정상적인 요청임
	404	Not Found	요청된 URI가 서버상에 존재하지 않음
서버에러	500	Internal Sever Error	서버 내부에 문제가 발생함
	<b>503</b>	Service Unavailable	서비스를 이용할 수 없음

## • 크롤링 용어 정리

- > Scraping
  - 데이터를 수집하는 작업
- > Crawling
  - 여러페이지의 특정 데이터들을 수집하고 분류하는 작업
- > Spider & Web crawler
  - 웹 데이터를 수집하는 소프트웨어
- > Bot
  - 인터넷 상에서 자동화된 작업을 실행하는 소프트웨어

통상적으로 인터넷에서 데이터를 수집하는 행위를 크롤링이라고 부른다.

- 웹 크롤링 방법

- > requests 이용
  - 받아오는 문자열에 따라 두가지 방법으로 구분  
: json 타입, html 타입
- > selenium 이용
  - 브라우저를 직접 열어서 데이터를 받는 방법

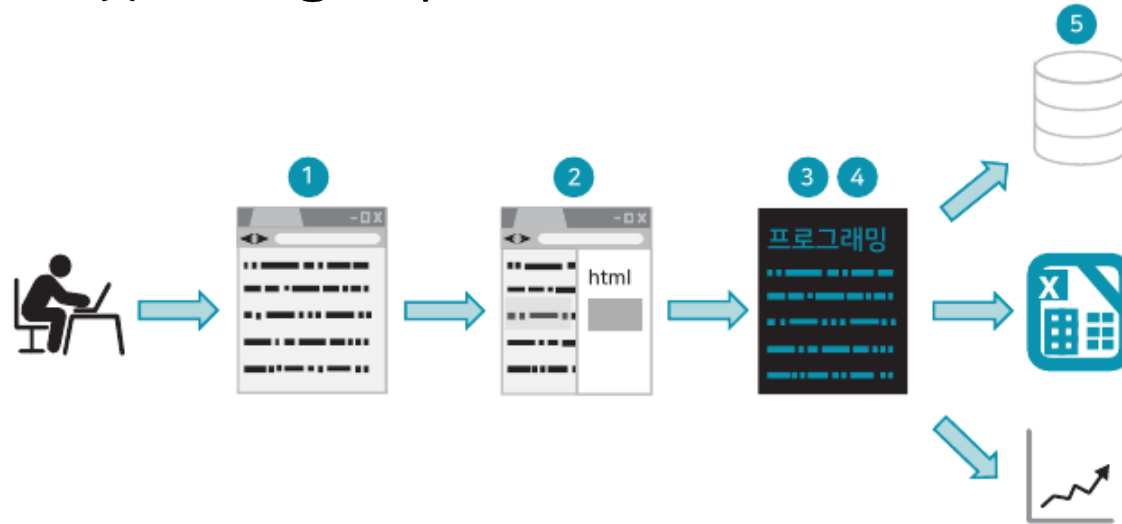
	정적 수집	동적 수집
사용 패키지	requests / urllib	selenium
수집 커버리지	정적 웹 페이지	정적/동적 웹 페이지
수집 속도	빠름 (별도 페이지 조작 필요 X)	상대적으로 느림
파싱 패키지	beautifulsoup	beautifulsoup / selenium



# 웹 크롤링 시작하기

## • 웹 크롤링 프로세스

### > 웹 크롤링의 다섯가지 과정 요약



- > ① 웹 페이지 접속(정보를 얻고자 하는 사이트)
- > ② 브라우저 디벨로퍼 툴(F12)을 사용하여 정보의 위치 확인
- > ③ 파이썬 코드를 사용하여 HTML 소스 불러오기
- > ④ 불러온 데이터에서 원하는 정보를 가공 후 추출
- > ⑤ 추출한 정보를 CSV, 데이터베이스 등 다양한 형태로 저장, 가공, 시각화

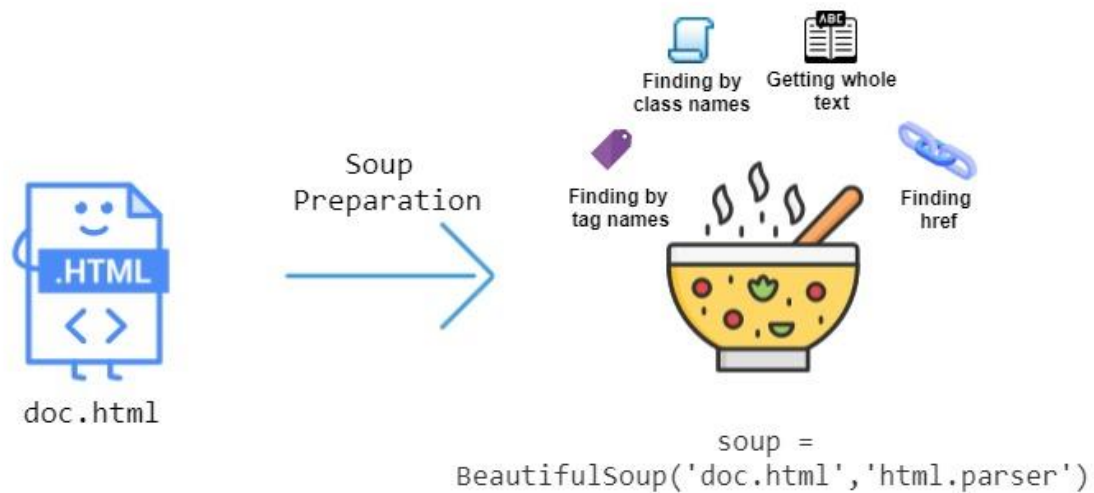
## • 크롤링 기초 실습

### > BeautifulSoup 임포트

```
From bs4 import BeautifulSoup
```

### > BeautifulSoup이란?

- HTML Parser
- HTML과 XML 문서들의 구문을 분석하기 위한 파이썬 패키지



- 크롤링 기초 실습

> BeautifulSoup

```
html_doc = """
<!doctype html>
<html>
  <head>
    <title> 기초 크롤링 </title>
  </head>
  <body>
    크롤링을 해봅시다.
  </body>
</html>
"""
```

## • 크롤링 기초 실습

> BeautifulSoup

```
BeautifulSoup(html_doc, "html.parser")
```

Out :

<!DOCTYPE html>

<html>

<head>

<title> 기초 크롤링 </title>

</head>

<body>

크롤링을 해봅시다.

</body>

</html>

HTML 문서의 문자열

HTML Parser의 종류

Beautifulsoup  
객체를 반환한다

• 크롤링 기초 실습

> HTML Parser의 종류

Parser	선언방법	장점	단점
html.parser	BeautifulSoup(markup, 'html.parser')	파이썬에 포함된 모듈이라 설치할 필요 없음	일부 복잡한 HTML 문서를 처리하는 데 있어서 다른 파서들보다 성능이 떨어짐
lxml	BeautifulSoup(markup, 'lxml')	매우 빠름, XML 문서를 처리하는데 용이	lxml 추가 설치 필요
html5lib	BeautifulSoup(markup, 'html5lib')	웹 브라우저와 같은 방식으로 페이지를 파싱 유효한 HTML5 생성, 복잡한 HTML 문서를 처리하는 데 안정적	html5lib 추가 설치 필요 매우 느림

- 크롤링 기초 실습

> HTML parsing

```
bs_obj = BeautifulSoup(html_doc, "html.parser")
head = bs_obj.find("head")
print(head)
```

**Out :** <head>  
          <title> 기초 크롤링 </title>  
          </head>

```
body = bs_obj.find("body")
print(body)
```

**Out :** <body>  
          크롤링을 해봅시다.  
          </body>

- 크롤링 기초 실습

> HTML parsing

```
bs_obj = BeautifulSoup(html_doc, "html.parser")
head = bs_obj.select_one("head")
print(head)
```

**Out :** <head>  
    <title> 기초 크롤링 </title>  
    </head>

```
body = bs_obj.select_one("body")
print(body)
```

**Out :** <body>  
    크롤링을 해봅시다.  
    </body>

- 크롤링 기초 실습

- > HTML parsing

```
html_doc = """
<!doctype html>
<html>
  <head>
    기초 크롤링 따라하기
  </head>
  <body>
    <div> 첫 번째 부분 </div>
    <div> 두 번째 부분 </div>
  </body>
</html>
"""
```



- 크롤링 기초 실습

> HTML parsing

```
bs_obj = BeautifulSoup(html_doc, "html.parser")  
body = bs_obj.find("body")  
print(body)
```

**Out :** <body>  
    <div> 첫 번째 부분 </div>  
    <div> 두 번째 부분 </div>  
    </body>

```
div1 = bs_obj.find("div")  
print(div1)
```

**Out :** <div> 첫 번째 부분 </div>

- 크롤링 기초 실습

> HTML parsing

```
div_total = bs_obj.find_all("div")  
print(div_total)
```

**Out :** [<div> 첫 번째 부분 </div>, <div> 두 번째 부분 </div>]

```
div_total = bs_obj.select("div")  
print(div_total)
```

**Out :** [<div> 첫 번째 부분 </div>, <div> 두 번째 부분 </div>]

## • 크롤링 기초 실습

### > HTML parsing

```
html_doc = """
<!doctype html>
<html>
  <head>
    <title> 기초 크롤링 </title>
  </head>
  <body>
    <table id="tb1" border="1">
      <caption> 과일 가격 </caption>
      <tr><th> 상품 </th><th> 가격 </th></tr>
      <tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>
      <tr class="fruits"><td> 사과 </td><td> 150 </td></tr>
    </table>
    <table id="tb2" border="2">
      <caption> 의류 가격 </caption>
      <tr><th> 상품 </th><th> 가격 </th></tr>
      <tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>
      <tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>
    </table>
  </body>
</html>
"""
```

상품	가격
오렌지	100
사과	150

상품	가격
셔츠	30000
바지	50000

- 크롤링 기초 실습

> HTML parsing

```
bs_obj = BeautifulSoup(html_doc, "html.parser")
tables = bs_obj.find_all("table")
print(tables)
```

**Out :** [<table border="1" id="tb1">  
<caption> 과일 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>  
<tr class="fruits"><td> 사과 </td><td> 150 </td></tr>  
</table>, <table border="2" id="tb2">  
<caption> 의류 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>  
<tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>  
</table>]

- 크롤링 기초 실습

> HTML parsing

```
table1 = bs_obj.find("table", {"id":"tb1"})  
print(table1)
```

**Out :** <table border="1" id="tb1">  
<caption> 과일 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>  
<tr class="fruits"><td> 사과 </td><td> 150 </td></tr>  
</table>

```
table2 = bs_obj.find("table", {"border":"2"})  
print(table2)
```

**Out :** <table border="2" id="tb2">  
<caption> 의류 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>  
<tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>  
</table>

- 크롤링 기초 실습

> HTML parsing

```
fruits = bs_obj.find_all("tr", {"class":"fruits"})  
print(fruits)
```

**Out :** [<tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>, <tr class="fruits"><td> 사과 </td><td> 150 </td></tr>]

```
clothes = bs_obj.find_all("tr", {"class":"clothes"})  
print(clothes)
```

**Out :** [<tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>, <tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>]

- 크롤링 기초 실습

> HTML parsing

```
table1 = bs_obj.select_one("table#tb1")  
print(table1)
```

**Out :** <table border="1" id="tb1">  
<caption> 과일 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>  
<tr class="fruits"><td> 사과 </td><td> 150 </td></tr>  
</table>

```
table2 = bs_obj.select_one("table[border='2']")  
print(table2)
```

**Out :** <table border="2" id="tb2">  
<caption> 의류 가격 </caption>  
<tr><th> 상품 </th><th> 가격 </th></tr>  
<tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>  
<tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>  
</table>

- 크롤링 기초 실습

> HTML parsing

```
fruits = bs_obj.select("tr.fruits")  
print(fruits)
```

**Out :** [<tr class="fruits"><td> 오렌지 </td><td> 100 </td></tr>, <tr class="fruits"><td> 사과 </td><td> 150 </td></tr>]

```
clothes = bs_obj.select("tr.clothes")  
print(clothes)
```

**Out :** [<tr class="clothes"><td> 셔츠 </td><td> 30000 </td></tr>, <tr class="clothes"><td> 바지 </td><td> 50000 </td></tr>]



## • CSS Selector 종류

### > Element Selector

- element를 이용하여 선택할 때 사용
- css selector로 div 검색하면 div를 사용하는 태그를 선택

```
<div>dss1</div>
```

```
<p>dss2</p>
```

```
<span>dss3</span>
```

## • CSS Selector 종류

### > ID Selector

- 아이디를 이용하여 선택할 때 사용
- #(아이디 이름) 으로 선택 가능
- #ds2로 검색하면 dss2가 선택됨
- #ds2, #ds3 로 여러 개 검색하여 선택가능

```
<p id="ds1">dss1</p>
```

```
<p id="ds2">dss2</p>
```

```
<p id="ds3">dss3</p>
```

## • CSS Selector 종류

### > Class Selector

- 클래스 이름을 이용하여 선택할 때 사용
- .(클래스 이름) 으로 선택 가능
- .ds2로 검색하면 dss2, dss3가 선택됨

```
<p class="ds1">dss1</p>
```

```
<p class="ds2">dss2</p>
```

```
<p class="ds2">dss3</p>
```

## • CSS Selector 종류

### > First-child Selector

- elemen로 감싸져 있는 가장 첫 결과가 나옴
- .ds:first-child 로 검색하면 ds1, ds3이 나옴

```
<body>
  <p class="ds" id="ds1">ds1 </p>
  <p class="sc" id="ds2">ds2 </p>
  <div class="ds">
    <p class="ds">ds3 </p>
    <p class="ds">ds4 </p>
    <p class="ds">ds5 </p>
    <p class="ds">ds6 </p>
    <p class="ds">ds7 </p>
  </div>
</body>
```

## • CSS Selector 종류

### > Last-child Selector

- elemen로 감싸져 있는 가장 마지막 결과가 나옴
- .ds:last-child 로 검색하면 ds3~ds7, ds7이 나옴

```
<body>
  <p class="ds" id="ds1">ds1 </p>
  <p class="sc" id="ds2">ds2 </p>
  <div class="ds">
    <p class="ds">ds3 </p>
    <p class="ds">ds4 </p>
    <p class="ds">ds5 </p>
    <p class="ds">ds6 </p>
    <p class="ds">ds7 </p>
  </div>
</body>
```

## • CSS Selector 종류

### > Nth-child Selector

- elemen로 감싸져 있는 N번째 결과가 나옴
- .ds:nth-child(3)으로 검색하면 ds4가 나옴
- 숫자는 0이 아닌 1로 시작( 1 == first-child )

```
<div class="wrap">  
  <span class="ds">ds2</span>  
  <p class="ds ds1">ds3</p>  
  <p class="ds ds2">ds4</p>  
  <p class="ds ds3">ds5</p>  
  <p class="ds ds4">ds6</p>  
  <p class="ds ds5">ds7</p>  
</div>
```

## • CSS Selector 종류

### > Depth(공백) Selector

- 공백 문자로 하위 element를 선택 가능
- <div> 하위 element 모두에서 선택
- .contants h1을 선택하면 inner\_1, inner\_3가 선택됨

```
<div class="contants">
  <h1>inner_1</h1>
  <p> inner_2</p>
  <span class='txt'>
    <h1>inner_3</h1>
  </span>
</div>
```

## • CSS Selector 종류

### > Depth(>) Selector

- > 문자로 하위 element를 선택 가능
- <div> 하위 element 에서만 선택
- .contants>h1을 선택하면 inner\_1만 선택됨

```
<div class="contants">
  <h1>inner_1</h1>
  <p> inner_2</p>
  <span class='txt'>
    <h1>inner_3</h1>
  </span>
</div>
```



- 크롤링 사이트 확인

- <https://ai-dev.tistory.com/1>

- > 실제 블로그 크롤링 실습

크롤링

**크롤링의 세계에 오신 것을 환영합니다.**

로스카츠 | 2021. 1. 21. 14:06

Hello, world!

♡ 13



...

구독하기

'크롤링' 카테고리의 다른 글

크롤링 예제 페이지 - 02 (6)

2021.08.08

- > urllib 또는 requests 라이브러리를 통해 HTML 값 가져오기

- 크롤링 사이트 확인

```
from urllib.request import urlopen

url = "https://ai-dev.tistory.com/1"
html = urlopen(url)
print(html.read())
```

**Out :**

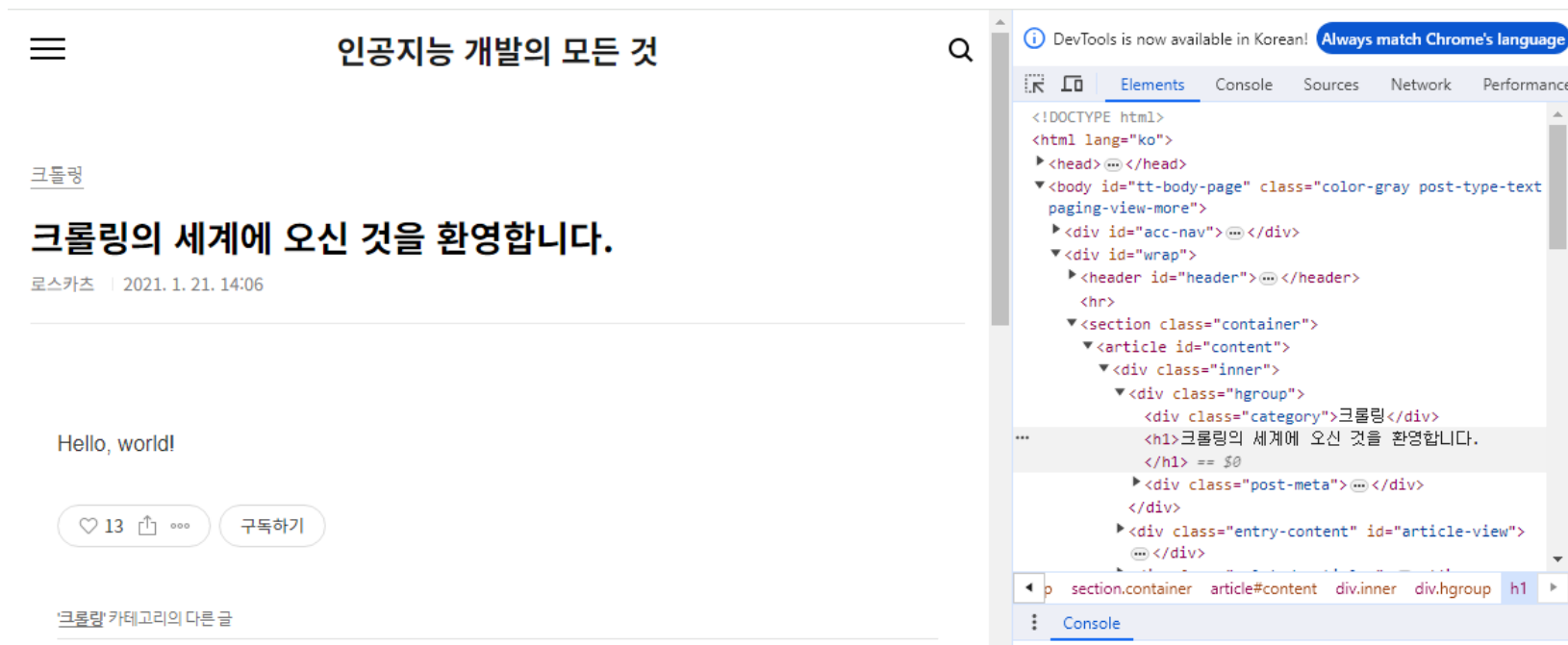
```
b'<!doctype html>#n<html lang="ko">#nn#n          <head>#nn
<script type="text/javascript">if (!window.T) { window.T = {} }#nnwindow.T.config = {"TOP_SSL_URL":"https://www.tistory.com","PREVIEW":f
alse,"ROLE":{"guest"},"PREV_PAGE":"","NEXT_PAGE":"","BLOG":{"id":4442027,"name":"ai-dev","title":"#xec#w9d#w8#wxa#xb3#wb5#wxec#wxa7#w80#wxe
b#wx8a#wxa5 #xea#wxb0#w9c#wxb#wx0#w9c#wec#wx9d#wx98 #xeb#wxaa#wx8#wxb#wx93#wxa0 #xea#wxb2#wx83"},"isDormancy":false,"nickName":"#xeb#wxa1#w9c#wec#wx
8a#wxa4#wec#wxb9#wx4#wec#wxb8#wxa0"},"status":{"open"},"NEED_COMMENT_LOGIN":false,"COMMENT_LOGIN_CONFIRM_MESSAGE":"","LOGIN_URL":"","https://ww
w.tistory.com/auth/login?redirectUrl=https://ai-dev.tistory.com/1"},"DEFAULT_URL":"https://ai-dev.tistory.com"},"USER":{"name":null,"hom
epage":null,"id":0,"profileImage":null},"SUBSCRIPTION":{"status":"none","isConnected":false,"isPending":false,"isWait":false,"isProcess
ing":false,"isNone":true},"IS_LOGIN":false,"HAS BLOG":false,"IS_SUPPORT":false,"TOP_URL":"http://www.tistory.com","JOIN_URL":"https://w
ww.tistory.com/member/join"},"ROLE_GROUP":"visitor"};#nnwindow.appInfo = {"domain":"tistory.com","topUri":"https://www.tistory.com","logi
nUri":"https://www.tistory.com/auth/login","logoutUri":"https://www.tistory.com/auth/logout"};#nnwindow.initData = {};#nn#nnwindow.Tistory
Blog = {#nn basePath: "",#nn url: "https://ai-dev.tistory.com/#nn tistoryUri: "https://ai-dev.tistory.com/#nn manageUri: "htt
ps://ai-dev.tistory.com/manage"#nn token: "8uU4GB+x47Q/OSBl+XDpgOrhmjGw780bw7/p9YghDKsCZInno+beLgD9PkOhtvlj"#nn};#nnvar servicePath =
```

# 웹 크롤링 시작하기

## • 크롤링 사이트 확인

### > urllib.request

- 개발자 툴을 사용하여 제목 태그 찾기



- 크롤링 사이트 확인

> urllib.request

```
from urllib.request import urlopen
from bs4 import BeautifulSoup

url = "https://ai-dev.tistory.com/1"
html = urlopen(url)
bs_obj = BeautifulSoup(html, "html.parser")
title = bs_obj.find_all("h1")
print(title)
```

**Out :** [<h1><a href="/">인공지능 개발의 모든 것</a></h1>, <h1>크롤링의  
세계에 오신 것을 환영합니다. </h1>]

- 크롤링 사이트 확인

> urllib.request

```
url = "https://ai-dev.tistory.com/1"
html = urlopen(url)
bs_obj = BeautifulSoup(html, "html.parser")
title = bs_obj.select_one("div > h1")
print(title)
```

**Out :** <h1>크롤링의 세계에 오신 것을 환영합니다. </h1>

```
print(title.text)
```

**Out :** 크롤링의 세계에 오신 것을 환영합니다.

- 크롤링 사이트 확인

> requests

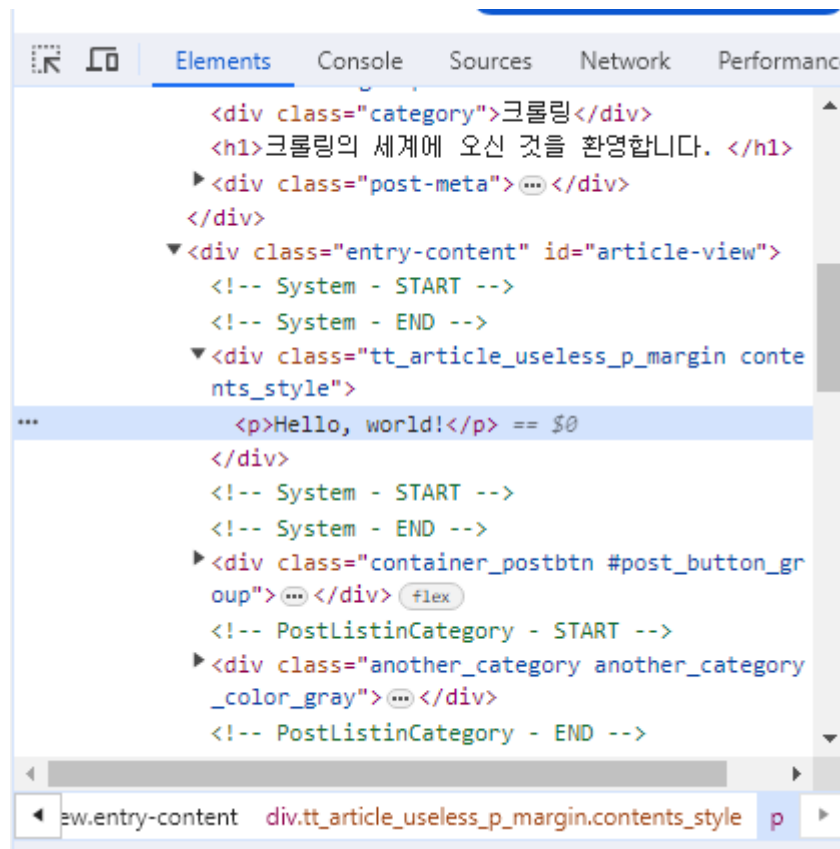
```
import requests

url = "https://ai-dev.tistory.com/1"
response = requests.get(url)
html = response.content
bs_obj = BeautifulSoup(html, "html.parser")
title = bs_obj.select_one("div > h1")
print(title.text)
```

**Out :** 크롤링의 세계에 오신 것을 환영합니다.

## • 크롤링 사이트 확인

> 개발자 툴을 사용하여 본문 태그 찾기



<div id='article-view'>  
하위 <p> 태그

- 크롤링 사이트 확인

> 개발자 툴을 사용하여 본문 태그 찾기

```
url = "https://ai-dev.tistory.com/1"  
response = requests.get(url)  
html = response.content  
bs_obj = BeautifulSoup(html, "html.parser")  
content = bs_obj.select_one("div#article-view p")  
print(content.text)
```

**Out** : Hello, world!



• 다양한 태그 크롤링

▪ <https://ai-dev.tistory.com/2>

> 테이블

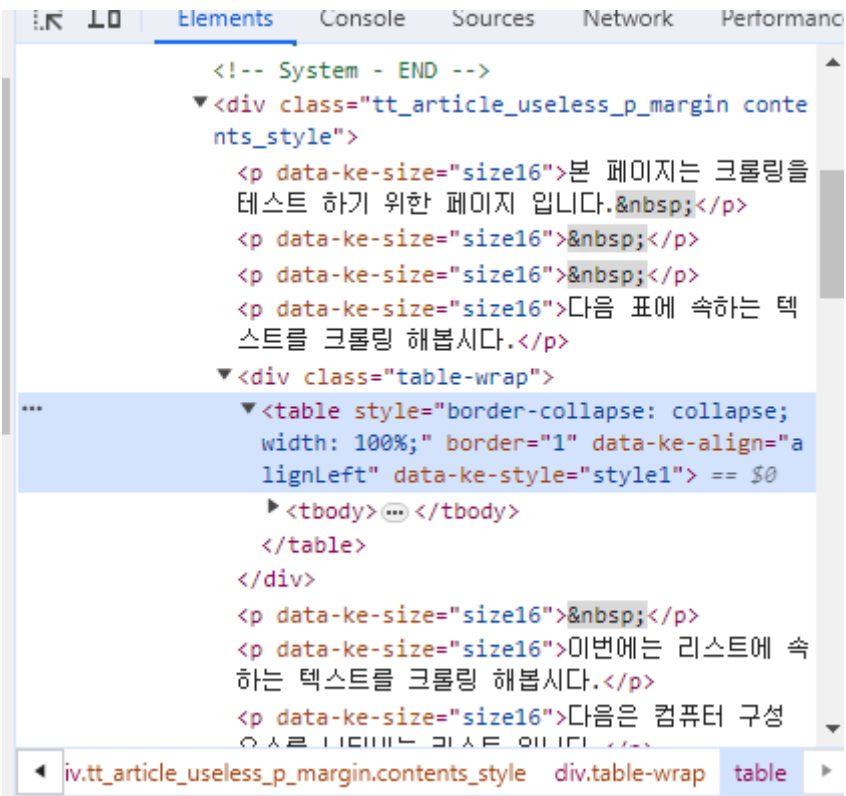
상품	색상	가격
셔츠1	빨강	20000
셔츠2	파랑	19000
셔츠3	초록	18000
바지1	검정	50000
바지2	파랑	51000

> 리스트

- 모니터
- CPU
- 메모리
- 그래픽카드
- 하드디스크
- 키보드
- 마우스

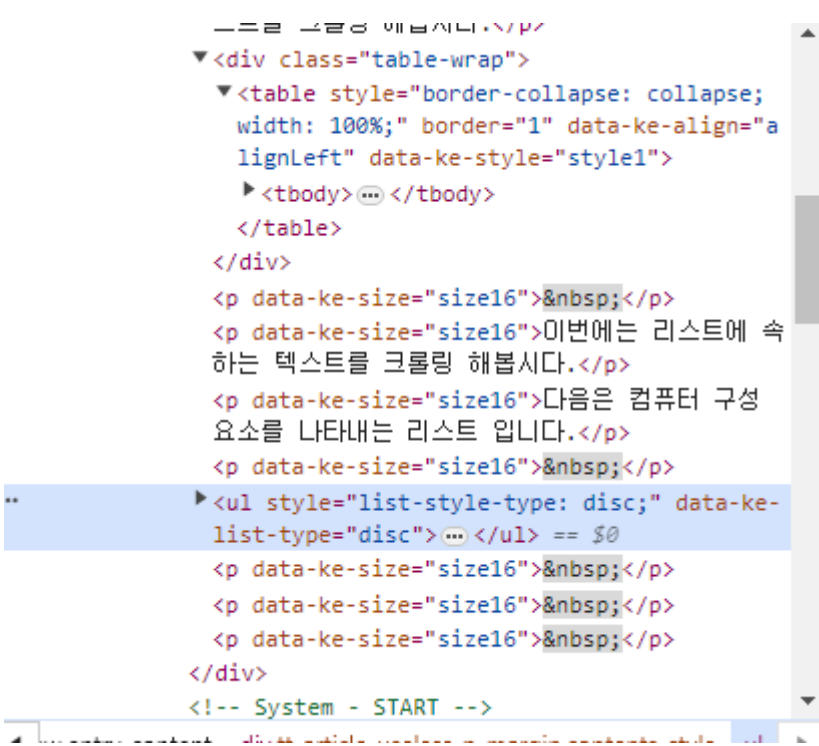
## • 다양한 태그 크롤링

> 개발자 툴을 사용하여 태그 찾기



The screenshot shows the Chrome DevTools 'Elements' panel. The DOM tree is expanded to show a `<table>` element within a `<div class="table-wrap">`. The selected `<table>` element has attributes: `style="border-collapse: collapse; width: 100%; border="1" data-ke-align="alignLeft" data-ke-style="style1"`. The breadcrumb at the bottom indicates the path: `div.tt_article_useless_p_margin.contents_style > div.table-wrap > table`.

```
<!-- System - END -->
<div class="tt_article_useless_p_margin contents_style">
  <p data-ke-size="size16">본 페이지는 크롤링을 테스트 하기 위한 페이지 입니다.</p>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16">다음 표에 속하는 텍스트를 크롤링 해봅시다.</p>
  <div class="table-wrap">
    <table style="border-collapse: collapse; width: 100%; border="1" data-ke-align="alignLeft" data-ke-style="style1">
      <tbody>
      </tbody>
    </table>
  </div>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16">이번에는 리스트에 속하는 텍스트를 크롤링 해봅시다.</p>
  <p data-ke-size="size16">다음은 컴퓨터 구성 요소를 나타내는 리스트 입니다.</p>
</div>
<!-- System - START -->
```



The screenshot shows the Chrome DevTools 'Elements' panel with a different selection. The DOM tree is expanded to show a `<ul>` element within a `<div class="table-wrap">`. The selected `<ul>` element has attributes: `style="list-style-type: disc;" data-ke-list-type="disc"`. The breadcrumb at the bottom indicates the path: `div.tt_article_useless_p_margin.contents_style > div.table-wrap > ul`.

```
<!-- System - START -->
<div class="table-wrap">
  <table style="border-collapse: collapse; width: 100%; border="1" data-ke-align="alignLeft" data-ke-style="style1">
    <tbody>
    </tbody>
  </table>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16">이번에는 리스트에 속하는 텍스트를 크롤링 해봅시다.</p>
  <p data-ke-size="size16">다음은 컴퓨터 구성 요소를 나타내는 리스트 입니다.</p>
  <ul style="list-style-type: disc;" data-ke-list-type="disc">
  </ul>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16"></p>
  <p data-ke-size="size16"></p>
</div>
<!-- System - START -->
```

## • 다양한 태그 크롤링

### > 테이블 태그 크롤링

```
url = "https://ai-dev.tistory.com/2"
response = requests.get(url)
html = response.content
bs_obj = BeautifulSoup(html, "html.parser")
table = bs_obj.select_one("table")
print(table)
```

**Out :** `<table border="1" data-ke-align="alignLeft" data-ke-style="style1" style="border-collapse: collapse; width: 100%;">`  
`<tbody>`  
`<tr>`  
`<td style="width: 33.3333%; text-align: center;">상품</td>`  
`<td style="width: 33.3333%; text-align: center;">색상</td>`  
`<td style="width: 33.3333%; text-align: center;">가격</td>`  
`</tr>`  
`<tr>`  
`<td style="width: 33.3333%; text-align: center;">셔츠1</td>`  
`<td style="width: 33.3333%; text-align: center;">빨강</td>`  
`<td style="width: 33.3333%; text-align: center;">20000</td>`  
`</tr>`  
`<tr>`  
`<td style="width: 33.3333%; text-align: center;">셔츠2</td>`  
`<td style="width: 33.3333%; text-align: center;">파랑</td>`  
`<td style="width: 33.3333%; text-align: center;">15000</td>`  
`</tr>`  
`<tr>`  
`<td style="width: 33.3333%; text-align: center;">셔츠3</td>`  
`<td style="width: 33.3333%; text-align: center;">노랑</td>`  
`<td style="width: 33.3333%; text-align: center;">12000</td>`  
`</tr>`  
`</tbody>`  
`</table>`

## • 다양한 태그 크롤링

### > 테이블 태그 크롤링

```
rows = table.select('tr')  
print(rows)
```

**Out :** [  
 <tr>  
 <td style="width: 33.3333%; text-align: center;">상품</td>  
 <td style="width: 33.3333%; text-align: center;">색상</td>  
 <td style="width: 33.3333%; text-align: center;">가격</td>  
 </tr>, <tr>  
 <td style="width: 33.3333%; text-align: center;">셔츠1</td>  
 <td style="width: 33.3333%; text-align: center;">빨강</td>  
 <td style="width: 33.3333%; text-align: center;">20000</td>  
 </tr>, <tr>  
 <td style="width: 33.3333%; text-align: center;">셔츠2</td>  
 <td style="width: 33.3333%; text-align: center;">파랑</td>  
 <td style="width: 33.3333%; text-align: center;">19000</td>  
 </tr>, <tr>  
 <td style="width: 33.3333%; text-align: center;">셔츠3</td>  
 <td style="width: 33.3333%; text-align: center;">초록</td>  
 <td style="width: 33.3333%; text-align: center;">18000</td>  
 </tr>, <tr>  
 <td style="width: 33.3333%; text-align: center;">바지1</td>  
 <td style="width: 33.3333%; text-align: center;">검정</td>  
 <td style="width: 33.3333%; text-align: center;">50000</td>  
 </tr>, <tr>  
 <td style="width: 33.3333%; text-align: center;">바지2</td>  
 <td style="width: 33.3333%; text-align: center;">파랑</td>  
 <td style="width: 33.3333%; text-align: center;">51000</td>  
 </tr>]

- 다양한 태그 크롤링

- > 테이블 태그 크롤링

```
columns = []  
column = table.select_one('tr').select('td')  
for col in column:  
    columns.append(col.text)  
print(columns)
```

**Out :** ['상품', '색상', '가격']

- 다양한 태그 크롤링

- > 테이블 태그 크롤링

```
datas = []
rows = table.select('tr:not(:first-child)')
for row in rows:
    data = []
    tds = row.select('td')
    for td in tds:
        data.append(td.text)
    datas.append(data)
print(datas)
```

**Out :** [['셔츠1', '빨강', '20000'], ['셔츠2', '파랑', '19000'], ['셔츠3', '초록', '18000'], ['바지1', '검정', '50000'], ['바지2', '파랑', '51000']]

## • 다양한 태그 크롤링

### > 리스트 태그 크롤링

```
url = "https://ai-dev.tistory.com/2"  
response = requests.get(url)  
html = response.content  
bs_obj = BeautifulSoup(html, "html.parser")  
ul = bs_obj.select_one("div > ul")  
print(ul)
```

**Out :** <ul data-ke-list-type="disc" style="list-style-type: disc;">  
 <li>모니터</li>  
 <li>CPU</li>  
 <li>메모리</li>  
 <li>그래픽카드</li>  
 <li>하드디스크</li>  
 <li>키보드</li>  
 <li>마우스</li>  
</ul>

- 다양한 태그 크롤링

- > 리스트 태그 크롤링

```
items = []  
lis = ul.select('li')  
for li in lis:  
    items.append(li.text)  
print(items)
```

**Out :** ['모니터', 'CPU', '메모리', '그래픽카드', '하드디스크', '키보드', '마우스']



## > 네이버 검색 연관검색어 크롤링하기

