

본 자료 및 영상 컨텐츠는 저작권법 제25조 2항에 의해 보호를 받습니다. 본 컨텐츠 및 컨텐츠 일부 문구등을 외부에 공개, 게시하는 것을 금지합니다. 특히 자료에 대해서는 저작권법을 엄격하게 적용하겠습니다.

Scrapy 와 Open API

Scrapy 와 프레임워크 정리

- scrapy 장점
 - 속도가 빠름
 - 다양한 기능 제공
- scrapy 단점
 - 프레임워크 사용법에 익숙해야 함

일반적으로 프레임워크 내부 코드를 이해할 필요는 없음

Scrapy와 오픈 API

Scrapy로 네이버 Open API 사용하기

- 사전 작업
 - <https://developers.naver.com/> 사이트에서 '내 애플리케이션'으로 '검색' 권한을 신청
 - Client ID 와 Client Secret 확인
 - 기존 프로젝트에, 다음 spider 추가

```
> scrapy startproject mynaverapi
> cd mynaverapi
> scrapy genspider navershopapi openapi.naver.com/v1/search/shop.json
```

- 유의 사항
 - OPEN API 사용법은 파이썬 입문과 크롤링 부트캠프에서 상세히 설명드리고 있습니다
 - 본 강의는 해상 강의를 선수강 후, 이어서 들으시는 강의로 설명드리고 있습니다
 - 최대한 다양한 기능을 보여드리고자 넣은 내용이므로, 참고로 보셔도 좋습니다

Scrapy와 오픈 API: spider 코드 작성

크롤링 주소에 추가적인 정보를 넣고자 할때 활용 가능

- scrapy.Request()에 header 정보 추가 가능

```
class NavershopapiSpider(scrapy.Spider):
    name = 'navershopapi'
    allowed_domains = ['openapi.naver.com/v1/search/shop.json']
    start_urls = ['https://openapi.naver.com/v1/search/shop.json']

    def start_requests(self):
        client_id = '나의 client id'
        client_secret = '나의 client secret'
        header_params = {'X-Naver-Client-Id': client_id,
                         'X-Naver-Client-Secret': client_secret}
        query = 'iphone'
        for url in self.start_urls:
            yield scrapy.Request(url + '?query=' + query, headers=header_params)

    def parse(self, response):
        print(response.text)
        pass
```

- start_urls 에 자동으로 http:// 와 마지막에 / 가 들어갈 수 있으므로, https:// 로 변경 및, 마지막의 / 문자는 삭제 필요

settings.py 추가 설정 이해

- Forbidden by robots.txt 에러 이해
 - Scrapy 는 크롤링시, 해당 서버의 robots.txt를 먼저 다운로드
 - 해당 파일을 기반으로 허용된 크롤링만 진행
 - 오픈 API 사용 시에도 동일 메세지 수신 가능
 - `settings.py` 파일에서 다음 설정 변경 필요

```
ROBOTSTXT_OBEY = False
```

Scrapy와 오픈 API 연습

- 사전 작업
 - <https://developers.naver.com> 사이트에서 '내 애플리케이션'으로 '검색' 권한을 신청
 - Client ID 와 Client Secret 확인
 - 기존 프로젝트에, 다음 spider 추가

```
> scrapy genspider navershopapi_item openapi.naver.com/v1/search/shop.json
```

- 해당 spider 코드에서, start_urls 에 자동으로 http:// 와 마지막에 / 가 들어갈 수 있으므로, https:// 로 변경 및, 마지막의 / 문자는 삭제 필요
- 전체 코드 업데이트 및 테스트는 별도로 공유드린 프로젝트 파일 참조

JSON 데이터의 크롤링 처리

- 전체 데이터를 유니코드로 인코딩해서 가져옴
 - `response.body_as_unicode()`
- json 라이브러리 임포트 및 `json.loads()`로 json 데이터를 파이썬 사전 데이터로 변환
- 크롤링한 JSON 데이터 처리 예

```
def parse(self, response):
    data = json.loads(response.body_as_unicode())
    for item in data['items']:
        print(item['title'])
```

데이터 후처리 예

- 가져온 데이터는 정규표현식 등, 파이썬의 다양한 기능을 활용해서, 변환 가능
- 크롤링한 데이터의 후처리 예
 - 정규표현식은 파이썬 입문과 크롤링 부트캠프 강의에서 상세히 설명드림
 - \S+ 는 white space(스페이스, 탭, 엔터 등등) 가 아닌 문자열을 의미함

```
def parse(self, response):
    data = json.loads(response.body_as_unicode())
    for item in data['items']:
        print (re.sub('<\S+>', '', item['title']))
```

참고: 정규표현식 축약 표현

정규 표현식	축약 표현	사용 예
[0-9]	\d	숫자를 찾음
[^0-9]	\D	숫자가 아닌 것을 찾음(텍스트, 특수 문자, white space(스페이스, 탭, 엔터 등등)를 찾을 때)
[\t\n\r\f\v]	\s	white space(스페이스, 탭, 엔터 등등) 문자인 것을 찾음
[^ \t\n\r\f\v]	\S	white space(스페이스, 탭, 엔터 등등) 문자가 아닌 것을 찾음(텍스트, 특수 문자, 숫자를 찾을 때)
[A-Za-z0-9]	\w	문자, 숫자를 찾음
[^A-Za-z0-9]	\W	문자, 숫자가 아닌 것을 찾음