Introduction to Scrapy

Scrapy

- 대규모 웹 크롤링을 위한 파이썬 애플리케이션 프레임워크
- 웹사이트로부터 구조화된 데이터를 효과적으로 추출, 처리, 저장하는데 필요한 모든 도구 제공
- 다양한 종류의 애플리케이션에서 활용할 수 있는 유용한 데이터 수집 도구
- 웹 스크래핑뿐만 아니라 공개 API 데이터 수집 및 범용 크롤링 도구로도 사용 가능
- 도움말 링크
  - » https://docs.scrapy.org/en/latest/

Scrapy

- 설치

```
pip install scrapy # or conda install –c conda-forge scrapy
```

- 주요 의존 패키지

패키지 이름	설명
lxml	HTML, XML 파서
parsel	lxml 기반 HTML, XML 데이터 추출 라이브러리
w3lib	URL 및 웹 페이지 인코딩 처리 유틸리티
twisted	비동기 네트워크 프레임워크
cryptography, pyOpenSSL	다양한 네트워크 수준 보안 도구

- 가상 파이썬 환경 사용 권장

Quick Example

- write quotes_spider.py

```
import scrapy

class QuotesSpider(scrapy.Spider):
    name = 'quotes'
    start_urls = [
        'http://quotes.toscrape.com/tag/humor/',
    ]

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'author': quote.xpath('span/small/text()').get(),
                'text': quote.css('span.text::text').get(),
            }

        next_page = response.css('li.next a::attr("href")').get()
        if next_page is not None:
            yield response.follow(next_page, self.parse)
```

- run command → scrapy runspider quotes_spider.py –o quotes.jl

Quick Example

- 실행 결과 (quotes.jl)


```
{"author": "Jane Austen", "text": "\u201cThe person, be it gentleman or lady, who has not plea
{"author": "Steve Martin", "text": "\u201cA day without sunshine is like, you know, night.\u20
{"author": "Garrison Keillor", "text": "\u201cAnyone who thinks sitting in church can make you
{"author": "Jim Henson", "text": "\u201cBeauty is in the eye of the beholder and it may be nec
{"author": "Charles M. Schulz", "text": "\u201cAll you need is love. But a little chocolate nc
{"author": "Suzanne Collins", "text": "\u201cRemember, we're madly in love, so it's all right
{"author": "Charles Bukowski", "text": "\u201cSome people never go crazy. What truly horrible
{"author": "Terry Pratchett", "text": "\u201cThe trouble with having an open mind, of course,
{"author": "Dr. Seuss", "text": "\u201cThink left and think right and think low and think high
{"author": "George Carlin", "text": "\u201cThe reason I talk to myself is because I\u2019m the
{"author": "W.C. Fields", "text": "\u201cI am free of all prejudice. I hate everyone equally.
{"author": "Jane Austen", "text": "\u201cA lady's imagination is very rapid; it jumps from adm
{"author": "Jane Austen", "text": "\u201cThe person, be it gentleman or lady, who has not plea
{"author": "Steve Martin", "text": "\u201cA day without sunshine is like, you know, night.\u20
{"author": "Garrison Keillor", "text": "\u201cAnyone who thinks sitting in church can make you
{"author": "Jim Henson", "text": "\u201cBeauty is in the eye of the beholder and it may be nec
{"author": "Charles M. Schulz", "text": "\u201cAll you need is love. But a little chocolate nc
{"author": "Suzanne Collins", "text": "\u201cRemember, we're madly in love, so it's all right
{"author": "Charles Bukowski", "text": "\u201cSome people never go crazy. What truly horrible
{"author": "Terry Pratchett", "text": "\u201cThe trouble with having an open mind, of course,
{"author": "Dr. Seuss", "text": "\u201cThink left and think right and think low and think high
{"author": "George Carlin", "text": "\u201cThe reason I talk to myself is because I\u2019m the
{"author": "W.C. Fields", "text": "\u201cI am free of all prejudice. I hate everyone equally.
{"author": "Jane Austen", "text": "\u201cA lady's imagination is very rapid; it jumps from adm
```

quick start summary

크롤링 웹사이트

- 구현 내용
  - » 1. Scrapy 프로젝트 만들기
  - » 2. 웹사이트에서 데이터를 추출하는 Spider 구현
  - » 3. 명령행 도구를 사용해서 추출된 데이터 저장
  - » 4. 여러 개의 링크를 이어서 크롤링하는 Spider 구현
  - » 5. Spider 전달인자 사용

프로젝트 만들기

- 프로젝트 만들기 명령

```
scrapy startproject tutorial
```

```
tutorial/
    scrapy.cfg            # deploy configuration file

    tutorial/             # project's Python module, you'll import your code from
        __init__.py

        items.py          # project items definition file

        middlewares.py    # project middlewares file

        pipelines.py      # project pipelines file

        settings.py       # project settings file

        spiders/          # a directory where you'll later put your spiders
            __init__.py
```

Spider

- Scrapy 프레임워크가 웹사이트 정보를 수집하기 위해 사용하는 사용자 정의 클래스
  - 1개 또는 다수의 사이트를 대상으로 데이터 추출, 다른 링크로 이동 등 사이트에 대한 스크랩 방법을 정의한 클래스
  - 반드시 scrapy.Spider를 상속하도록 구현
  - 최초 요청, 페이지의 링크를 추적하는 방법, 다운로드된 페이지를 파싱하는 방법 등을 정의

- 동작 구조
  - start_requests() 또는 start_urls 등에 지정된 내용에 따라 시작 페이지에 대한 요청을 보내고 응답이 도착하면 응답을 처리하기 위해 지정한 메서드 호출
  - 호출된 처리 메서드에서 선택자를 통해 데이터 추출
  - 호출된 처리 메서드는 추출한 데이터를 사용해서 데이터 객체를 반환하거나 다른 페이지에 대한 요청 객체 반환
  - 다른 페이지에 대한 요청 객체가 반환된 경우 요청 객체의 내용에 따라 요청을 수행하고 지정된 처리 메서드 호출 (재귀적 실행)
  - 데이터 객체가 반환된 경우 데이터를 데이터베이스와 같은 영구 저장소에 저장

스파이더 만들기 1

- 구현

project-path/spiders/quotes-spider.py

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"
    def start_requests(self):
        urls = [
            'http://quotes.toscrape.com/page/1/',
            'http://quotes.toscrape.com/page/2/',
        ]
        for url in urls:
            yield scrapy.Request(url=url, callback=self.parse)

    def parse(self, response):
        page = response.url.split("/")[-2]
        filename = f'quotes-{page}.html'
        with open(filename, 'wb') as f:
            f.write(response.body)
        self.log(f'Saved file {filename}')
```

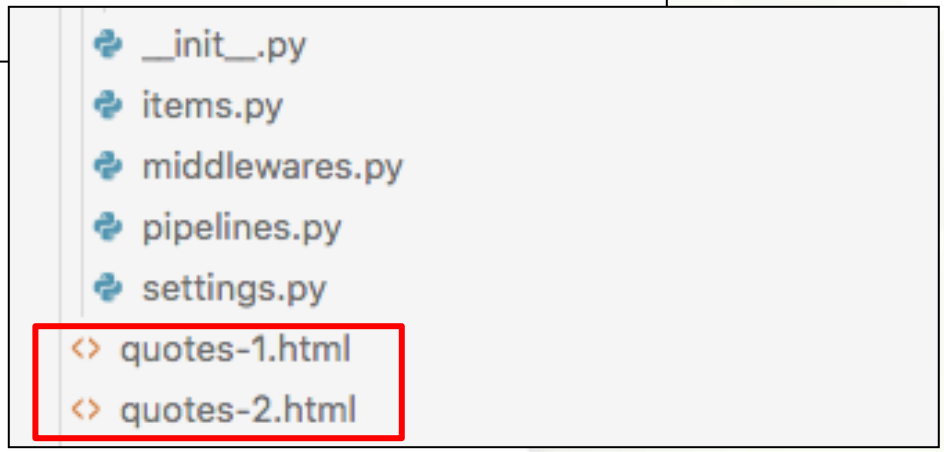
- 프로젝트 전역에서 고유한 이름
- Spider에게 웹크롤링 시작 위치 제공
- Requests 객체의 iterable 형식 반환
- 각 요청에 의해 다운로드된 응답을 처리하기 위해 Spider가 호출하는 함수
- TextResponse 형식의 전달인자를 통해 응답 정보 사용

스파이더 실행

- 실행 명령

```
scrapy crawl quotes
```

```
... (omitted for brevity)
2016-12-16 21:24:05 [scrapy.core.engine] INFO: Spider opened
2016-12-16 21:24:05 [scrapy.extensions.logstats] INFO: Crawled 0 pages (at 0 pages
2016-12-16 21:24:05 [scrapy.extensions.telnet] DEBUG: Telnet console listening on
2016-12-16 21:24:05 [scrapy.core.engine] DEBUG: Crawled (404) <GET http://quotes.t
2016-12-16 21:24:05 [scrapy.core.engine] DEBUG: Crawled (200) <GET http://quotes.t
2016-12-16 21:24:05 [scrapy.core.engine] DEBUG: Crawled (200) <GET http://quotes.t
2016-12-16 21:24:05 [quotes] DEBUG: Saved file quotes-1.html
2016-12-16 21:24:05 [quotes] DEBUG: Saved file quotes-2.html
2016-12-16 21:24:05 [scrapy.core.engine] INFO: Closing spider (finished)
...
```



Scrapy shell

- Scrapy의 명령어를 테스트 할 수 있는 대화형 실행 환경

- Scrapy shell 시작

```
scrapy shell "http://quotes.toscrape.com/page/1/"
```

```
[ ... Scrapy log here ... ]
2016-09-19 12:09:27 [scrapy.core.engine] DEBUG: Crawled (200) <GET http://quotes.t
[s] Available Scrapy objects:
[s]   scrapy     scrapy module (contains scrapy.Request, scrapy.Selector, etc)
[s]   crawler    <scrapy.crawler.Crawler object at 0x7fa91d888c90>
[s]   item       {}
[s]   request    <GET http://quotes.toscrape.com/page/1/>
[s]   response   <200 http://quotes.toscrape.com/page/1/>
[s]   settings   <scrapy.settings.Settings object at 0x7fa91d888c10>
[s]   spider     <DefaultSpider 'default' at 0x7fa91c8af990>
[s] Useful shortcuts:
[s]   shelp()           Shell help (print this help)
[s]   fetch(req_or_url) Fetch request (or URL) and update local objects
[s]   view(response)    View response in a browser
```

view(response) 명령으로 응답 결과를 브라우저에서 확인 가능

Scrapy shell

- 명령 실행

```
>>> response.css('title')
[<Selector xpath='descendant-or-self::title' data='<title>Quotes to Scrape</title>'>]
```

```
>>> response.css('title::text').getall()
['Quotes to Scrape']
```

```
>>> response.css('title').getall()
['<title>Quotes to Scrape</title>']
```

```
>>> response.css('title::text').get()
'Quotes to Scrape'
```

```
>>> response.css('title::text')[0].get()
'Quotes to Scrape'
```

```
>>> response.css('title::text').re(r'Quotes.*')
['Quotes to Scrape']
>>> response.css('title::text').re(r'Q\w+')
['Quotes']
>>> response.css('title::text').re(r'(\w+) to (\w+)')
['Quotes', 'Scrape']
```

스파이더 만들기 2

- start_urls 변수 (list 형식)
  - » start_request 메서드의 약식 버전
  - » start_request 메서드 대신 start_urls 변수에 크롤링 웹사이트 URL 목록 저장

- 구현

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"
    start_urls = [
        'http://quotes.toscrape.com/page/1/',
        'http://quotes.toscrape.com/page/2/',
    ]

    def parse(self, response):
        page = response.url.split("/")[-2]
        filename = f'quotes-{page}.html'
        with open(filename, 'wb') as f:
            f.write(response.body)
```

- Spider에게 웹크롤링 시작 위치 제공
- Default 버전의 start_requests() 메서드에서 사용

스파이더 만들기 3

- parse 메서드에서 데이터 읽기
  - » css("선택자") 메서드를 사용해서 HTML 문서 요소 탐색
  - » yield 반환 구문으로 데이터 반환

- 구현

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"
    start_urls = [
        'http://quotes.toscrape.com/page/1/',
        'http://quotes.toscrape.com/page/2/',
    ]

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').get(),
                'author': quote.css('small.author::text').get(),
                'tags': quote.css('div.tags a.tag::text').getall(),
            }
```

- 크롤링한 데이터를 객체 형식으로 반환

스파이더 만들기 3

- 실행 결과

```
2016-09-19 18:57:19 [scrapy.core.scraper] DEBUG: Scraped from <200 http://quotes.toscrape.com/
{'tags': ['life', 'love'], 'author': 'André Gide', 'text': '“It is better to be hated for what
2016-09-19 18:57:19 [scrapy.core.scraper] DEBUG: Scraped from <200 http://quotes.toscrape.com/
{'tags': ['edison', 'failure', 'inspirational', 'paraphrased'], 'author': 'Thomas A. Edison',
```

크롤링한 데이터 저장

- 실행 명령
  - » json 형식으로 저장 (-O : overwrite, -o : append)

```
scrapy crawl quotes -O quotes.json
```

  - » json lines 형식으로 저장

```
scrapy crawl quotes -o quotes.jl
```

스파이더 만들기 4

- 재귀적으로 다른 사이트를 탐색하는 스파이더
  - » 크롤링 페이지에 포함된 다른 페이지의 링크를 재귀적으로 크롤링

- 구현
  - » 페이지에 포함된 다른 페이지의 링크 추출

```
<ul class="pager">
    <li class="next">
        <a href="/page/2/">Next <span aria-hidden="true">&rarr;</span></a>
    </li>
</ul>
```

```
>>> response.css('li.next a::attr(href)').get()
'/page/2/'
```

```
>>> response.css('li.next a').attrib['href']
'/page/2/'
```

스파이더 만들기 4

- 구현
  - » parse 메서드가 Request 객체를 yield 반환하도록 구현

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"
    start_urls = [
        'http://quotes.toscrape.com/page/1/',
    ]

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').get(),
                'author': quote.css('small.author::text').get(),
                'tags': quote.css('div.tags a.tag::text').getall(),
            }

        next_page = response.css('li.next a::attr(href)').get()
        if next_page is not None:
            next_page = response.urljoin(next_page)
            yield scrapy.Request(next_page, callback=self.parse)
```

스파이더 만들기 4

- 구현
  - » parse 메서드가 response.follow 함수의 결과 값을 yield 반환하도록 구현

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"
    start_urls = [
        'http://quotes.toscrape.com/page/1/',
    ]

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').get(),
                'author': quote.css('span small::text').get(),
                'tags': quote.css('div.tags a.tag::text').getall(),
            }

        next_page = response.css('li.next a::attr(href)').get()
        if next_page is not None:
            yield response.follow(next_page, callback=self.parse)
```

스파이더 만들기 4

- 구현
  - » response.follow 함수의 결과 값을 반환할 때 링크 문자열 또는 하이퍼링크 객체를 반환하도록 구현 가능

```
for href in response.css('ul.pager a::attr(href)'):
    yield response.follow(href, callback=self.parse)
```

```
for a in response.css('ul.pager a'):
    yield response.follow(a, callback=self.parse)
```

```
anchors = response.css('ul.pager a')
yield from response.follow_all(anchors, callback=self.parse)
```

```
yield from response.follow_all(css='ul.pager a', callback=self.parse)
```

스파이더 만들기 4

- 구현
  - » 링크 반환 메서드와 데이터 반환 메서드를 구분해서 구현

```
import scrapy


class AuthorSpider(scrapy.Spider):
    name = 'author'

    start_urls = ['http://quotes.toscrape.com/']

    def parse(self, response):
        author_page_links = response.css('.author + a')
        yield from response.follow_all(author_page_links, self.parse_author)

        pagination_links = response.css('li.next a')
        yield from response.follow_all(pagination_links, self.parse)

    def parse_author(self, response):
        def extract_with_css(query):
            return response.css(query).get(default='').strip()

        yield {
            'name': extract_with_css('h3.author-title::text'),
            'birthdate': extract_with_css('.author-born-date::text'),
            'bio': extract_with_css('.author-description::text'),
        }
```

Spider 명령 행 전달 인자

- Spider 명령 실행 시 전달 인자 지정 가능

```
scrapy crawl quotes -O quotes-humor.json -a tag=humor
```

```
import scrapy


class QuotesSpider(scrapy.Spider):
    name = "quotes"

    def start_requests(self):
        url = 'http://quotes.toscrape.com/'
        tag = getattr(self, 'tag', None)
        if tag is not None:
            url = url + 'tag/' + tag
        yield scrapy.Request(url, self.parse)

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').get(),
                'author': quote.css('small.author::text').get(),
            }

        next_page = response.css('li.next a::attr(href)').get()
        if next_page is not None:
            yield response.follow(next_page, self.parse)
```

Scrapy 명령 행 도구

- Scrapy 명령 행 도구를 사용해서 scrapy의 동작 제어
- 명령 실행 형식
  - » 프로젝트 위치와 관계 없이 실행하는 명령

```
Scrapy X.Y - no active project

Usage:
  scrapy <command> [options] [args]

Available commands:
  crawl         Run a spider
  fetch         Fetch a URL using the Scrapy downloader
[...]
```

  - » 프로젝트 내부에서 실행하는 명령

```
Scrapy X.Y - project: myproject

Usage:
  scrapy <command> [options] [args]

[...]
```

Scrapy 명령 행 도구

- 전역 명령 종류

명령	설명
startproject	Scrapy 프로젝트 만들기
genspider	현재 경로에 Spider 만들기
settings	Scrapy 설정 반환
runspider	지정된 Spider 실행
shell	Scrapy shell 실행
fetch	Scrapy Downloader를 사용해서 주어진 URL의 요청에 대한 응답을 표준 출력 장치로 출력
view	주어진 URL의 요청에 대한 응답을 브라우저에 표시
version	Scrapy 버전 표시
bench	Benchmark 테스트 실행

Scrapy 명령 행 도구

- 프로젝트 지역 명령 종류

명령	설명
crawl	Spider를 사용해서 크롤링 시작
check	Spider 크롤링 유효성 검사
list	실행 가능한 Spider 목록 반환
edit	지정된 Spider 편집
parse	주어진 URL의 요청에 대한 응답을 지정된 메서드를 사용해서 Parsing

Items

- 수집된 데이터를 저장하는 객체
- Key – Value 쌍으로 정의되는 객체
- 종류

타입	설명
Dictionary	파이썬 dict 객체
Item	dict와 유사하지만 추가 속성을 지원하는 객체
Dataclass	필드 이름으로 Item 클래스 생성
attr.s	필드 이름으로 Item 클래스 생성

Items 구현

```python
import scrapy

class Product(scrapy.Item):
    name = scrapy.Field()
    price = scrapy.Field()
    stock = scrapy.Field()
    tags = scrapy.Field()
    last_updated = scrapy.Field(serializer=str)
```

```
>>> product = Product(name='Desktop PC', price=1000)
>>> print(product)
Product(name='Desktop PC', price=1000)
```

```
>>> product['name']
Desktop PC
>>> product.get('name')
Desktop PC
```

```
>>> product['last_updated']
Traceback (most recent call last):
    ...
KeyError: 'last_updated'
```

Items 구현

```python
import scrapy

class Product(scrapy.Item):
    name = scrapy.Field()
    price = scrapy.Field()
    stock = scrapy.Field()
    tags = scrapy.Field()
    last_updated = scrapy.Field(serializer=str)
```

```
>>> product['last_updated'] = 'today'
>>> product['last_updated']
today
```

```
>>> product['lala'] = 'test' # setting unknown field
Traceback (most recent call last):
    ...
KeyError: 'Product does not support field: lala'
```

Item Pipeline

- Spider가 데이터를 수집해서 Item에 저장해서 반환하면 Pipeline에 전달

- Pipeline은 약속된 재정의 메서드가 포함된 파이썬 클래스

함수	설명
process_item(self, item, spider)	Spider에서 Item을 반환할 때마다 호출
open_spider(self, spider)	Spider가 Open 될 때 호출
close_spider(self, spider)	Spider가 Close 될 때 호출
from_crawler	사용자 정의 Pipeline 구현

- 사용 사례
  - » HTML 데이터 정리
  - » 데이터 유효성 검사
  - » 데이터 중복 검사
  - » 데이터저장소에 데이터 저장

Item Pipeline 구현

```python
class MongoPipeline:

    collection_name = 'scrapy_items'

    def __init__(self, mongo_uri, mongo_db):
        self.mongo_uri = mongo_uri
        self.mongo_db = mongo_db

    @classmethod
    def from_crawler(cls, crawler):
        return cls(
            mongo_uri=crawler.settings.get('MONGO_URI'),
            mongo_db=crawler.settings.get('MONGO_DATABASE', 'items')
        )

    def open_spider(self, spider):
        self.client = pymongo.MongoClient(self.mongo_uri)
        self.db = self.client[self.mongo_db]

    def close_spider(self, spider):
        self.client.close()

    def process_item(self, item, spider):
        self.db[self.collection_name].insert_one(ItemAdapter(item).asdict())
        return item
```

Pipeline 활성화

- Settings.py 파일에 ITEM_PIPELINES 목록에 등록

```
ITEM_PIPELINES = {
    'myproject.pipelines.PricePipeline': 300,
    'myproject.pipelines.JsonWriterPipeline': 800,
}
```

» 0 ~ 1000 범위의 숫자 지정 (실행 우선 순위)

» 낮은 숫자 → 높은 숫자 순서로 실행