데이터 크롤링과 정제

4장. 웹 크롤링 모델

목차

- 4.2 다양한 웹사이트 레이아웃 다루기
  - brookings.edu 레이아웃 구성
  - oreilly.com 레이아웃 구성
  - reuters.com 레이아웃 구성

- 4.3 크롤러 구성
  - 검색을 통한 사이트 크롤링
  - 링크를 통한 사이트 크롤링
  - 여러 페이지 유형 크롤링

4.2 다양한 웹사이트 레이아웃 다루기

- 구문 분석 기능
  - 제목 요소를 선택하고 제목 텍스트 추출
  - 기사의 주요 콘텐츠 선택
  - 다른 필요한 콘텐츠 선택

- 3개의 서로 다른 웹사이트의 콘텐츠 구성 분석
  - brookings.edu (브루킹스 연구소)
  - oreilly.com (출판사)
  - reuters.com (언론사)

brookings.edu 콘텐츠 구성

- brookings.edu 웹사이트
  - 제목 텍스트 추출: <h1>태그

FUTURE DEVELOPMENT

Delivering inclusive urban access: 3 uncomfortable truths

```
<h1 class="report-title">Delivering inclusive urban access: 3
uncomfortable truths</h1>
```

  - 기사의 콘텐트 추출:

The past few decades have been filled with a deep optimism about the role of cities and suburbs across the world. These engines of economic growth host a majority of world population, are major drivers of economic innovation, and have created pathways to opportunities for untold amounts of people.

```
<div class="post-body post-body-enhanced" itemProp="articleBody">
    <p>The past few decades have been filled with a deep optimism about the role
of cities and suburbs across the world. These engines of economic growth host a
majority of world population, are major drivers of economic innovation, and have
created pathways to opportunities for untold amounts of people.</p>
```

https://www.brookings.edu/blog/future-development/2018/01/26/delivering-inclusive-urban-access-3-uncomfortable-truths/

예제 1: 소스 코드

```
import requests
from bs4 import BeautifulSoup

class Content:
    def __init__(self, url, title, body):
        self.url = url
        self.title = title
        self.body = body

def getPage(url):
    req = requests.get(url)
    return BeautifulSoup(req.text, 'html.parser')
```

입력 파라미터를 이용하여 Content 객체 생성

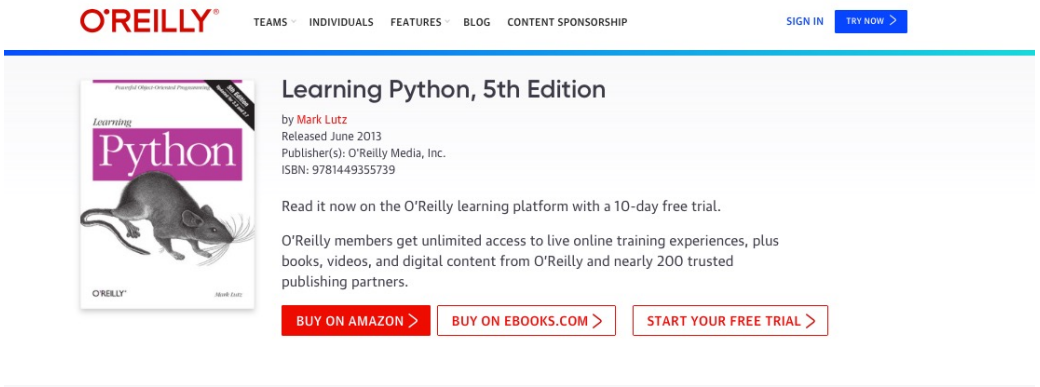
```
def scrapeBrookings(url):
    bs = getPage(url)
    title = bs.select_one('h1').text
    body = bs.select_one('div.post-body').text
    return Content(url, title, body)
```

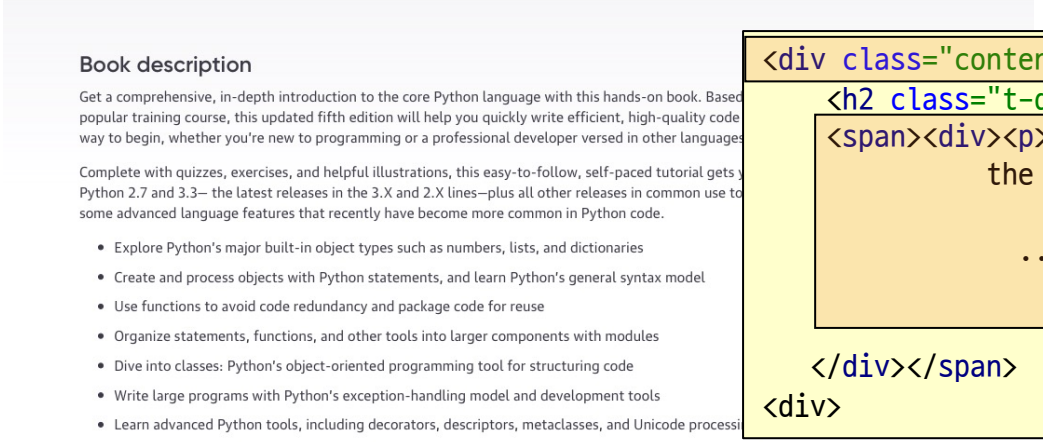
```
def scrapeBrookings(url):
    bs = getPage(url)
    title = bs.find('h1').text
    body = bs.find('div', {'class': 'post-body'}).text
    return Content(url, title, body)
```

```
url = 'https://www.brookings.edu/blog/future-development/2018/01/26/delivering-inclusive-urban-access-3-uncomfortable-truths/’

content = scrapeBrookings(url)
print('Title: {}'.format(content.title))
print('URL: {}\n'.format(content.url))
print(content.body)
```

O'REILLY 콘텐츠 구성

- 사이트 주소
  - http://shop.oreilly.com/product/0636920028154.do






```
<div class="content">
    <h2 class="t-description-heading">Book description</h2>
    <span><div><p>Get a comprehensive, in-depth introduction to
                the core Python language with this hands-on book.

                ...

   </div></span>
<div>
```

Reuter 콘텐츠 구성

- 사이트 주소
  - https://www.reuters.com/article/us-usa-epa-pruitt-idUSKBN19W2D0

Environmental Protection Agency Administrator Scott Pruitt speaks during an interview for Reuters at his office in Washington, U.S., July 10, 2017. REUTERS/Yuri Gripas

The move comes as the administration of President Don
back a slew of Obama-era regulations limiting carbon di
fossil fuels, and begins a withdrawal from the Paris Clim
pact to stem planetary warming through emissions cuts.

"There are lots of questions that have not been asked an
climate change)," EPA Administrator Scott Pruitt told R
on Monday.

"Who better to do that than a group of scientists... getti
robust discussion for all the world to see," he added wit
scientists would be chosen.

공통된 class 속성값을 가짐

```
<p class="Paragraph-paragraph-2Bgue ArticleBody-para-TD_9x">The move
     comes as the administration of President Donald Trump
                ...
</p>
<p class="Paragraph-paragraph-2Bgue ArticleBody-para-TD_9x">
    "There are lots of questions that have not been asked and answered (about
                ...
</p>
<p class="Paragraph-paragraph-2Bgue ArticleBody-para-TD_9x">
    "Who better to do that than a group of scientists...
                ...
</p>
```

예제 2: 소스 코드 #1

```
import requests
from bs4 import BeautifulSoup
import time

class Content:
    """
    글/페이지 전체에 사용할 기반 클래스
    """
    def __init__(self, url, title, body):
        self.url = url
        self.title = title
        self.body = body

    def print(self):
        print('URL: {}'.format(self.url))
        print('TITLE: {}'.format(self.title))
        print('BODY:\n{}'.format(self.body))
        print()

class Website:
    """
    웹사이트 구조에 관한 정보를 저장할 클래스
    """
    def __init__(self, name, url, titleTag, bodyTag):
        self.name = name
        self.url = url
        self.titleTag = titleTag
        self.bodyTag = bodyTag
```

예제 2: 소스 코드 #2

```
class Crawler:
    def getPage(self, url):
        try:
            req = requests.get(url)
            time.sleep(2) # 페이지 요청 후 응답을 기다릴 시간
        except requests.exceptions.RequestException:
            return None
        return BeautifulSoup(req.text, 'html.parser')

    def safeGet(self, pageObj, selector):
        """
        BeautifulSoup 객체와 선택자를 받아 콘텐츠 문자열을 추출하는 함수
        """
        selectedElems = pageObj.select(selector)
        if selectedElems is not None and len(selectedElems) > 0:
            return '\n'.join([elem.get_text() for elem in selectedElems])
        else:
            return ''

    def parse(self, site, url):
        """
        URL을 받아 콘텐츠를 추출함
        """
        bs = self.getPage(url)
        if bs is not None:
            title = self.safeGet(bs, site.titleTag)
            body = self.safeGet(bs, site.bodyTag)
            if title != '' and body != '':
                content = Content(url, title, body)
                print('-' * 100)
                content.print()
```

str.join(리스트)
- 리스트의 문자열을 합침

각 웹사이트 마다 다른 titleTag, bodyTag를 사용하여 크롤링

예제 2: 소스 코드 #3

```
crawler = Crawler()

siteData = [
    ['O\'Reilly Media', 'http://oreilly.com ' , 'h1', 'div.content > div.metadata'],
    ['Reuters', 'http://reuters.com ' ,          'h1', 'p.Paragraph-paragraph-2Bgue'],
    ['Brookings', 'http://www.brookings.edu ' , 'h1', 'div.post-body']
]

websites = []
for row in siteData:
    websites.append(Website(row[0], row[1], row[2], row[3]))

crawler.parse(websites[0], 'http://shop.oreilly.com/product/0636920028154.do')
crawler.parse(websites[1], 'http://www.reuters.com/article/us-usa-epa-pruitt-idUSKBN19W2D0')
crawler.parse(websites[2],
    'https://www.brookings.edu/blog/techtank/2016/03/01/idea-to-retire-old-methods-of-policy-
education/')
```

해당 웹사이트의 Content가 있는 tag들

교재 내용과 다름

예제 2: 소스 코드 #3

해당 웹사이트의 Content가 있는 tag들

```
crawler = Crawler()

siteData = [
    ['O\'Reilly Media', 'http://oreilly.com', 'h1', 'div.content > span'],
    ['Reuters', 'http://reuters.com ', 'h1’, 'p.Paragraph-paragraph-2Bgue'],
    ['Brookings', 'http://www.brookings.edu', 'h1’, 'div.post-body']
]

websites = []

for row in siteData:
    websites.append(Website(row[0], row[1], row[2], row[3]))

crawler.parse(websites[0], 'http://shop.oreilly.com/product/0636920028154.do')

crawler.parse(websites[1], 'http://www.reuters.com/article/us-usa-epa-pruitt-idUSKBN19W2D0')

crawler.parse(websites[2],
    'https://www.brookings.edu/blog/techtank/2016/03/01/idea-to-retire-old-methods-of-
     policy-education/')
```

교재 내용과 다름

```
URL: http://shop.oreilly.com/product/0636920028154.do
TITLE: Learning Python, 5th Edition
BODY:

Learning Python, 5th Edition
by
Released
Publisher(s):
ISBN: None
```

4.3 크롤러 구성

- 이전 예제 문제점
  - 웹사이트의 유연성
  - 첫 번째 예제
    - 각 웹사이트에 필요에 따라 HTML을 선택하고 구문 분석해야 됨
  - 두 번째 예제
    - 각 웹사이트에 대상 필드가 존재해야 됨
    - 데이터를 추출할 수 있음
    - 각 대상 필드에 고유한 CSS 선택자가 있어야함
- 개선된 프로그램
  - 자동으로 링크를 수집하고 데이터를 검색
  - 확장성이 있는 웹 크롤러 구성

4.3.1 검색을 통한 사이트 크롤링

- 검색을 통한 크롤링
  - 웹 페이지의 내부 링크 및 외부 링크를 검색
  - 해당 링크(내부, 외부)를 사용하여 사이트 전체를 크롤링

- 검색 방법
  - URL에 검색어를 삽입해서 검색 결과를 얻음

```
http://example.com?search=검색어
```

  - 링크 목록 확인

```
<span class="result">
```

  - 결과 링크의 속성 저장(절대 URL, 상대 URL)
    - 절대 URL: `http://example.com/articles/page.html`
    - 상대 URL: `/articles/page.html`

4.3.1 예제 #1

```
class Content:
    def __init__(self, topic, url, title, body):
        self.topic = topic
        self.url = url
        self.title = title
        self.body = body

    def print(self):
        print('New article found for topic: {}'.format(self.topic))
        print('URL: {}'.format(self.url))
        print('TITLE: {}'.format(self.title))
        print('BODY:\n {}'.format(self.body))

class Website:
    def __init__(self, name, url, searchUrl, resultListing, resultUrl,
                 absoluteUrl, titleTag, bodyTag):
        self.name = name
        self.url = url
        self.searchUrl = searchUrl # URL에 검색어 추가
        self.resultListing = resultListing # 각 결과에 대한 정보 저장
        self.resultUrl = resultUrl # 결과에서 정확한 URL을 추출할 때 사용
        self.absoluteUrl = absoluteUrl # 절대 경로인지, 상대 경로인지 구분
        self.titleTag = titleTag
        self.bodyTag = bodyTag
```

4.3.1 예제 #2

```
import requests
from bs4 import BeautifulSoup

class Crawler:
    def getPage(self, url):
        try:
            req = requests.get(url)
        except requests.exceptions.RequestException:
            return None
        return BeautifulSoup(req.text, 'html.parser')

    def safeGet(self, pageObj, selector):
        childObj = pageObj.select(selector)
        if childObj is not None and len(childObj) > 0:
            return childObj[0].get_text()
        else:
            return ' '

    def getAllBody(self, pageObj, selector):
        # 해당 tag를 가지는 모든 내용을 출력함
        childObj = pageObj.select(selector)
        bodyText = ""
        if childObj is not None:
            for i in range (len(childObj)):
                bodyText = bodyText + childObj[i].get_text() + '\n '
            return bodyText
        else:
            return ''
```

검색된 모든 기사 내용 중 첫번째 항목만 출력

검색된 모든 기사 내용 출력

4.3.1 예제 #3

```
def search(self, topic, site):
    # site: Website 객체
    print('searchUrl+topic:', site.searchUrl + topic)

    bs = self.getpage(site.searchUrl + topic)
    searchResults = bs.select(site.resultListing)

    for result in searchResults:
        url = result.select(site.resultUrl)[0].attrs['href']
        if(site.absoluteUrl):
            bs = self.getPage(url)
        else:
            bs = self.getPage(self.url + url)
        if bs is None:
            print('Something was wrong with that page or URL. Skipping')
            return
        title = self.safeGet(bs, site.titleTag)
        #body = self.safeGet(bs, site.bodyTag) # 첫 번째 paragraph만 출력
        body = self.getAllBody(bs, site.bodyTag) # 전체 기사 출력

        if title != '' and body != '':
            content = Content(topic, url, title, body)
            content.print()
```

절대경로 사용 여부에 따라 사용 url이 달라짐

4.3.1 예제 #4

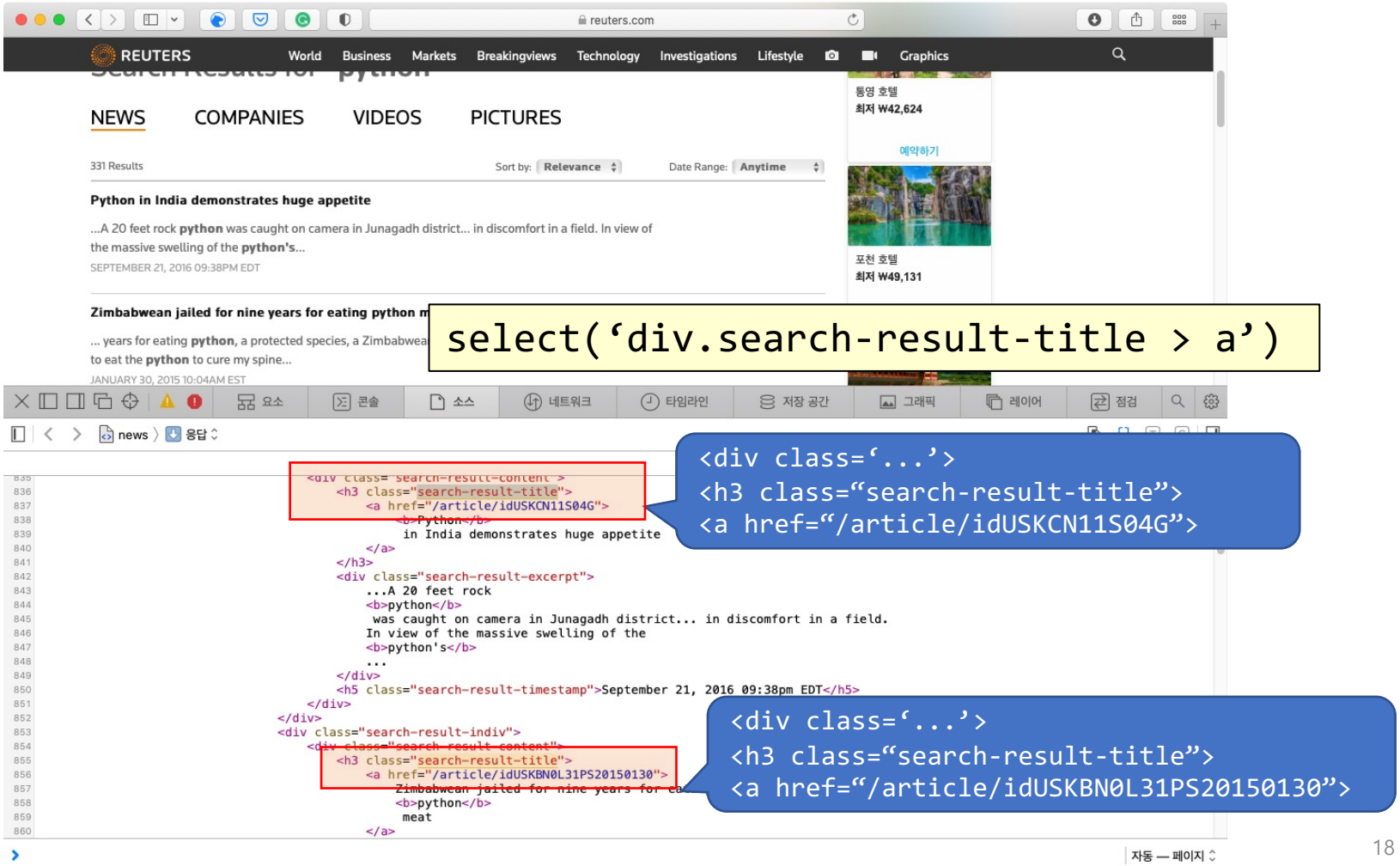
```
crawler = Crawler()

siteData1 = [
    ['Reuters',                                # Website.name
     'http://reuters.com',                     # Website.url
     'http://www.reuters.com/search/news?blob=', # Website.searchUrl: 검색을 위한 URL
     'div.search-result-content',  # Website.resultListing: 검색 결과에 대한 정보
     'h3.search-result-title > a', # Website.resultUrl: 결과에서 URL을 추출할 때 사용할 태그
     False,                          # Website.absoluteUrl 사용 여부
     'h1',                           # Website.titleTag
     'p.Paragraph-paragraph-2Bgue']  # Website.bodyTag
]

sites = []
for row in siteData1:
    sites.append(Website(row[0], row[1], row[2], row[3],
                         row[4], row[5], row[6], row[7]))

topics = ['python']
for topic in topics:
    print('GETTING INFO ABOUT: ' + topic)
    for targetSite in sites:
        crawler.search(topic, targetSite)
```

topics: 검색 항목 리스트 지정

4.3.1 예제 실행 1단계: 검색 결과 분석

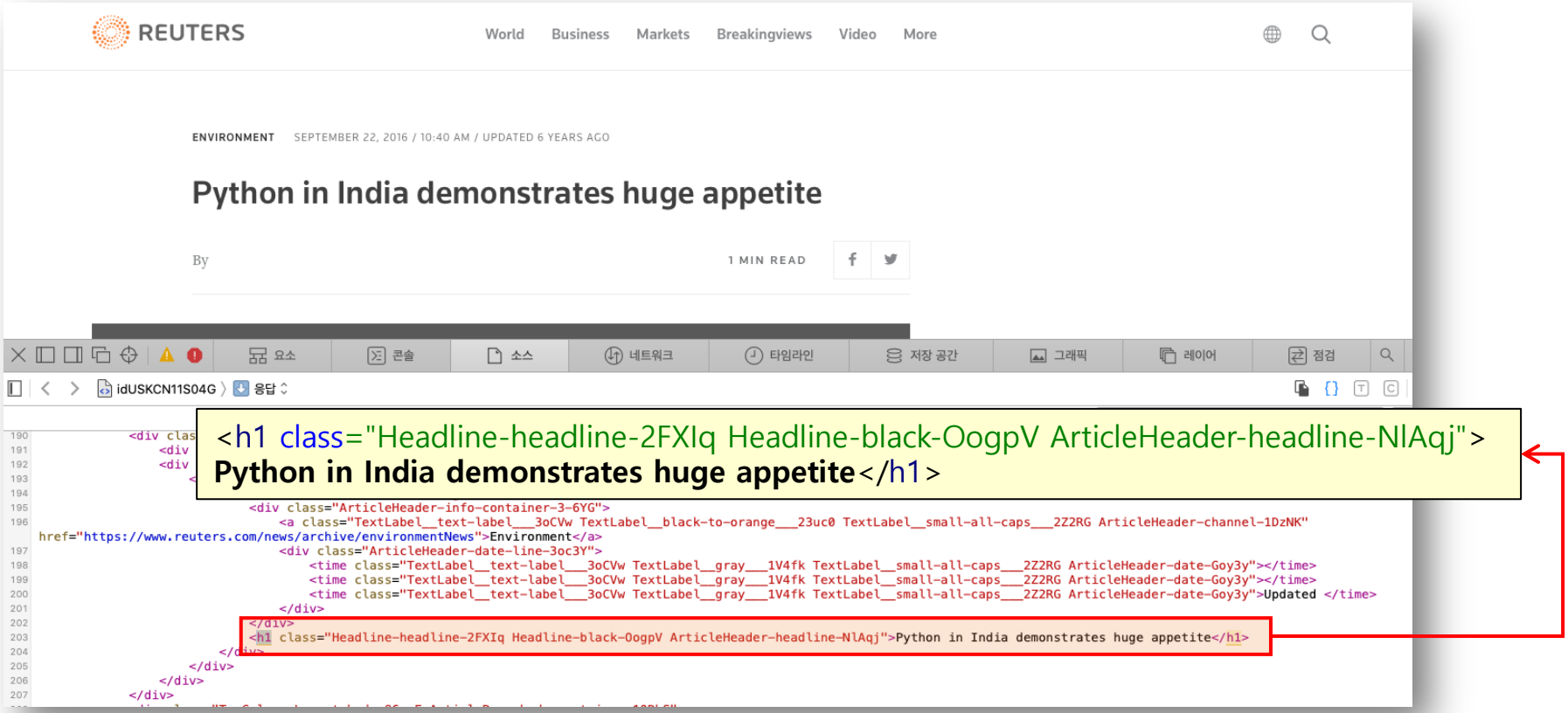
- 검색 결과에서 href 링크만 추출

  http://www.reuters.com/search/news?blob=python



4.3.1 예제 실행 2단계: 기사 제목 추출

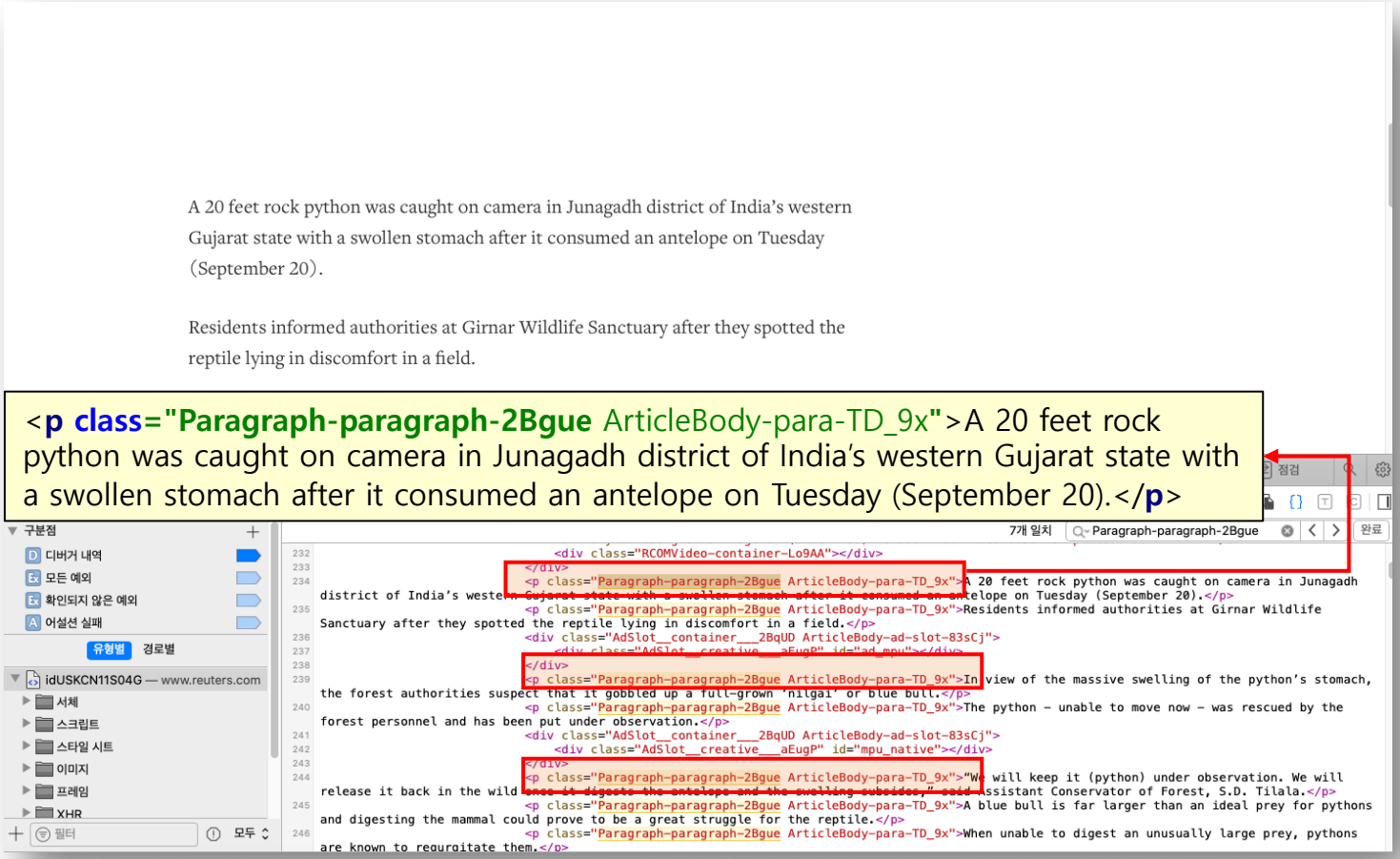
- 해당 URL로 이동
  - https://www.reuters.com/article/idUSKCN11S04G
- <h1> 태그 추출
  - `title = self.safeGet(bs, site.titleTag) -> bs.select('h1')`



4.3.1 예제 실행 3단계: 기사 내용 추출

- 기사 전체 내용 추출
  - 기사 내용

```
childObj = pageObj.select('p.Paragraph-paragraph-2Bgue' )
```



4.3.1 예제: 실행 결과

```
GETTING INFO ABOUT: python
searchUrl+topic: http://www.reuters.com/search/news?blob=python
------------------------------------------------------------
searchResult len:  10
전체 url: http://reuters.com/article/idUSKCN11S04G
------------------------------------------------------------
New article found for topic: python
URL: /article/idUSKCN11S04G
TITLE: Python in India demonstrates huge appetite
BODY:
 A 20 feet rock python was caught on camera in Junagadh district of India's western Gujarat state with a swollen stomach after it consumed an antelope on Tuesday (September 20).
Residents informed authorities at Girnar Wildlife Sanctuary after they spotted the reptile lying in discomfort in a field.
In view of the massive swelling of the python's stomach, the forest authorities suspect that it gobbled up a full-grown 'nilgai' or blue bull.
The python - unable to move now - was rescued by the forest personnel and has been put under observation.
"We will keep it (python) under observation. We will release it back in the wild once it digests the antelope and the swelling subsides," said Assistant Conservator of Forest, S.D. Tilala.
A blue bull is far larger than an ideal prey for pythons and digesting the mammal could prove to be a great struggle for the reptile.
When unable to digest an unusually large prey, pythons are known to regurgitate them.

------------------------------------------------------------
전체 url: http://reuters.com/article/idUSKBN0L31PS20150130
------------------------------------------------------------
New article found for topic: python
URL: /article/idUSKBN0L31PS20150130
TITLE: Zimbabwean jailed for nine years for eating python meat
BODY:
 ...
```

4.3.2 링크를 통한 크롤링

- 링크를 통한 크롤링
  - 특정 URL 패턴과 일치하는 모든 링크를 따라감
  - 특정 검색 결과나 페이지에 국한되지 않음
  - 사이트 전체에서 데이터를 수집하는 프로젝트에 활용

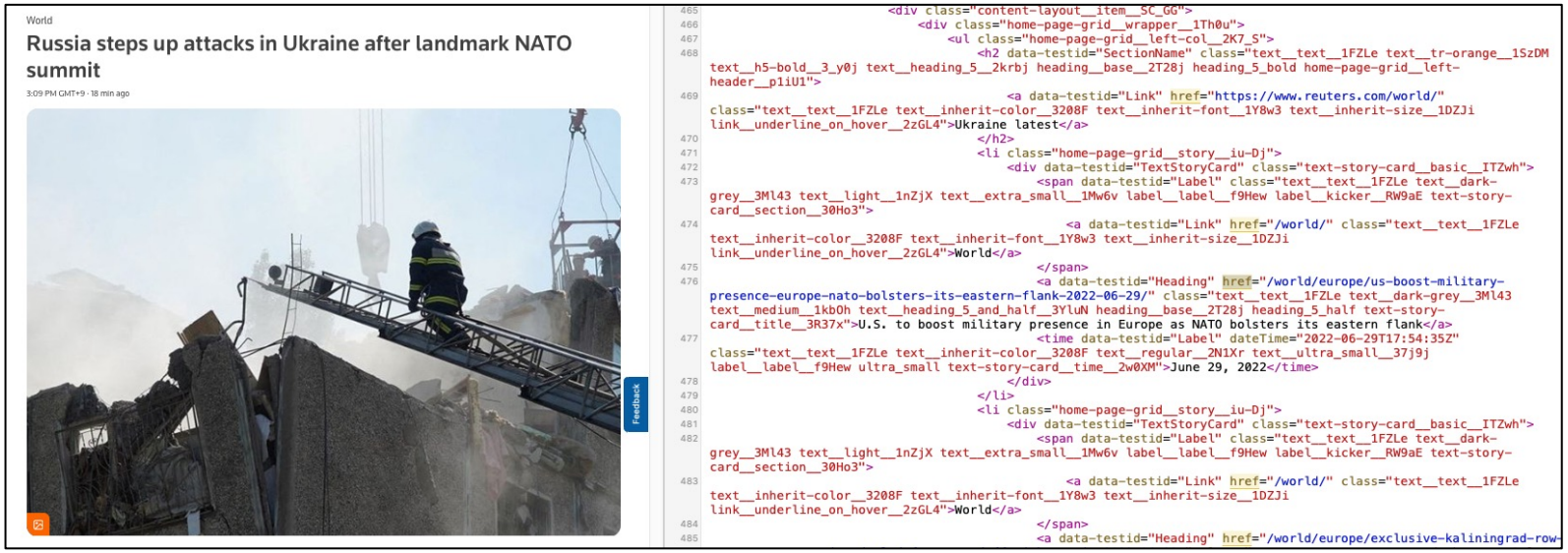
4.3.2 링크를 통한 크롤링

- 로이터통신 홈페이지 링크 패턴 분석
  - 소스 보기에서 "href" 검색
  - 여러 종류의 <a> 태그

```
<a data-testid="Heading" href=”...” >
```

```
<a data-testid=”Link" href=”...” >
```

```
<div class="media-story-card__placement-container__1R55-">
<a href=”...”>
...
</div>
```



4.3.2 링크를 통한 크롤링

- <a> 태그 중 특정 속성값 가져오기
  - data-testid 속성 값이 'Heading', 'Link'인 url 링크 검색

```
import requests
from bs4 import BeautifulSoup

url = 'https://www.reuters.com'
link_list = []
req = requests.get(url)
soup = BeautifulSoup(req.text, 'html.parser')

data_testid_links = soup.find_all('a',
                                  attrs={'data-testid' : ['Heading', 'Link']})

i = 0
for link in data_testid_links:
    if link['href'] not in link_list:
        link_list.append(link['href'])
        print('[{0:4}]: {1}'.format(i, link['href']))
        i += 1

print('link_list 길이:', len(link_list))
```

data-testid 속성값이 'Heading' 또는 'Link'인 href 값을 모두 검색

중복되지 않은 URL만 리스트에 추가

```
[   0]: https://www.reuters.com/world/
[   1]: /world/
[   2]: /world/europe/us-boost-military-presence-europe-nato-bolsters-its-eastern-flank-2022-06-29/
[   3]: /world/europe/putin-still-wants-most-ukraine-war-outlook-grim-us-intelligence-chief-2022-06-29/
. . .
```

4.3.2 예제 소스: 데이터 추출 검색어

- URL 링크 중 아래의 속성인 링크만 추출

```
<div class="media-story-card__placement-container__1R55-">
<a href=”...”>
...
</div>
```

URL 추출을 위한 검색어로 사용

- 해당 링크로 이동한 다음, 기사 내용 추출
  - 검색 내용: `class` 속성이 “`text__text__`”를 포함
    - https://www.reuters.com/markets/europe/chinas-june-factory-services-activity-expands-first-time-four-months-2022-06-30/

```
<p data-testid="primary-image-caption" id="primary-image-caption"
class="text__text__1FZLe text__medium-grey__3A_RT text__regular__2N1Xr
text__small__1kCg2 body__base__22dCE body__caption__3L8vY article-
body__p[illegible]n__WHVnb">
A worke[illegible]e steel rim at a factory manufacturing sports
equipment in Hangzhou, Zhejiang province, China September 2, 2019. China
Daily via REUTERS
</p>
```

기사 내용 추출을 위한 검색어로 사용

4.3.2 예제 소스 코드 #1

```
import requests.exceptions
from bs4 import BeautifulSoup
import re

class Website:

    def __init__(self, name, url, targetPattern, absoluteUrl, titleTag, bodyTag):
        self.name = name
        self.url = url
        self.targetPattern = targetPattern
        self.absoluteUrl = absoluteUrl
        self.titleTag = titleTag
        self.bodyTag = bodyTag

class Content:

    def __init__(self, url, title, body):
        self.url = url
        self.title = title
        self.body = body

    def print(self):
        print('[URL]: {}'.format(self.url))
        print('[TITLE]: {}'.format(self.title))
        print('[BODY]:\n{}'.format(self.body))
```

4.3.2 예제 소스 코드 #2: Crawler 클래스

```
class Crawler:
    def __init__(self, site):
        self.site = site # Website 객체
        self.visited = []

    def getPage(self, url):
        try:
            req = requests.get(url)
        except requests.exceptions.RequestException:
            return None
        return BeautifulSoup(req.text, 'html.parser')

    def safeGet(self, pageObj, selector):
        selectedElems = pageObj.select(selector)
        if selectedElems is not None and len(selectedElems) > 0:
            return '\n'.join([elem.get_text() for elem in selectedElems])
        else:
            return ''

    def safeGetBody(self, pageObj, bodyTag):
        bodyElems = pageObj.find_all('p', class_= re.compile(bodyTag))
        bodyText = ''
        if bodyElems is not None and len(bodyElems) > 0:
            for body in bodyElems:
                bodyText += body.get_text() + '\n'
            return bodyText
        else:
            return ''
```

기사 내용을 추출: 정규식 사용
- class의 속성값이 아래 문자열을 포함
'^text__text__+'

4.3.2 예제 소스 코드 #3: Crawler 클래스

```
    def parse(self, url):
        '''
        titleTag와 bodyTag를 검색해서 화면 출력
        '''
        bs = self.getPage(url)
        if bs is not None:
            title = self.safeGet(bs, self.site.titleTag)
            body = self.safeGetBody(bs, self.site.bodyTag)
            if title != '' and body != '':
                content = Content(url, title, body)
                content.print()

    def crawl(self):
        '''
        사이트 홈페이지에서 페이지를 가져옴
        '''
        bs = self.getPage(self.site.url)
        targetPages = bs.find_all('div', class_ = re.compile(self.site.targetPattern))

        for targetPage in targetPages:
            targetPage = targetPage.find('a')['href ' ]
            if targetPage not in self.visited:
                self.visited.append(targetPage)

                if not self.site.absoluteUrl:
                    targetPage = '{}{}'.format(self.site.url, targetPage)

                self.parse(targetPage)
```

URL 추출을 위해 정규식 사용
- class의 속성값이 아래 문자열을 포함
'^media-story-card__placement-container+'

4.3.2 예제 소스 코드 #4: 크롤링 시작

```
link_pattern = '^media-story-card__placement-container+'
body_pattern = '^text__text__+'

reuters = Website('Reuters',                    # Website.name
                  'https://www.reuters.com',# Website.url
                  link_pattern,                 # Website.targetPattern
                  False,                        # Website.absoluteUrl
                  'h1',                         # Website.titleTag
                  body_pattern)                 # Website.bodyTag

crawler = Crawler(reuters)
crawler.crawl()
```

```
[URL]: https://www.reuters.com/markets/europe/chinas-june-factory-services-activity-expands-first-time-four-months-2022-06-30/
[TITLE]: China's factory, service sectors shake off 3 months of lockdown pain
[BODY]:
A worker polishes a bicycle steel rim at a factory manufacturing sports equipment in Hangzhou, Zhejiang province, China September 2, 2019. China Daily via REUTERS
BEIJING, June 30 (Reuters) - China's factory and service sectors snapped three months of activity decline in June, business surveys showed on Thursday, as authorities lifted a strict COVID lockdown in Shanghai,
. . .

[URL]: https://www.reuters.com/world/asia-pacific/tokyos-june-flames-out-record-heatwave-power-plant-shutdown-stokes-blackout-2022-06-30/
[TITLE]: Japan power plant shutdown raises fear of shortage in sweltering heat
[BODY]:
A woman holding an umbrella is reflected on a window while walking along a street, as the Japanese
. . .
```

Questions?