웹 스크래핑

파이썬 프로그래밍

목차

1. 웹의 이해
2. HTML 데이터 다루기
3. 정규 표현식
4. Lab: 웹 스크래핑 실습

01. 웹의 이해

❖ 웹의 개념

- 월드 와이드 웹(World Wide Web)은 인터넷에 연결된 컴퓨터를 이용하여 사람들과 정보를 공유할 수 있도록 거미줄처럼 엮인 공간이다.
- 월드 와이드 웹을 줄여 웹(web)이라고 한다.

01. 웹의 이해

❖ 웹 컴포넌트: HTML과 HTTP : HTML

- HTML(Hyper Text Markup Language)은 웹상의 정보를 구조적으로 표현하기 위한 언어이다.


```
<h1>손하트 날리는 BTS 지민</h1>

<image src=http://img.yonhapnews.
co.kr/photo/yna/YH/2018/10/07//
PYH20181007069200072OO_P4.jpg>

<p>(뉴욕=연합뉴스) 이준서 특파원 = 세계적 케이
팝 그룹 방탄소년단(BTS)이 6일 밤(현지시간) 미
국 뉴욕의 시티필드에서 '러브 유어셀프'(Love
Yourself) 북미투어의 대미를 장식하는 피날레
공연을 하고 있다. 2018.10.7 [빅히트 엔터테인
먼트 제공]</p>
```

[HTML의 예]

01. 웹의 이해

❖ 웹 컴포넌트: HTML과 HTTP : HTML

- 태그(tag)는 꺾쇠 괄호 < >로 둘러싸여 있고, 그 안에 정보에 대한 의미를 적는다. 그리고 그 의미가 끝나는 부분에 슬래시(/)를 사용하여 해당 태그를 종료한다.

```
<title> Hello, World </title>                    # 제목 요소, 값은 Hello, World
```

01. 웹의 이해

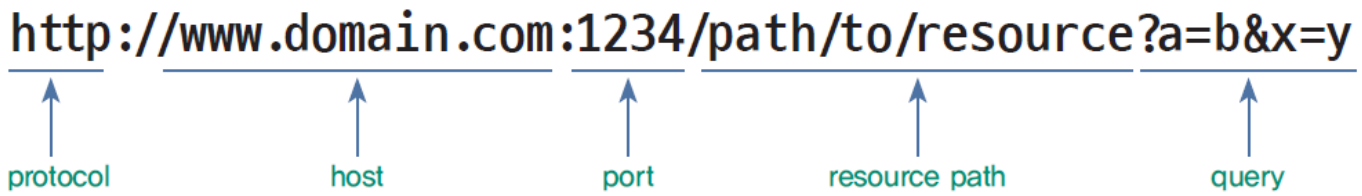
❖ 웹 컴포넌트: HTML과 HTTP : HTTP

- HTTP(Hypertext Transaction Protocol)는 인터넷에서 컴퓨터 간에 정보를 주고받을 때 사용하는 일종의 약속을 말하며, 일반적으로 컴퓨터 과학에서는 이러한 약속을 프로토콜(protocol)이라고 한다.

01. 웹의 이해

❖ 웹의 동작 순서

- 웹에 있는 정보를 보기 위해 먼저 하는 일은 웹 브라우저를 시작하고, 거기에 주소 정보를 입력하는 것이다. 주소 정보의 공식 이름은 URL(Uniform Resource Locator)이라고 한다.

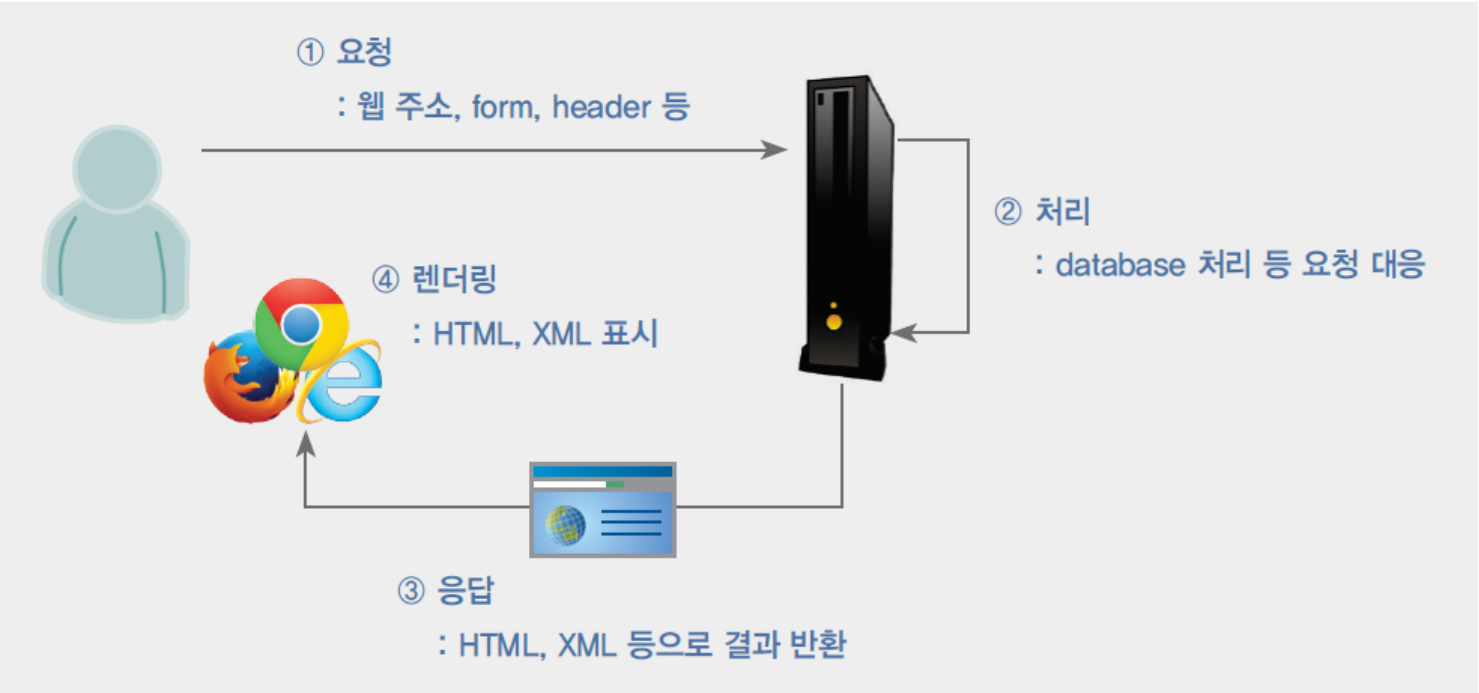


[URL의 구조]

- URL에는 해당 서버가 위치한 인터넷 주소 정보인 도메인 네임(domain name)이 있다. 흔히 도메인 정보 또는 서버 주소라고도 하는 이 주소를 통해 웹의 정보를 제공하는 서버에 접속한다.
- 일반적으로 컴퓨터는 인터넷 프로토콜 주소(Internet Protocol address), 즉 IP 주소(IP address)라고 부르는 주소값을 가진다.
- IP 주소를 컴퓨터의 주소로 생각하면 이 주소에 접속하기 위해 사용하는 도메인 네임과 연결하기 위한 도메인 네임 서버(Domain Name Server, DNS)가 운영된다.

01. 웹의 이해

❖ 웹의 동작 순서



[웹의 동작 순서]

01. 웹의 이해

❖ 웹 스크래핑

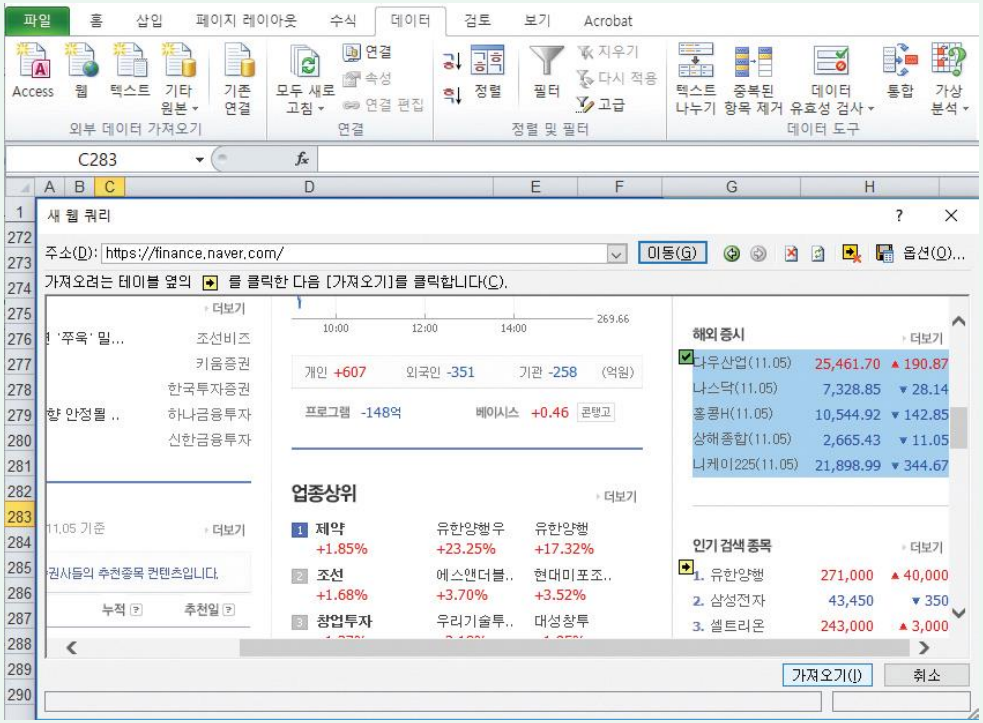
- 모든 웹은 HTML로 구성되어 있으므로, HTML의 규칙을 파악한다면 얼마든지 HTML에서 필요한 정보를 가져올 수 있다. 이러한 과정을 일반적으로 웹 스크래핑(web scrapping)이라고 한다.

01. 웹의 이해

엑셀에서 웹의 정보 가져오기

여기서 잠깐!

- 꼭 파이썬을 사용하지 않더라도 엑셀에서 간단하게 웹의 정보를 가져올 수 있다. 먼저 엑셀을 실행하여 [데이터] - [웹] 메뉴를 선택한다. 대화상자가 열리면 추출하고 싶은 데이터가 있는 웹 주소를 입력한다. 추출하고자 하는 테이블을 선택하면 데이터가 출력되어 엑셀에 기록되는 것을 확인할 수 있다.



[엑셀을 사용한 웹 데이터 추출 방법]

02. HTML 데이터 다루기

❖ 웹에서 데이터 다운로드하기

코드 14-1 html.py

```
1  import urllib.request      # urllib 모듈 호출
2  url ="http://storage.googleapis.com/patents/grant_full_text/2014/ipg140107.zip"
                              # 다운로드 URL 주소
3
4  print("Start Download")
5  fname, header = urllib.request.urlretrieve(url, 'ipg140107.zip') # urlretrieve()
   함수 호출(URL 주소, 다운로드할 파일명), 결과값으로 다운로드한 파일명과 header 정보를 언패킹
6
7  print("End Download")
```

```
Start Download
End Download
```

02. HTML 데이터 다루기

❖ 웹에서 데이터 다운로드하기

- 1행에서는 파이썬의 웹 페이지 연결 모듈인 urllib 모듈을 호출한다. 2행에서 파일이 있는 다운로드 URL 주소를 지정하고, 5행에서 urlretrieve( ) 함수와 URL 주소, 파일명을 입력하여 특정 파일을 저장할 수 있다.
- 이러한 예제는 언제 사용할 수 있을까? 특정 그림이나 강의 자료 같은 데이터를 자동화하여 다운로드할 때 매우 유용하다. 예를 들어, 구글 검색을 통해 특정 인물의 사진만 다운로드 할 수 있는 프로그램을 작성하여 이를 실행하면 효율적으로 특정 인물의 사진을 다운로드 할 수 있다.

02. HTML 데이터 다루기

❖ HTML 파싱

- 웹 페이지의 HTML을 분석하여 필요한 URL을 추출하는 작업이 HTML 파싱이다.
  파싱(parsing)은 특정 텍스트를 분석하여 그 데이터로부터 필요한 정보를 추출하는 과정이다.

```
https://finance.naver.com/item/main.nhn?code=005930
```

페이지를 생성하는 파일(프로그램)

http://finance.naver.com/item/main.nhn?code=005930

본 프로그램에서는 상장코드에 따라 다른 정보를 생성함
(GET 방식)

해당 프로그램의 변수
code: 변수명, 005930: 값

[URL 정보의 표현 방법]

02. HTML 데이터 다루기

❖ HTML 파싱

- 다음으로 웹 페이지에서 필요한 정보만 추출하는 방법은 무엇일까? 아래의 테이블에서 필요한 주식 정보를 어떻게 가져올 수 있을까?

[주식 정보 테이블]

➡ 가장 좋은 방법은 테이블을 구성하는 HTML 페이지를 찾아 그 정보를 가져오는 것이다. 먼저 해당 웹 페이지를 열고 마우스 오른쪽 버튼을 누른 후, '페이지 소스 보기'를 클릭한다. 웹 페이지의 정보가 있는 HTML 코드를 확인할 수 있다. 이 코드를 분석하여 원하는 정보만 따로 추출할 수 있는데, 이를 위해 해당 정보를 표현하는 코드의 패턴을 찾아 HTML 생성 규칙을 파악하고, 이를 추출하는 프로그램을 만들어야 한다.

02. HTML 데이터 다루기

❖ HTML 파싱

- 테이블 정보가 크게는 <dl>~</dl> 클래스 사이의 코드 안에 있고, 각각의 개별 정보는 <dd>~</dd> 태그 사이에 있는 것을 알 수 있다. 이 두 가지 정보만 안다면, 이 패턴으로부터 원하는 주식 데이터를 쉽게 추출할 수 있다.

```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2018년 10월 26일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 41,000 전일대비 보합 0  0.00 퍼센트</dd>
    <dd>전일가 41,000</dd>
    <dd>시가 41,100</dd>
    <dd>고가 41,300</dd>
    <dd>상한가 53,300</dd>
    <dd>저가 40,400</dd>
    <dd>하한가 28,700</dd>
    <dd>거래량 14,000,889</dd>
    <dd>거래대금 572,471백만</dd>
</dl>
```

<dl class="blind"> ~ </dl> 사이 데이터 존재

각 데이터는 <dd> ~ </dd>로 나타나며
데이터 생성 순서는 일시, 종목명 ~ 거래대금 순

[주식 정보 HTML 구조]

03. 정규 표현식

❖ 정규 표현식의 개념

- 정규 표현식(regular expression) 은 일종의 문자를 표현하는 공식으로, 특정 규칙이 있는 문자열 집합을 추출할 때 자주 사용하는 기법이다

일반 문자	정규 표현식
010-0000-0000	^\d{3}-\d{4}-\d{4}$
203.252.101.40	^\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}$

[일반 문자와 정규 표현식의 연결]

03. 정규 표현식

정규 표현식 스스로 연습하기

- 정규 표현식은 문법 자체가 매우 방대하므로 여기서 전부 다룰 수는 없다. 학습자가 스스로 찾아 공부하는 것이 매우 중요하며, 필요한 것이 있을 때마다 인터넷을 검색하며 확인하는 노력이 필요하다.
- 정규 표현식을 연습하기 위한 다양한 방법이 있는데, 가장 좋은 방법 중 하나는 웹에서 제공하는 정규 표현식 연습장을 사용하는 것이다. 정규 표현식 연습장 웹 사이트(http://www.regexr.com)에 접속하면 아래쪽에는 텍스트가, 위쪽에는 정규 표현식을 넣을 수 있는 구조로 설계되어 있다. 여기에 파싱할 텍스트와 정규 표현식을 넣고 실험해 보도록 한다.

03. 정규 표현식

❖ 정규 표현식 문법

- 정규 표현식을 배우기 위해 기본으로 알아야 하는 개념은 메타문자(meta-characters) 이다. 메타문자는 문자를 설명하기 위한 문자로, 문자의 구성을 설명하기 위해 원래의 의미가 아니라 다른 의미로 쓰이는 문자를 뜻한다.

```
. ^ $ * + ? {} [] \ | ()
```

03. 정규 표현식

❖ 정규 표현식 문법 : 기본 메타문자 []

- 먼저 대괄호 [ ]는 [와] 사이의 문자와 매칭하라는 뜻으로 사용된다. 예를 들어, [abc]는 어떤 텍스트에 a 또는 b 또는 c라는 텍스트가 있는지 찾으라는 뜻이다.
- [ ]에는 or의 의미가 있다. 예를 들어, 비틀스의 노래 가사 중 'yesterday' 또는 'Yesterday'라는 단어를 한 번에 찾으려면 [Yy]esterday라고 입력하면 한 번에 정규 표현식으로 검색할 수 있다.
- 만약 [abc]라는 정규 표현식을 쓴다면 다음과 같은 텍스트에서 검색되는 텍스트는 어떤 것이 있을까? 여기서는 'a'의 a, 'before'의 b와 같은 방식으로 검색될 것이다.

```
a, before, deep, dud, sunset
```

- 알파벳 전체나 한글 전체 텍스트를 찾고 싶다면 어떻게 하면 될까? 바로 -을 사용한다. [A-Za-z]나 [가-힣]과 같은 기호로 문자열에서 알파벳과 한글을 추출할 수 있다. 숫자 전체를 추출한다면 [0-9]로 쓸 수 있다.

03. 정규 표현식

❖ 정규 표현식 문법 : 반복 관련 메타문자 -, +, *, ?, { }

- 앞에서 배운 [ ]는 매우 유용하지만, 한 번에 여러 개의 글자를 표현할 수 없다. 예를 들어, 휴대전화번호를 찾고 싶다면 다음과 같이 정규 표현식을 작성해야 한다.

```
[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]-[0-9][0-9][0-9]
```

- 하지만 이렇게 하면 몇 가지 문제점이 있다. 먼저 텍스트를 너무 많이 적는다. 이럴 때 쓸 수 있는 메타문자가 +이다. +는 해당 글자가 1개 이상 출현하는 것을 뜻한다.

```
[0-9]+-[0-9]+-[0-9]+
```

03. 정규 표현식

❖ 정규 표현식 문법 : 반복 관련 메타문자 -, +, *, ?, { }

- 출현 횟수를 조정해야 할 때 사용하는 메타문자는 중괄호 {}이다. 예를 들어, [a-zA-Z]{3,4}이면 알파벳이 3자부터 4자까지 출현할 수 있다는 뜻이다. 반복 횟수는 {1,}, {0,}, {1,3}처럼 시작값이나 끝값은 지정하지 않고 오픈할 수 있다. 만약 {1,}라고 쓴다면 한번 이상 출현해야 한다는 제약이 있다.

```
[0-9]{3}-[0-9]{3,4}-[0-9]{4}
```

- *는 +와 달리 해당 글자가 0번부터 무한대까지 반복할 수 있다. { }를 사용하여 표현한다면 +는 {1,}이고 *는 {0,}을 뜻한다. 예를 들어, tomor*ow라고 표현하면 이 정규 표현식에 해당하는 글자는 무엇일까?

```
tomorrow
tomoow
tomorrrrow
```

03. 정규 표현식

❖ 정규 표현식 문법 : 그외 메타문자 (), ., |, ^, $, \

- 메타문자 ( )는 묶음을 표시하는 것으로, 좀 더 쉽게 메타문자의 묶음을 표시하는 역할을 한다.
- [.]와 (.)의 뜻이 다른데, [.]는 일반적인 마침표를 뜻하고 (.)는 줄 바꿈 기호를 제외한 전체 문자를 뜻한다. 만약 다음과 같은 자막 데이터에서 시간 정보를 없애고 싶다면 어떻게 해야 할까?

```
00:00:09.785 --> 00:00:11.474
안녕하세요. 최성철입니다.

00:00:11.474 --> 00:00:16.215
이번 주차에는 좀 다양한 내용을 다루는데, 이번 주에는 로그를 관리하는
```

- 메타문자 |나 ^은 or와 not의 의미로 많이 사용한다.
- 정규 표현식의 처음과 끝에는 메타문자 ^과 $를 주로 붙인다.

03. 정규 표현식

메타문자를 찾고 싶을 때

- 메타문자를 찾고 싶다면 어떻게 해야 할까? 예를 들어 [를 찾고 싶거나 {를 찾고 싶을 때가 있다. 이 경우에는 ₩ 문자를 사용한다. 파이썬과 마찬가지고 ₩는 특수 기호로, 매우 많은 의미를 가지고 사용할 수 있다. 줄 바꿈 기호의 경우에도 윈도의 cp949 계열 인코딩에서는 ₩n을 쓸 수 있고, 맥의 utf8에서는 ₩r₩n을 쓸 수 있다. 만약 문서 전체를 지정하고 싶다면, (.)+(₩r ₩n)+로 모든 문서를 선택할 수 있다.

03. 정규 표현식

❖ 정규 표현식 연습

- 정규 표현식을 연습해 보자. 먼저 구글에서 제공하는 미국 특허 정보 데이터 세트 홈페이지에 접속한다.

https://www.google.com/googlebooks/uspto-patents-grants-text.html

03. 정규 표현식

❖ 정규 표현식 연습

```
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150106.zip">
ipg150106.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150113.zip">
ipg150113.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150120.zip">
ipg150120.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150127.zip">
ipg150127.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150203.zip">
ipg150203.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150210.zip">
ipg150210.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150217.zip">
ipg150217.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150224.zip">
ipg150224.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150303.zip">
ipg150303.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150310.zip">
ipg150310.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150317.zip">
ipg150317.zip</a> 
```

03. 정규 표현식

❖ 정규 표현식 연습

➡ href 태그가 보일 것이다. 이 태그는 링크(link) 태그로, 해당 주소를 웹 브라우저에 넣으면 파일을 다운로드할 수 있다. 이제 zip 파일을 다운로드하기 위해 정규 표현식을 이용해 글자들을 추출할 것이다.

정규 표현식 연습장(http://www.regexr.com)을 열어 모든 html 소스를 연습장에 붙여넣는다.

다음으로 상단에 정규 표현식을 기재한다. 정규 표현식을 생각해 보면 시작은 http 끝은 zip으로 끝나는 것을 알 수 있다. 중간에는 매우 다양한 글자가 나온다. 따라서 이 문서에서는 (http)(.+)(zip)이라고 입력하면, 해당 링크들의 정규 표현식을 찾을 수 있다.

03. 정규 표현식

❖ 정규 표현식 연습

Expression

```
/(http)(.+)(zip)/g
```

Text

```
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150106.zip">
ipg150106.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150113.zip">
ipg150113.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150120.zip">
ipg150120.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150127.zip">
ipg150127.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150203.zip">
ipg150203.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150210.zip">
ipg150210.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150217.zip">
ipg150217.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150224.zip">
ipg150224.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150303.zip">
ipg150303.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150310.zip">
ipg150310.zip</a> 
<a href="http://storage.googleapis.com/patents/grant_full_text/2015/ipg150317.zip">
ipg150317.zip</a> 
```

[정규 표현식 연습장에서 정규 표현식 찾기]

04. Lab: 웹 스크래핑 실습

❖ 아이디 추출하기

- 먼저 특정 아이디를 추출하는 Lab이다. 다음 웹 사이트에는 이벤트 당첨자의 아이디를 발표한 웹 페이지가 있는데, 이 웹 페이지에서 필요한 아이디만 추출해 보겠다.

```
http://goo.gl/U7mSQl
```

50000 마일리지	codo***	양*로	20000 마일리지	outb7***	장*자
	dubba4***	지*보		multicuspi***	구*상
	crownm***	고*송		triformo***	진*섬
	spania***	이*용		magazin***	홍*익
	presby***	배*경		trophody***	문*형
	외 5명			외 5명	
10000 마일리지	nontr***	전*을	5000 마일리지	enranck***	손*금
	canc***	진*올		uncanker***	고*혁
	wrymo***	문*구		non***	이*후
	luminat***	고*의		oblig***	손*도
	anna***	심*루		hyperth***	나*주
	외 20명			외 35명	
1000 마일리지	toplabl***	김*홍		dolce0***	강*정
	rudals2***	유*현		jjw980***	최*수
	elvlz***	도*녕		skmid***	김*인
	qkep***	김*중		kisslov***	박*규
	maskman***	황*현		sungt***	오*탁
	외 90명				

[이벤트 당첨자 정보]

04. Lab: 웹 스크래핑 실습

❖ 아이디 추출하기

코드 14-2 lab1.py

```
1  import re
2  import urllib.request
3
4  url = "http://goo.gl/U7mSQl"                    # 접속할 웹 페이지
5  html = urllib.request.urlopen(url)               # 웹 페이지 열기
6  html_contents = str(html.read())                 # 웹 페이지의 내용을 문자열로 가져옴
7  id_results = re.findall(r"([A-Za-z0-9]+\*\*\*)", html_contents)
8  # findall 전체 찾기, 정규 표현식 패턴대로 데이터 찾기
9
10 for result in id_results:                        # 찾은 정보를 화면에 출력
11     print (result)
```

04. Lab: 웹 스크래핑 실습

❖ 아이디 추출하기

```
codo***
outb7***
dubba4***
multicuspi***
:                ← 생략
sungt***
```

➡ 위 코드를 보면 먼저 4행에서 접속할 웹 페이지의 링크를 작성하고 5행에서 웹 페이지에 접속한 후, 6행에서 해당 웹 페이지의 HTML 코드를 문자열로 가져온다. 다음으로 해당 HTML 코드를 7행에서 findall( ) 함수를 사용하여 정규 표현식 패턴을 넣어주면 패턴대로 데이터를 찾아서 id_results 변수에 넣어준다. 해당 변수는 튜플 형태로 반환되기 때문에 각각을 출력하기 위해서는 10행과 같이 for문을 사용한다.

04. Lab: 웹 스크래핑 실습

❖ 파일 자동 다운로드

- 파일을 자동으로 다운로드하는 Lab으로, 이전에 다루었던 특허 데이터 웹 페이지에서 파일을 다운로드하는 코드를 만들겠다.

코드 14-3 lab2.py

```
1  import urllib.request                    # urllib 모듈 호출
2  import re
3
4  url ="http://www.google.com/googlebooks/uspto-patents-grants-text.html" # url
   값 입력
5  html = urllib.request.urlopen(url)   # url 열기
6  html_contents = str(html.read().decode("utf8"))    # html 파일 읽고 문자열로 변환
7
8  url_list = re.findall(r"(http)(.+)(zip)", html_contents)    # html 파일 읽고 문자
                                                                열로 변환
9  for url in url_list:
10     full_url = "".join(url)              # 출력된 튜플 형태 데이터를 문자열로 join
11     print(full_url)
```

04. Lab: 웹 스크래핑 실습

❖ 파일 자동 다운로드

```
12      fname, header = urllib.request.urlretrieve(full_url, file_name) # file_
        name에 다운로드할 파일명 입력한 후, 파일 다운로드
13      print("End Download")
```

➡ 먼저 4~8행까지의 코드는 이전과 같다. URL을 연결하고, 해당 웹 페이지에서 HTML을 가져온 후, 정규 표현식으로 파싱한다. 10행에서 join( ) 함수를 사용하는 이유는 정규 표현식을 만들 때, (http)(.+)(zip)을 사용해서 만들었기 때문이다. 여기에서 ( ) 단위로 튜플이 생성된다. 그러므로 (http, .+, zip)이 개별 값으로 생성되기 때문에 실제 사용할 때는 10행과 같이 join( ) 함수를 사용해야 한다. 마지막으로 12행에 파일 다운로드 코드를 작성하고, file_name 부분에 다운로드할 파일명을 입력하여 필요한 파일을 다운로드한다.

04. Lab: 웹 스크래핑 실습

❖ HTML 파싱

- 주식 데이터에서 해당 부분을 파싱하는 코드를 작성하겠다.

(a) 삼성전자 주식 페이지

```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2018년 10월 26일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 41,000 전일대비 보합 0  0.00 퍼센트</dd>
    <dd>전일가 41,000</dd>
    <dd>시가 41,100</dd>
    <dd>고가 41,300</dd>
    <dd>상한가 53,300</dd>
    <dd>저가 40,400</dd>
    <dd>하한가 28,700</dd>
    <dd>거래량 14,000,889</dd>
    <dd>거래대금 572,471백만</dd>
</dl>
```

(b) 삼성전자 주식 페이지의 HTML 코드

[HTML 파싱을 위한 데이터]

04. Lab: 웹 스크래핑 실습

❖ HTML 파싱

- 코드는 크게 두 가지 부분으로 구성된다.
  - ~에 정보가 있음
  - ~ 정보를 추출

```
(\)([\s\S]+?)(\<\/dl>)
```

```
(\)([\s\S]+?)(\<\/dd>)
```

04. Lab: 웹 스크래핑 실습

❖ HTML 파싱

코드 14-4 lab3.py

```
1  import urllib.request
2  import re
3
4  url ="http://finance.naver.com/item/main.nhn?code=005930"
5  html = urllib.request.urlopen(url)
6  html_contents = str(html.read().decode("ms949"))
7
8  # 첫 번째 HTML 패턴
9  stock_results = re.findall("(\<dl class=\"blind\"\>)([\s\S]+?)(\<\/dl\>)", html_
   contents)
10 samsung_stock = stock_results[0]     # 두 개의 튜플 값 중 첫 번째 패턴
11 samsung_index = samsung_stock[1]     # 세 개의 튜플 값 중 두 번째 패턴
12
13 # 주식 정보만 추출함
14 index_list = re.findall("(\<dd\>)([\s\S]+?)(\<\/dd\>)", samsung_index)
```

04. Lab: 웹 스크래핑 실습

❖ HTML 파싱

```
15
16  for index in index_list:
17      print(index[1])                    # 세 개의 튜플 값 중 두 번째 값
```

➡ 코드는 이전과 동일하게 URL에 접속하여 HTML 코드를 추출한다. 다음으로 9~11행에서 정규 표현식을 사용하여 첫 번째 HTML을 추출한다. 해당 패턴을 가진 HTML 코드는 총 2개가 있기 때문에 그 중 첫 번째 튜플 값을 10행에서 선택한다. 앞서 마찬가지로 ( )를 사용해 정규 표현식을 작성했기 때문에, 각 정규 표현식이 튜플로 묶이게 되고 그 중 두 번째 값에 주식정보가 포함되어 있다. 그러므로 11행에서 samsung_stock[1]을 사용해 해당 코드를 가져온다. 마지막으로 dd 태그 사이에 있는 값을 추출하기 위해 해당 정규 표현식을 작성하고, 17행에서 최종 주식 정보를 추출한다.

Thank You !

파이썬 프로그래밍