학습 정리

팀	빅나물	구성원	이지현, 조영현

일정	발제자	주제
6일차(6/3)	이지현	BeautifulSoup 사용법 및 간단 웹 파싱 기초(2)

주요 내용 요약

- 정규표현식

참고사이트:
<http://pythonstudy.xyz/python/article/401-%EC%A0%95%EA%B7%9C-%ED%91%9C%ED%98%84%EC%8B%9D-Regex>

패턴	설명	예제
^	이 패턴으로 시작해야 함	^abc : abc로 시작해야 함 (abcd, abc12 등)
$	이 패턴으로 종료되어야 함	xyz$: xyz로 종료되어야 함 (123xyz, strxyz 등)
[문자들]	문자들 중에 하나이어야 함. 가능한 문자들의 집합을 정의함.	[Pp]ython : "Python" 혹은 "python"
[^문자들]	[문자들]의 반대로 피해야할 문자들의 집합을 정의함.	[^aeiou] : 소문자 모음이 아닌 문자들
|	두 패턴 중 하나이어야 함 (OR 기능)	a | b : a 또는 b 이어야 함
?	앞 패턴이 없거나 하나이어야 함 (Optional 패턴을 정의할 때 사용)	\d? : 숫자가 하나 있거나 없어야 함
+	앞 패턴이 하나 이상이어야 함	\d+ : 숫자가 하나 이상이어야 함
*	앞 패턴이 0개 이상이어야 함	\d* : 숫자가 없거나 하나 이상이어야 함
패턴{n}	앞 패턴이 n번 반복해서 나타나는 경우	\d{3} : 숫자가 3개 있어야 함

패턴{n, m}	앞 패턴이 최소 n번, 최대 m 번 반복해서 나타나는 경우 (n 또는 m 은 생략 가능)	\d{3,5} : 숫자가 3개, 4개 혹은 5개 있어야 함
\d	숫자 0 ~ 9	\d\d\d : 0 ~ 9 범위의 숫자가 3개를 의미 (123, 000 등)
\w	문자를 의미	\w\w\w : 문자가 3개를 의미 (xyz, ABC 등)
\s	화이트 스페이스를 의미하는데, [\t\n\r\f] 와 동일	\s\s : 화이트 스페이스 문자 2개 의미 (\r\n, \t\t 등)
.	뉴라인(\n) 을 제외한 모든 문자를 의미	.{3} : 문자 3개 (F15, 0x0 등)

예제1

```
import re
 text ="에러 1122 : 레퍼런스 오류\n 에러 1033: 아규먼트 오류"
 regex =re.compile("에러\s\d+")
 mc =regex.findall(text)
 print(mc)
 # 출력: ['에러 1122', '에러 1033']
```

예제2

```
import re

 text = "문의사항이 있으면 032-232-3245 으로 연락주시기 바랍니다."

 regex = re.compile(r'(\d{3})-(\d{3}-\d{4})')
 matchobj = regex.search(text)
 areaCode = matchobj.group(1)
 num = matchobj.group(2)
 fullNum = matchobj.group()
 print(areaCode, num) # 032 232-3245
```

예제3

```
import re
```

```
text = "문의사항이 있으면 032-232-3245 으로 연락주시기 바랍니다."

regex = re.compile(r'(?P<area>\d{3})-(?P<num>\d{3}-\d{4})')
matchobj = regex.search(text)
areaCode = matchobj.group("area")
num = matchobj.group("num")
print(areaCode, num)  # 032 232-3245
</num>
```

- 정규 표현식 활용

```
li=soup.find_all(href=re.compile(r"^https://"))
```

^https:// -> https://로 시작

r: rawdata

```
li=soup.find_all(href=re.compile(r"da"))
```

da -> da를 포함

- 다양한 css 선택자 이용해서 가져오기

```
print("1", soup.select("li:nth-of-type(4)")[1].string)
#각 li 태그 그룹의 4번째 요소 선택
print("2",soup.select_one("#ac-list > li:nth-of-type(4)").string)
print("3",soup.select("#ac-list > li[data-lo='cn']")[0].string)
print("4",soup.select("#ac-list > li.alcohol.high")[0].string)
# 클래스 2개인 경우 .으로 연결

param={"data-lo":"cn","class":"alcohol"}
print("5",soup.find("li",param).string)
# 딕셔너리형태도 들어갈 수 있다.
print("6",soup.find(id="ac-list").find("li",param).string)
# 이렇게 하면 정확하게 접근을 한 것이지만 5번째 방법이 더 가독성, 효율성이 좋음.

for ac in soup.find_all("li"):
    if ac['data-lo']=='us':
        print('data-lo==us',ac.string)
```

가져오는 방법만 다를 뿐이지, 궁극적으로 가져 올 값은 같음.

방법은 스스로 하는 것에 따라 다름.

- function 이용하기

```
def car_func(selector):
    print("car_func",soup.select_one(selector).string)

car_func("#gr")
# car_func Grandeur
car_func("li#gr")
car_func("ul > li#gr")
car_func("#cars #gr")
car_func("#cars > #gr")
# 공백으로 연결은 자손
# > 로 연결은 자식
car_func("li[id='gr']")
```

- lambda 이용하기

```
car_lambda=lambda q: print("car_lambda",soup.select_one(q).string)

car_lambda("#gr")
car_lambda("li#gr")
car_lambda("ul > li#gr")
car_lambda("#cars #gr")
car_lambda("#cars > #gr")
car_lambda("li[id='gr']")
```

참고 사이트
인프런
파이썬 입문 및 웹 크롤링을 활용한 다양한 자동화 어플리케이션 제작하기
https://www.inflearn.com/course/python-%ED%8C%8C%EC%9D%B4%EC%8D%AC-%EC%9B%B9-%EB%8D%B0%EC%9D%B4%ED%84%B0-%ED%81%AC%EB%A1%A4%EB%A7%81/dashboard