20220506

BeautifulSoup를 활용한 크롤링 연습

- **네이버 크롤링 (https://finance.naver.com/ 에 접속하여 달러환률 정보 확인!!) 연습**
  - 필요한 라이브러리 import

```
from bs4 import BeautifulSoup
import urllib.request as rq
```

  - urlopen과 BeautifulSoup를 이용하여 바이너리 데이터를 문자열로 변경

```
url = "https://finance.naver.com/"
resp = rq.urlopen(url).read()

data = resp.decode("euc-kr")

soup = BeautifulSoup(data,'html.parser')
```

  - 달러환률 태그를 찾고 확인

```
price = soup.select_one("div.group1 > table.tbl_home > tbody > tr.up.bold > td")
print("미국 환율 : " + price.string + " 원")
```

- **"https://finance.naver.com/" 에서 거래 상위 10개를 추출하여 출력하세요!!!**
  종목과 가격 => 대한전지 2140 와 같은 식으로 출력

```
### "https://finance.naver.com/" 에서 거래 상위 10개를 추출하여 출력하세요!!!
### 종목과 가격 => 대한전지 2140 와 같은 식으로 출력

from bs4 import BeautifulSoup
import urllib.request as rq

url = "https://finance.naver.com/"
resp = rq.urlopen(url).read()

data = resp.decode("euc-kr")

soup = BeautifulSoup(data,'html.parser')

top = soup.select("div.group_type.is_active > table.tbl_home > tbody > tr > th > a")
price2 = soup.select("div.group_type.is_active > table.tbl_home > tbody > tr > td")

dic = {}
lst = []
lst2 = []
for i in price2:
    if i.string != None:
        if i.string != '보합':
            lst.append(i.string)

for j in top:
```

```
    lst2.append(j.string)

for k in range(0,10):
    print(f"{k+1}위 {lst2[k]} - {lst[k]}원")
```

- Or

```
from bs4 import BeautifulSoup
import urllib.request as rq

url = "https://finance.naver.com/"
resp = rq.urlopen(url).read()

data = resp.decode("euc-kr")

soup = BeautifulSoup(data,'html.parser')

top = soup.select("div.group_type.is_active > table.tbl_home > tbody > tr > th > a")
top_list_pr = soup.select("div.group_type.is_active > table.tbl_home > tbody > tr > td")
top_list =[top[i].string for i in range(10)]
top_list_pr1 = [top_list_pr[i].string for i in range(0,30,3)]

for i in range(10):
    print(f"{i+1}위 {top_list[i]} - {top_list_pr1[i]}원")
```

- 링크에 있는 내용 한꺼번에 처리하기
  - 재귀적으로 탐색하기 위해서는 사용하는 함수
  - urljoin(base,"상대경로") 함수
  - 특정 경로를 찾아서 작업을 할때 사용하는 함수

```
from urllib.parse import urljoin

base = "http://example.com/html/a.html"

### 상대 경로
print(urljoin(base, "b.html"))
print(urljoin(base,"sub/c.html"))
print(urljoin(base, "../index.html"))
print(urljoin(base, "../img/home.png"))
print(urljoin(base, "../css/home.css"))

### 절대 경로
print(urljoin(base, "/index.html"))     # 절대 경로
print(urljoin(base, "http://otherExample.com/wiki"))    # 뒤에 있는 값이 주소인 경우 덮어쓰여진다.
print(urljoin(base, "//otherExample.com/test"))         # //를 사용하면 해당 주소로 덮어쓰여진다.
```

  - 처리순서
    1. HTML분석
    2. 링크추출
    3. 각 링크 대상에 대해 다음 동작을 구동
       a. 다운로드 파일인 경우 다운로드
       b. 파일이 HTML이라면, 1번으로 돌아가 다시 실행

- 모든 페이지를 한꺼번에 다운받는 프로그램을 만들기
  - 만들기전에 사이트에 해당하는 정보를 분석해야함
  - 모듈

```
from bs4 import BeautifulSoup
from urllib.request import *
from urllib.parse import *
from os import makedirs
import os.path, time, re
```

  - 처리한 내용이 파일인지 여부 확인을 위한 변수

```
proc_files = {}      # 딕셔너리 생성
```

  - HTML 내부에 있는 링크를 추출하는 함수를 동작

```
def enum_links(html, base):
    soup = BeautifulSoup(html, "html.parser")
    links = soup.select("link[rel ='stylesheet']")      ## CSS 참조
    links += soup.select("a[href]")     # 링크 참조

    result = []
    # href 속성을 추출하고, 링크를 절대 경로로 변환하여 저장
    for a in links:
        href = a.attrs['href']
        url = urljoin(base, href)
        result.append(url)
    return result
```

  - 파일을 다운로드 하고 저장하는 함수

```
def download_file(url):
    o = urlparse(url)
    savepath = "./" + o.netloc + o.path     ## netloc = http://를 제외한 도메인 주소 , path = 도메인 뒤에 있는 경로
    if re.search(r"/$", savepath):          # savepath의 마지막이 "/"라면
        savepath += "index.html"            # 기본 문서
    savedir = os.path.dirname(savepath)

    # 모두 다운로드됐는지 여부 확인
    if os.path.exists(savepath):
        return savepath

    # 다운로드 안됐을 시 다운받을 디렉터리를 생성
    if not os.path.exists(savedir):
        print("mkdir = ", savedir)
        makedirs(savedir)

    # 파일 다운받기
    try:
        print("download = ", url)
        urlretrieve(url,savepath)
        time.sleep(1)   # 1초 쉬었다가 동작
        return savepath
    except Exception as e:
        print("다운로드 오류 발생 : ", url)
        print("다운로드 에러 내용 : ", e)
        return None
```

- HTML을 분석하고 다운받는 함수

```
def analyze_html(url, root_url):
    savepath = download_file(url)
    if savepath is None:
        return
    if savepath in proc_files:      # 이미 처리되었다면 실행하지 않음
        return
    proc_files[savepath] = True
    print("analyze_html=",url)

    # 링크 추출
    html = open(savepath, "r", encoding="utf-8").read()
    links = enum_links(html, url)

    for link_url in links:
        # 링크가 루트 이외의 경로를 나타낸다면 무시
        if link_url.find(root_url) != 0:
            if not re.search(r".css$", link_url):
                continue

        # HTML이라면
        if re.search(r".(html|htm)$", link_url):
            # 재귀적으로 html 파일 분석
            analyze_html(link_url, root_url)
            continue
        # 기타 파일
        download_file(link_url)

# 일반 파이썬 파일인 경우 사용
# if __name__ == "__main__":
#     # url에 있는 모든 것 다운받기
#     url = "https://docs.python.org/3.10/library/"
#     analyze_html(url, url)

url = "https://docs.python.org/3.10/library/"
analyze_html(url, url)
```

- netloc과 path

```
url = "http://example.com/html/a.html"
t = urlparse(url)
t.netloc        # example.com
t.path          # /html/a.html
```

- **requests 패키지를 이용한 접근...**
  - 파이썬으로 로그인하기 (한빛미디어)

```
from selenium import webdriver
USER = "아이디 입력"
PASS = "비밀번호 입력"

path = "C:\\webdriver\\chromedriver"
driver = webdriver.Chrome(path)

# 한빛 로그인페이지
url_login = "https://www.hanbit.co.kr/member/login.html"
driver.get(url_login)

## 로그인 박스에 ID와 비밀번호 입력하기
id = driver.find_element_by_id("m_id")
```

```
id.clear()
id.send_keys(USER)
passwd = driver.find_element_by_id("m_passwd")
passwd.clear()
passwd.send_keys(PASS)

# 로그인 버튼 클릭
driver.find_element_by_class_name("btn_login").click()

# 마이페이지로 이동
driver.find_element_by_class_name("myhanbit").click()

# 가지고있는 마일리지
mileage = driver.find_element_by_css_selector(".mileage_section1 span")
print(f"한빛 마일리지 : {mileage.text}점")

# 한빛이코인 출력
coin = driver.find_element_by_css_selector(".mileage_section2 span")
print(f"한빛 이코인 : {coin.text}원")
```

네이버 로그인(캡챠로 인해 로그인이 막힘)

```
from selenium import webdriver
from random import randint
import os.path, time, re


USER = "아이디 입력"
PASS = "비밀번호 입력"

path = "C:\\webdriver\\chromedriver"
driver = webdriver.Chrome(path)

url_login = "https://www.naver.com/"
driver.get(url_login)

driver.find_element_by_class_name("link_login").click()

id = driver.find_element_by_id("id")
id.clear()
id.send_keys(USER)

passwd = driver.find_element_by_id("pw")
passwd.clear()
passwd.send_keys(PASS)

# 로그인 버튼 클릭
driver.find_element_by_class_name("btn_login").click()

driver.find_element_by_id("new.dontsave").click()

driver.get("http://order.pay.naver.com/home")

mileage = driver.find_element_by_class_name("link strong")

print(f"마일리지 : {mileage.text}원")
```