Python 네이버 크롤링

키워드: 웹 크롤러 (https://ko.wikipedia.org/wiki/%EC%9B%B9_%ED%81%AC%EB%A1%A4%EB%9F%AC), Selenium (https://www.seleniumhq.org/)

네이버 사이트는 국내 인터넷 사용자들이 가장 많이 사용하는 검색 엔진이자 포털 사이트입니다. 오래된 역사만큼 굉장히 많은 양의 데이터를 담고 있습니다. 사용자의 흥미 분석, 시장 동향 분석 등을 위해 네이버 뉴스, 카페, 블로그, 실시간 키워드 등의 데이터는 활용 범위가 굉장히 넓은 플랫폼이라고 할 수 있습니다. 테스트 코드로는 네이버 카페만을 선정하였지만, 더 다양한 카테고리의 데이터를 가져오는 크롤링 코드를 작성해보세요. :)

```
from selenium import webdriver
import requests
from bs4 import BeautifulSoup
import time

URL = "https://nid.naver.com/nidlogin.login"
DRIVER_DIR = '/Users/temp/Project_FC/chromedriver'
ID = "아이디"
PW = "비밀번호"

def naver_scrap():
    try:
        driver = webdriver.Chrome(DRIVER_DIR)
        driver.implicitly_wait(10)
        driver.get(URL)
        print("로그인 페이지에 접근")

        e = driver.find_element_by_id("id")
        e.clear() # 입력 폼을 지웁니다.
        e.send_keys(ID) # 입력 폼에 아이디를 자동 입력합니다.

        e = driver.find_element_by_id("pw")
        e.clear() # 입력 폼을 지웁니다.
        e.send_keys(PW) # 입력 폼에 비밀번호를 자동 입력합니다.

        form =
driver.find_element_by_css_selector("input.btn_global[type=submit]")
        form.submit()
        print("로그인 버튼을 클릭") # 로그인 버튼을 클릭합니다.

        # BASE 시작 냥이네 자유게시판
        driver.get('https://cafe.naver.com/clubpet?
iframe_url=/ArticleList.nhn%3Fsearch.clubid=10625072%26search.menuid=221%26sear
ch.boardtype=L%26search.questionTab=A%26search.totalCount=151%26search.page=1')
 # 원하는 정보가 담긴 주소를 찾습니다.
        driver.switch_to.frame('cafe_main') # iframe(자바스크립트 코드를 통해 동적으로
생성된 코드 구문 등) 코드를 분석하기 위해 사용합니다.
        html = driver.page_source # iframe 페이지 소스 코드를 가져옵니다.
        soup = BeautifulSoup(html, 'html.parser') # 소스 코드를 분석할 수 있는 HTML
DOM 형태로 파싱합니다.
```

```
        BASE = 'https://cafe.naver.com/' # 네이버 카페의 기본 주소입니다.
        spans = [] # 각각의 게시글의 주소를 담을 리스트를 선언합니다.
        # link 주소
        for span_tag in soup.select('table.board-box > tbody span.aaa'): #
게시판의 게시글 주소들을 가져옵니다.
            spans.append(BASE + str(span_tag.find('a')['href'])) # 가져온 정보 중 a
태그의 href 속성의 값을 추출하고, 네이버 카페의 기본 주소에 문자열 결합하여 할당합니다.

        for idx, span in enumerate(spans): # 게시글의 수 만큼 반복합니다.
            driver.get(span) # 하나의 페이지
            driver.switch_to.frame('cafe_main') # iframe 가져오기
            html = driver.page_source # 페이지 소스 가져오기
            soup = BeautifulSoup(html, 'html.parser') # 페이지 소스 html 코드로 파싱

            t = soup.select('div#main-area div.inbox')[0] # 본문 내용 가져오기
            author = t.select('div.fl td.p-nick a')[0].get_text() # 글쓴이
            print('(글쓴이)-> ', author) # 글쓴이 정보를 가져옵니다.

            cmt_list = t.find('ul', attrs={'id':'cmt_list'}) # 댓글 목록
            for com in cmt_list.select('li'):
                user = com.find('a', attrs={'class': '_nickUI'}) # 댓글 작성자
                date = com.find('span', attrs={'class', 'date'}) # 작성 날짜
                comm = com.find('span', attrs={'class', 'comm_body'}) # 작성 글
                if user is not None: # 사용자가 있다면 -> 없을 경우 None 값을 나타냅니다.
                    print(user.get_text(), date.get_text(), comm.get_text())
    except Exception as e:
        print(e)
    finally:
        driver.close() # 드라이버를 닫습니다.

if __name__ == "__main__":
    naver_scrap() # 네이버 크롤링 함수를 호출
```

위 코드에서 중요하게 봐야 할 부분은, "driver.switch_to.frame" 부분입니다. 생각보다 많은 사이트가 동적으로 HTML 문서에 콘텐츠를 끼워 넣는 방식인 "iframe" 코드로 구성되어 있는 경우가 더러 있습니다. 이러한 경우에 바로 해당 코드를 분석할 수 없기 때문에, iframe 정보를 먼저 찾고, 해당 iframe 정보를 분석할 수 있는 형태로 바꿔주어야 합니다. 이 부분을 제외하면, 기존의 크롤링 코드와 큰 차이점이 없습니다. 위 코드에선 각각의 게시물에 달린 댓글 정보를 가져오는 형태이지만, 코드를 약간만 추가하면 게시판의 글도 크롤링할 수 있습니다. 이 부분은 직접 코드로 작성해보세요.^^ :)