크 롤 링 함 수 코 드

2022.01.05

김혜정

[개발 – 자연어처리 팀]

1주차 진행 사항

- 논문 정독
- 크롤링 함수 코드 정리

크롤링 전체 구조

Read_naver_news

각 기사별 url 크롤링

Url_list 반환

News_content_crawling

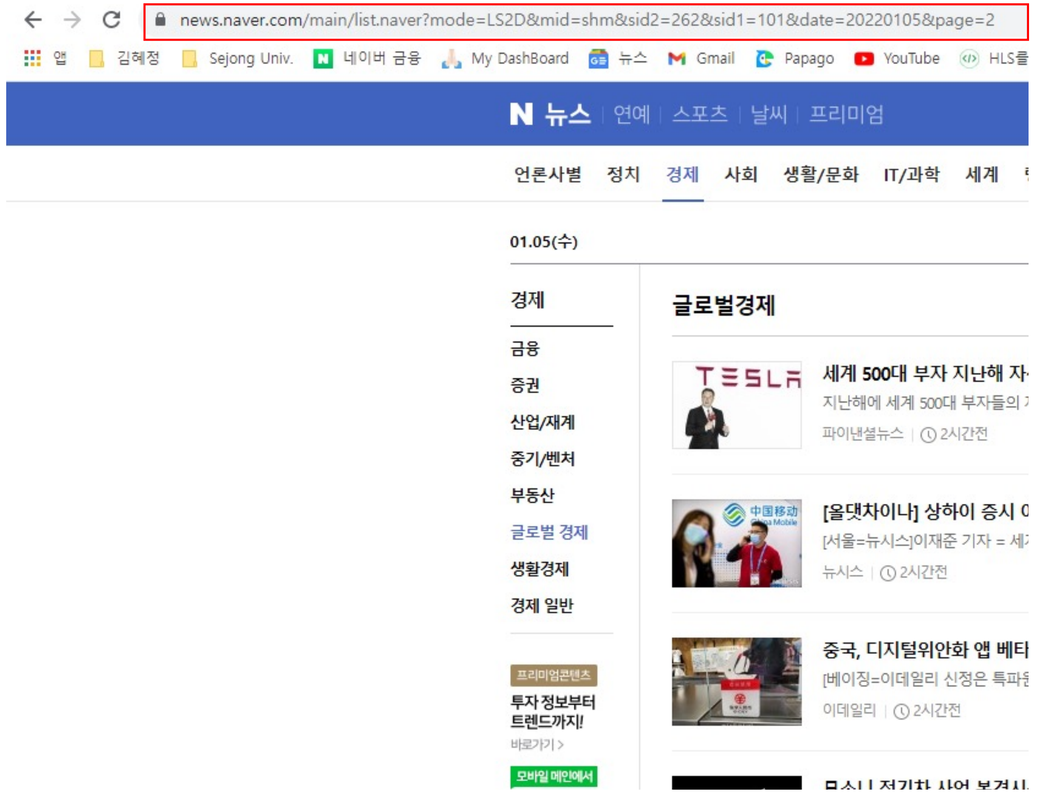
Url에 접근하여 필요한 데이터 크롤링

데이터 -> csv 파일로 저장

url list

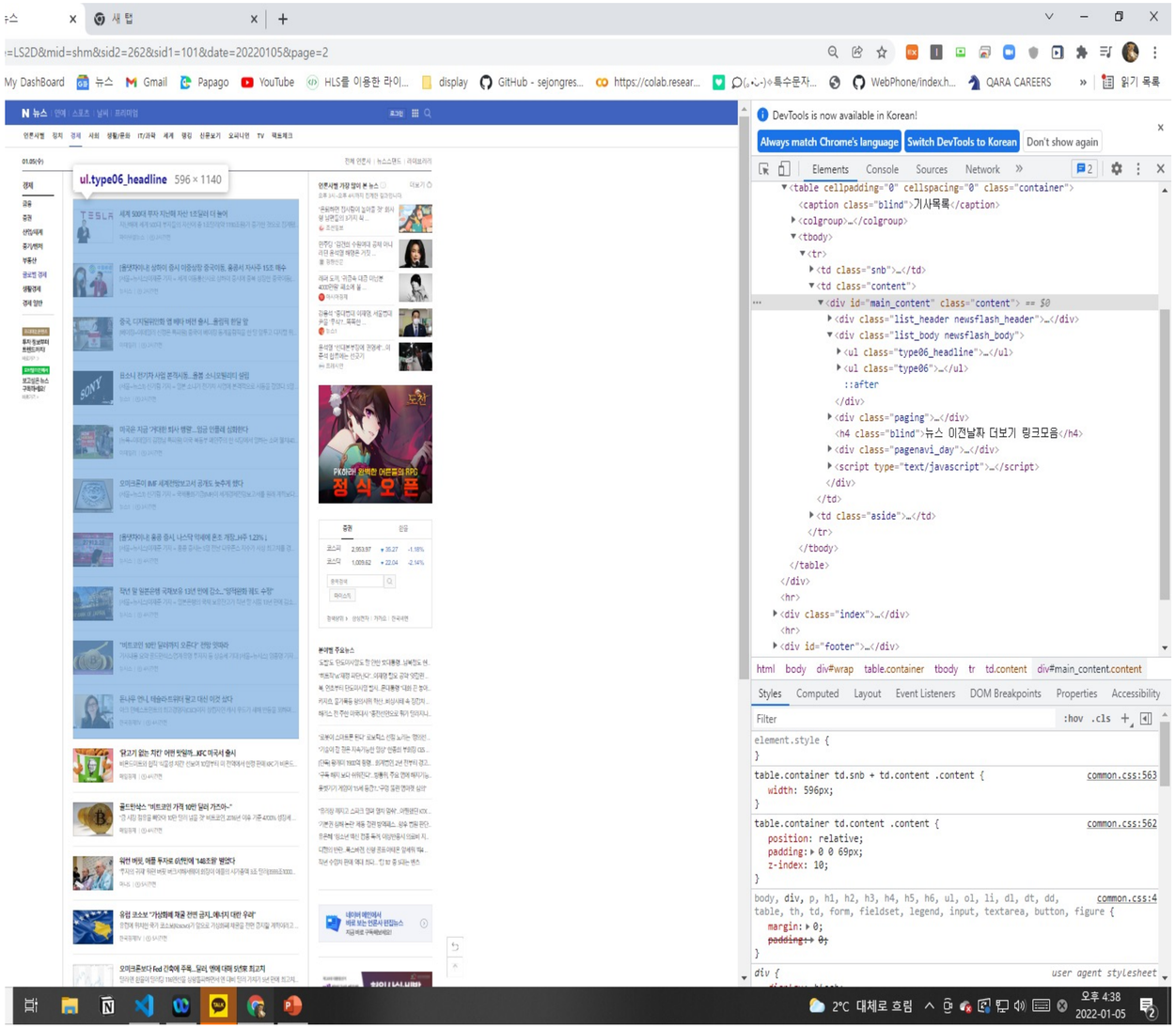
Read_naver_news

```python
def read_naver_news(self):  #뉴스 url list 반환 함수
    date = (datetime.today() - timedelta(8)).strftime("%Y%m%d") # 12월 14일 기준
    page = 1                                                    # 초기 페이지
    last_page = 8                                               # 최종 페이지
    count = 0                                                   # 반복 횟수 카운팅
    url_list = []                                               # 각 뉴스의 url 리스트

    try:
        while True:

            # 종료 조건
            if count == 10 : break                              # 10 page 크롤링 후, 종료

            # url 업데이트
            if page > last_page :                               # last_page이면 이전 날짜로 업데이트
                date = (datetime.today() - timedelta(9)).strftime("%Y%m%d") # 12월 13일
                page = 1

            # 네이버 금융 글로벌 경제 url
            url=f"https://news.naver.com/main/list.naver?mode=LS2D&sid2=262&mid=shm&sid1=101&date={date}&page={page}"
            html_news = BeautifulSoup(requests.get(url, verify=False, headers = {'User-agent' : 'Mozilla/5.0'}).text, 'lxml")

            # type06 headline html 가져오기 (윗 문단)
            ul_tag = html_news.find("ul",class_="type06_headline")
            a_tag = ul_tag.findAll("a")
            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']                     # url 원소 1개씩 저장
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :      # url 중복 방지
                    url_list.append(news_url)

            if page == last_page:                               # last page 에 type 06이 없어서 continue
                count+=1
                page+=1
                continue

            # type06 html 가져오기 (아래 문단)
            ul_tag = html_news.find("ul",class_="type06")
            a_tag = ul_tag.findAll("a")

            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :
                    url_list.append(news_url)

            count+=1                                            # url 크롤링 완료 후, 데이터 업데이트
            page+=1

        return url_list                                         # url list 반환

    except Exception as e:                                      # 에러 발생 시, 에러 출력
        print('Exception occured :', str(e))
        return None
```



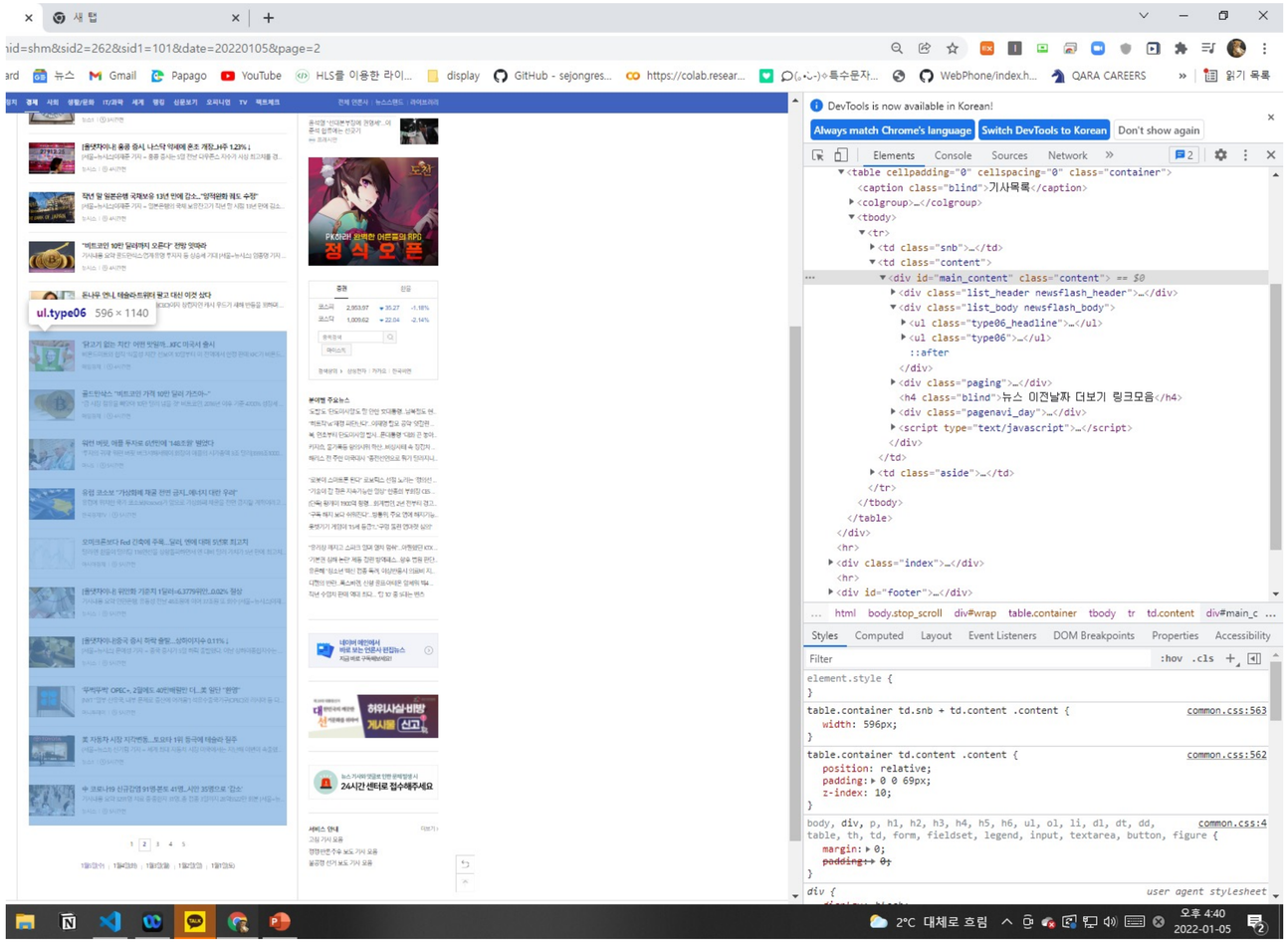
Read_naver_news

```python
def read_naver_news(self):  #뉴스 url list 반환 함수
    date = (datetime.today() - timedelta(8)).strftime("%Y%m%d") # 12월 14일 기준
    page = 1                                                    # 초기 페이지
    last_page = 8                                               # 최종 페이지
    count = 0                                                   # 반복 횟수 카운팅
    url_list = []                                               # 각 뉴스의 url 리스트

    try:
        while True:

            # 종료 조건
            if count == 10 : break                              # 10 page 크롤링 후, 종료

            # url 업데이트
            if page > last_page :                               # last_page이면 이전 날짜로 업데이트
                date = (datetime.today() - timedelta(9)).strftime("%Y%m%d") # 12월 13일
                page = 1

            # 네이버 금융 글로벌 경제 url
            url=f"https://news.naver.com/main/list.naver?mode=LS2D&sid2=262&mid=shm&sid1=101&date={date}&page={page}"
            html_news = BeautifulSoup(requests.get(url, verify=False, headers = {'User-agent' : 'Mozilla/5.0'}).text,"lxml")

            # type06 headline html 가져오기 (윗 문단)
            ul_tag = html_news.find("ul",class_="type06_headline")
            a_tag = ul_tag.findAll("a")
            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']                     # url 원소 1개씩 저장
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :  # url 중복 방지
                    url_list.append(news_url)

            if page == last_page:                               # last page 에 type 06이 없어서 continue
                count+=1
                page+=1
                continue

            # type06 html 가져오기 (아래 문단)
            ul_tag = html_news.find("ul",class_="type06")
            a_tag = ul_tag.findAll("a")

            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :
                    url_list.append(news_url)

            count+=1                                            # url 크롤링 완료 후, 데이터 업데이트
            page+=1

        return url_list                                         # url list 반환

    except Exception as e:                                      # 에러 발생 시, 에러 출력
        print('Exception occured :', str(e))
        return None
```



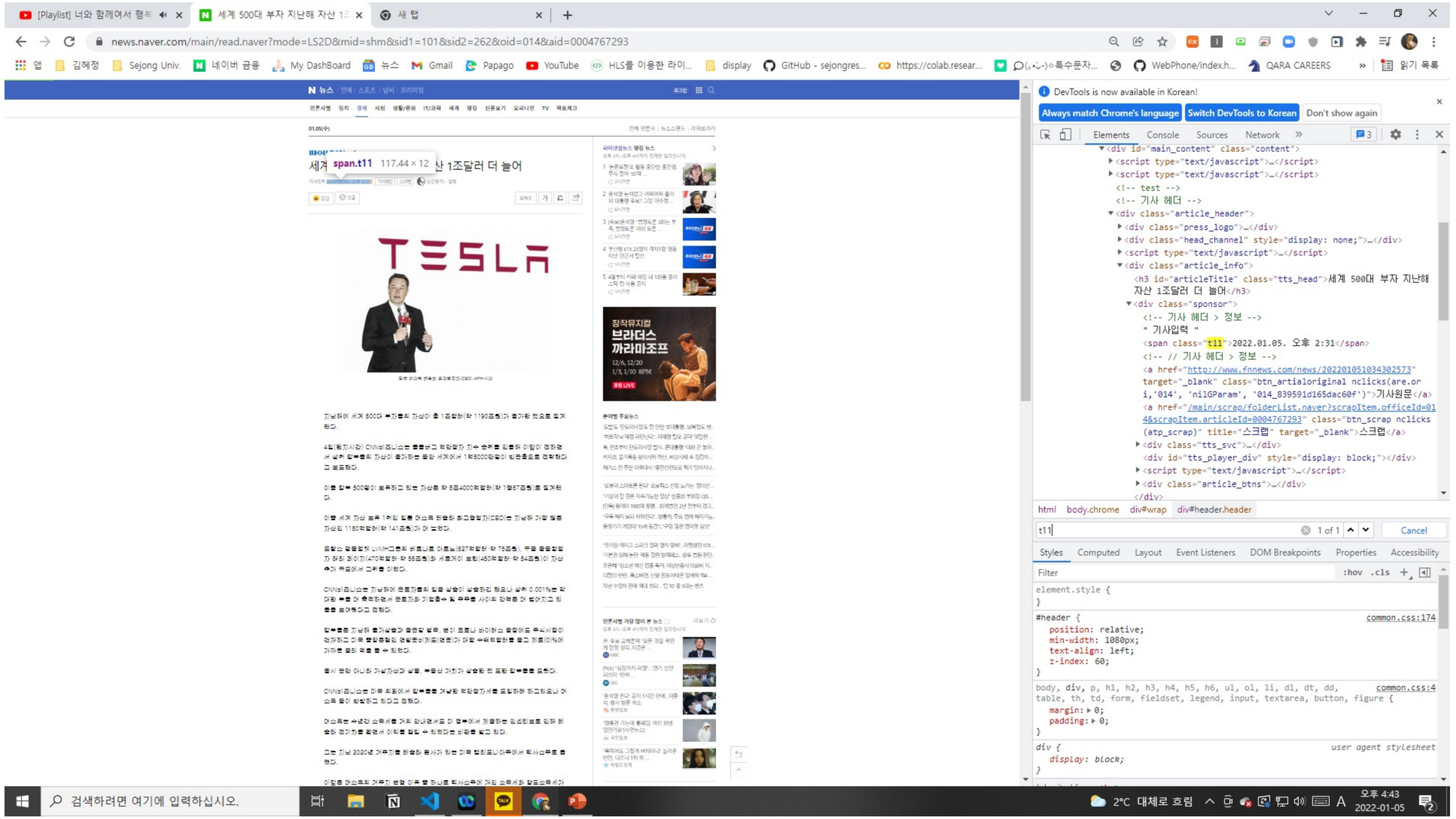
Read_naver_news

```python
def read_naver_news(self):  #뉴스 url list 반환 함수
    date = (datetime.today() - timedelta(8)).strftime("%Y%m%d") # 12월 14일 기준
    page = 1                                                    # 초기 페이지
    last_page = 8                                               # 최종 페이지
    count = 0                                                   # 반복 횟수 카운팅
    url_list = []                                               # 각 뉴스의 url 리스트

    try:
        while True:

            # 종료 조건
            if count == 10 : break                              # 10 page 크롤링 후, 종료

            # url 업데이트
            if page > last_page :                               # last_page이면 이전 날짜로 업데이트
                date = (datetime.today() - timedelta(9)).strftime("%Y%m%d") # 12월 13일
                page = 1

            # 네이버 금융 글로벌 경제 url
            url=f"https://news.naver.com/main/list.naver?mode=LS2D&sid2=262&mid=shm&sid1=101&date={date}&page={page}"
            html_news = BeautifulSoup(requests.get(url, verify=False, headers = {'User-agent' : 'Mozilla/5.0'}).text,"lxml")

            # type06 headline html 가져오기 (윗 문단)
            ul_tag = html_news.find("ul",class_="type06_headline")
            a_tag = ul_tag.findAll("a")
            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']                     # url 원소 1개씩 저장
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :  # url 중복 방지
                    url_list.append(news_url)

            if page == last_page:                               # last page 에 type 06이 없어서 continue
                count+=1
                page+=1
                continue

            # type06 html 가져오기 (아래 문단)
            ul_tag = html_news.find("ul",class_="type06")
            a_tag = ul_tag.findAll("a")

            #각 기사별 url 가져오기
            for i in range(len(a_tag)):
                news_url = a_tag[i]['href']
                if i == 0:
                    url_list.append(news_url)
                elif a_tag[i-1]['href'] != a_tag[i]['href']  :
                    url_list.append(news_url)

            count+=1                                            # url 크롤링 완료 후, 데이터 업데이트
            page+=1

        return url_list                                         # url list 반환

    except Exception as e:                                      # 에러 발생 시, 에러 출력
        print('Exception occured :', str(e))
        return None
```



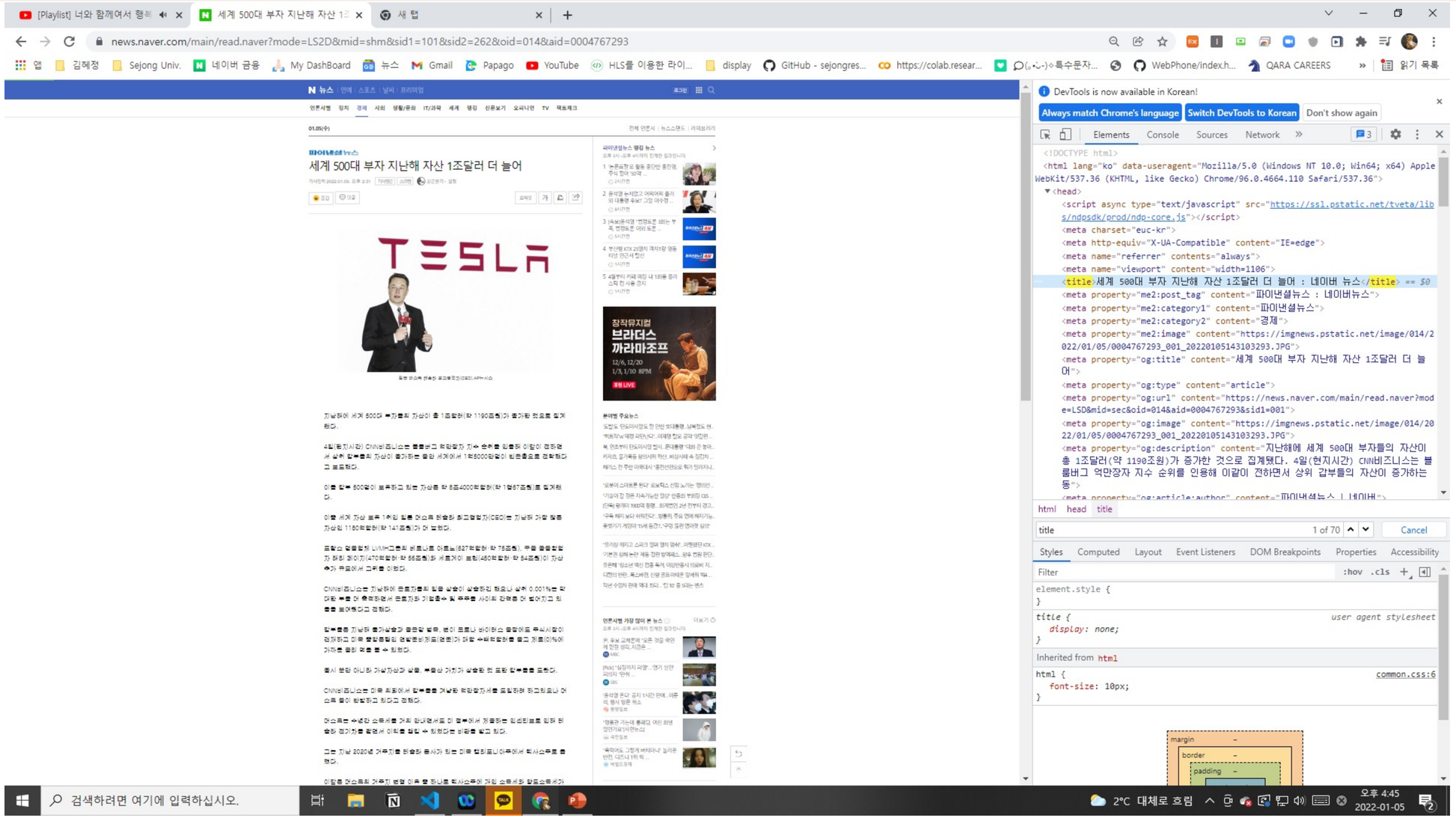
News_content_crawling

- Date 태그



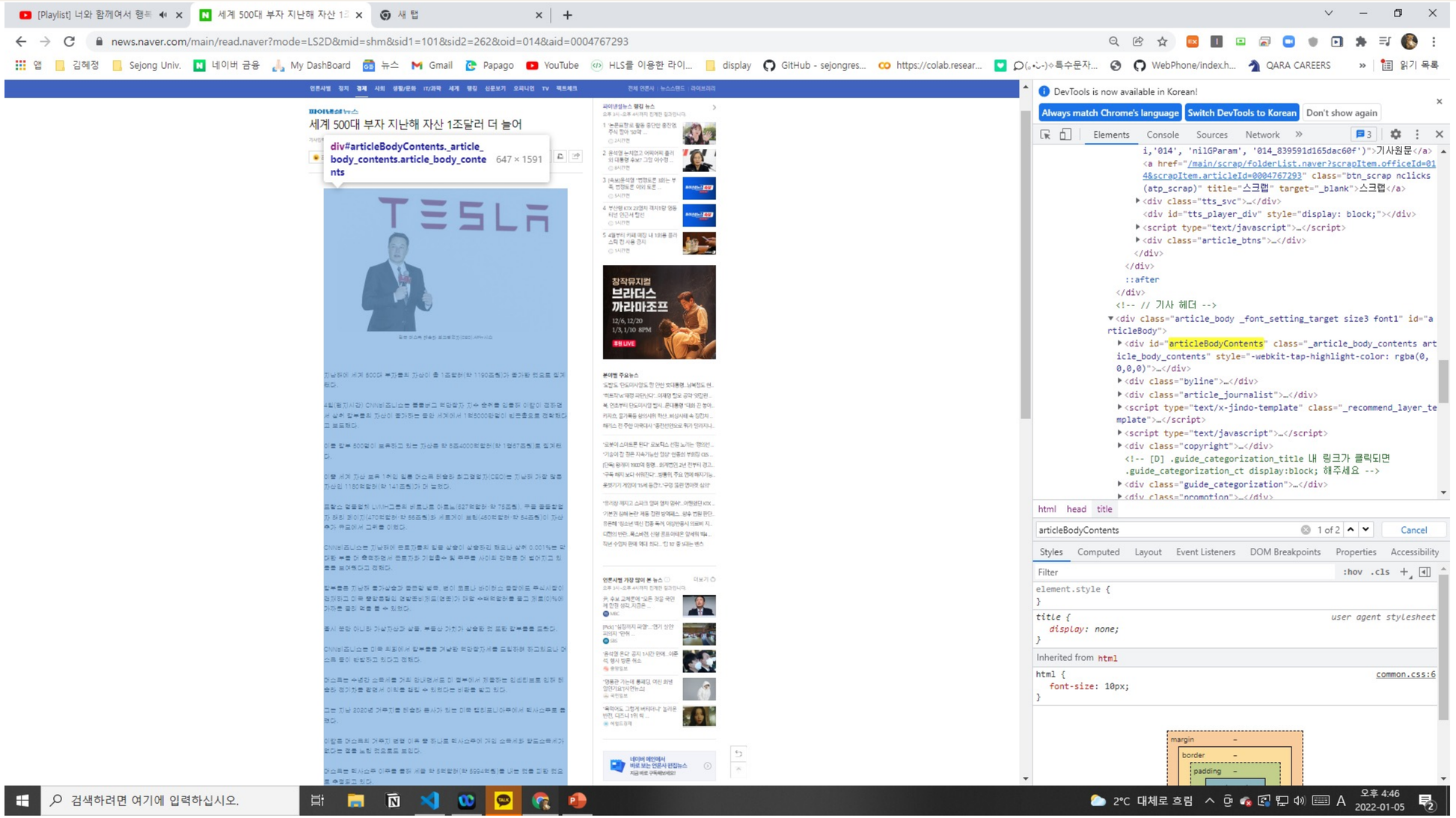
News_content_crawling

- title 태그



News_content_crawling

- articleBodyContents 태그



News_content_crawling

- 해당 태그 찾기

```python
def news_content_crawling(self):  # 뉴스 데이터 크롤링 함수
        (module) pd
    df = pd.DataFrame(columns={"date","title","content"})          # df 생성
    url_list = news.read_naver_news()                              # url 읽어오기

    for i in range(len(url_list)):

        url=url_list[i]                                            # 1개의 url 가져오기
        html_news = BeautifulSoup(requests.get(url, verify=False, headers = {'User-agent' : 'Mozilla/5.0'}).text,"lxml") #parser

        time = html_news.find("span",class_="t11").get_text()      # time 긁어오기
        title = html_news.find("title").get_text()                 # title 긁어오기
        content = html_news.find(id="articleBodyContents").get_text() # 뉴스 본문 긁어오기

        data_list = {"date":time, "title":title, "content":content} # series 형식으로 데이터 저장
        df = df.append(data_list, ignore_index=True)               # 데이터 프레임에 추가

        print('{}/{} pages are downloading...'.format(i+1,len(url_list))) # 다운로드 현황 출력


    df = df[["date","title","content"]]                            # column 순서 맞추기
    print(df)


    #csv 파일로 저장
    dataframe = pd.DataFrame(df)
    dataframe.to_csv("C:/Users/user/OneDrive/바탕 화면/AI_QUANT/news_raw.csv",header=False,index=False)

    #self.replace_into_db(df)                                      # DB 업데이트 함수 실행
```

감사합니다 ☺