실전 웹 크롤링과 자동화 (For 신호용)

1권: 실전 크롤링 입문 - 원티드(Wanted) 정복하기

Chapter 1-5: 데이터 정제 및 저장 - pandas로 CSV 만들기

Prologue: 구슬도 꿰어야 보배

지금까지 우리는 웹페이지에서 '회사명', '직무' 등 개별 데이터를 성공적으로 추출했습니다. 하지만 이 데이터 조각들은 아직 흩어져 있는 구슬과 같습니다. 이 구슬들을 하나의 실에 꿰어 목걸이로 만들어야 비로소 가치 있는 '정보'가 됩니다.

이번 챕터에서는 `pandas` 라는 강력한 데이터 분석 라이브러리를 사용하여, 우리가 수집한 데이터들을 **표(Table)** 형태로 깔끔하게 정리하고, 엑셀 등에서 쉽게 열어볼 수 있는 **CSV(Comma-Separated Values) 파일**로 저장하는 방법을 배웁니다. 이 과정을 통해 우리의 첫 번째 크롤러는 드디어 실용적인 결과물을 만들어내게 될 것입니다.

---

1. 학습 목표 (Objectives)

- `pandas` 라이브러리의 `DataFrame`이 무엇인지 설명할 수 있다.
- 수집한 데이터를 `DataFrame`으로 변환하는 코드를 작성할 수 있다.
- `DataFrame`을 `.to_csv()` 메서드를 사용하여 CSV 파일로 저장할 수 있다.

---

2. 핵심 개념 (Core Concepts)

2.1 `pandas`와 `DataFrame`: 파이썬의 엑셀 시트

- `pandas`: 파이썬에서 데이터를 쉽게 다룰 수 있도록 도와주는 필수 라이브러리입니다. 데이터 분석, 가공, 통계 등 데이터와 관련된 거의 모든 작업에 사용됩니다.
- `DataFrame`: `pandas`의 핵심 자료구조로, 우리가 흔히 보는 **엑셀 시트와 같은 2차원 표(Table)** 형태라고 생각하시면 됩니다. 행(Row)과 열(Column)으로 구성되어 있습니다.

우리는 크롤링한 각 채용 공고를 `DataFrame`의 **행(Row)**으로 만들고, '회사명', '직무' 등의 정보는 각 **열(Column)**에 해당하는 데이터로 구성할 것입니다.

2.2 CSV (Comma-Separated Values)

CSV는 이름 그대로, **'쉼표(Comma)로 값을 구분한 파일'**입니다. 매우 단순한 텍스트 파일 형식이라 용량이 작고, 거의 모든 데이터 분석 도구나 엑셀, 구글 시트 등에서 완벽하게 호환된다는 큰 장점이 있습니다.

[CSV 파일 예시]

```
"회사명","직무","링크"
"현대오토에버","[SDx] Backend Developer - DPL 시스템 운영","/wd/314583"
"카카오","백엔드 개발자","/wd/123456"
```

---

3. 기초 실습 (Basic Practice)

이제 `main.py`의 마지막 부분을 수정하여, 찾은 모든 `job_cards`를 순회하며 데이터를 추출하고, 최종적으로 CSV 파일로 저장하는 코드를 완성하겠습니다.

[환경 설정]

```
pip install pandas
```

[코드 수정] `main.py`의 뒷부분을 아래와 같이 수정합니다.

```
# main.py
... (이전 soup 객체 생성까지 동일) ...

# 7. HTML 파싱하여 정보 추출하기 (전체)
# pandas 라이브러리를 임포트합니다.
import pandas as pd

# 각 공고의 정보를 담을 빈 리스트를 생성합니다.
job_list = []

# CSS Selector로 모든 공고 카드를 찾습니다.
job_cards = soup.select("div[role='listitem'] a")

print(f"총 {len(job_cards)}개의 채용 공고를 찾았습니다.")

# 각 공고 카드를 순회하며 정보를 추출합니다.
for card in job_cards:
    # 'strong' 태그는 직무 제목을 담고 있습니다.
    title = card.select_one("strong[class*='JobCard_title']").text
    # 'span' 태그 중 회사 이름을 담고 있는 요소를 찾습니다.
    company_name = card.select_one("span[class*='CompanyName']").text
    # 공고 상세 페이지로 연결되는 링크는 <a> 태그의 'href' 속성에 있습니다.
    link = "https://www.wanted.co.kr" + card['href']

    # 추출한 정보를 딕셔너리 형태로 저장합니다.
    job_info = {
        'title': title,
        'company': company_name,
        'link': link
    }
    # 리스트에 딕셔셔너리를 추가합니다.
    job_list.append(job_info)
```

```
# 8. pandas DataFrame으로 변환하고 CSV 파일로 저장하기
# 딕셔너리 리스트를 바탕으로 DataFrame을 생성합니다.
df = pd.DataFrame(job_list)

# DataFrame을 CSV 파일로 저장합니다.
# 'index=False'는 엑셀의 행 번호(0, 1, 2...)가 파일에 저장되지 않도록 하는 옵션입니다.
# 'encoding='utf-8-sig''는 한글이 깨지지 않도록 보장해주는 설정입니다.
df.to_csv("wanted_backend_jobs.csv", index=False, encoding='utf-8-sig')

print("CSV 파일 저장이 완료되었습니다.")

# 9. 브라우저 닫기
driver.quit()

print("Selenium 실행이 완료되었습니다.")
```
