# 실전 웹 크롤링과 자동화 (For 신호용)

### 1권: 실전 크롤링 입문 - 원티드(Wanted) 정복하기

---

## Chapter 1-3: 동적 페이지 탐색과 상호작용

### Prologue: 로봇에게 생명을 불어넣기

이전 챕터에서 우리는 브라우저를 열고 특정 URL로 이동하는 데 성공했습니다. 하지만 원티드 같은 사이트의 채용 공고는 한 페이지에 모두 담겨있지 않습니다. 스크롤을 끝까지 내려야만 새로운 공고들이 계속해서 로딩됩니다.

이번 챕터에서는 우리의 크롤러가 마치 사람처럼 **마우스 스크롤을 내리고, 특정 버튼을 클릭하며, 필요하다면 글자를 입력하는 등** 웹페이지와 '상호작용'하는 방법을 배웁니다. 이 기술을 통해 우리는 숨겨진 모든 데이터를 빠짐없이 수집할 수 있게 될 것입니다.

---

### 1. 학습 목표 (Objectives)

- 웹페이지의 특정 HTML 요소를 **클릭**하고, **텍스트를 입력**하는 코드를 작성할 수 있다.
- `Selenium`으로 페이지를 맨 아래까지 자동으로 스크롤하는 코드를 작성할 수 있다.
- 페이지 로딩을 기다리는 다양한 대기(Wait) 방법의 차이를 이해하고 적용할 수 있다.

---

### 2. 핵심 개념 (Core Concepts)

#### 2.1 요소(Element) 찾기: 크롤러의 눈

`Selenium`이 스크롤을 하거나 버튼을 클릭하려면, 먼저 **'무엇을'** 클릭할지 알려줘야 합니다. 웹페이지의 모든 구성요소(버튼, 입력창, 글 제목 등)를 **'요소(Element)'**라고 부릅니다. 우리는 `driver.find_element()` 메서드를 사용하여 이 요소들을 찾을 수 있습니다.

요소를 찾는 방법은 여러 가지가 있습니다.

- `By.ID`: `id` 속성으로 찾기 (가장 빠르고 확실하지만, `id`가 없는 경우가 많음)
- `By.CLASS_NAME`: `class` 속성으로 찾기
- `By.TAG_NAME`: 태그 이름(`div`, `a` 등)으로 찾기
- **`By.CSS_SELECTOR` (가장 중요!)**: CSS 선택자 문법을 사용하여 매우 정밀하게 요소를 찾을 수 있습니다. 우리가 가장 많이 사용하게 될 방법입니다.

#### 2.2 페이지 스크롤: 숨겨진 정보 찾아내기

원티드와 같은 '무한 스크롤' 페이지의 모든 정보를 얻으려면, 페이지 맨 아래까지 계속 스크롤을 내려 더 이상 새로운 콘텐츠가 로딩되지 않을 때까지 반복해야 합니다. `Selenium`에서는 **JavaScript 코드를 직접 실행**하는

방식으로 스크롤을 제어할 수 있습니다.

> **[Developer's Tip]** `driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")` 이 코드는 "현재 페이지의 맨 아래(document.body.scrollHeight)로 스크롤을 이동시켜라" 라는 의미의 JavaScript 명령어입니다.

### 2.3 현명하게 기다리기: `Explicit Waits`

이전 챕터에서 우리는 `time.sleep(5)`를 사용했습니다. 이것은 '무작정 5초 기다리기'라는 단순한 방법입니다. 하지만 네트워크 상황에 따라 5초가 부족할 수도, 혹은 너무 길 수도 있습니다.

더 똑똑한 방법은 **'명시적 대기(Explicit Waits)'**를 사용하는 것입니다. 이것은 **"최대 N초까지 기다리되, 만약 내가 지정한 '어떤 요소'가 화면에 나타나면 더 이상 기다리지 말고 바로 다음 작업을 시작해!"** 라고 명령하는 방식입니다. 훨씬 더 효율적이고 안정적입니다.

---

## 3. 기초 실습 (Basic Practice)

`main.py` 파일을 아래 내용으로 수정하고, 신호용 님이 분석한 내용이 어떻게 코드로 반영되었는지 확인해 봅시다.

```
# main.py

import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
# 키보드 입력을 위해 Keys 모듈을 임포트합니다.
from selenium.webdriver.common.keys import Keys

# 1. WebDriver 설정
service = ChromeService(executable_path=ChromeDriverManager().install())
driver = webdriver.Chrome(service=service)

# 2. 원티드 메인 페이지로 이동
URL = "https://www.wanted.co.kr/"
driver.get(URL)
time.sleep(2)

# 3. 검색창 활성화를 위해 돋보기 아이콘 클릭하기
# 신호용 님이 분석한대로, 먼저 돋보기 버튼을 클릭해야 검색창이 나타납니다.
# 버튼의 class 이름을 사용하여 요소를 찾습니다.
search_button = driver.find_element(By.CSS_SELECTOR,
"button.Aside_searchButton__Xh65A")
search_button.click() # 찾은 버튼을 클릭합니다.
time.sleep(1) # 검색창이 나타날 때까지 잠시 대기

# 4. 검색창을 찾아 '백엔드' 입력하기
# 이제 활성화된 검색창을 찾습니다.
# placeholder 속성을 사용하거나, 더 명확한 class 이름을 사용할 수 있습니다.
search_input = driver.find_element(By.CSS_SELECTOR,
```

```
    "input.SearchInput_SearchInput__R6jwT")
    search_input.send_keys("백엔드") # 검색창에 '백엔드' 텍스트를 입력합니다.
    # Keys.ENTER를 사용하여 엔터 키를 누르는 것을 시뮬레이션합니다.
    search_input.send_keys(Keys.ENTER)
    time.sleep(2)

    # 5. 페이지 끝까지 스크롤하기
    # (이하 스크롤 코드는 이전과 동일합니다)
    last_height = driver.execute_script("return document.body.scrollHeight")
    while True:
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(2)
        new_height = driver.execute_script("return document.body.scrollHeight")
        if new_height == last_height:
            break
        last_height = new_height

    print("페이지 스크롤이 완료되었습니다.")
    time.sleep(5)

    # 6. 브라우저 닫기
    driver.quit()

    print("Selenium 실행이 완료되었습니다.")
```

**[코드 설명 보강]**

- `search_button.click()`: `find_element`로 찾은 요소를 마우스로 클릭하는 역할을 합니다.
- `Keys.ENTER`: `send_keys()` 메서드는 단순히 텍스트만 입력하는 것이 아니라, `Keys` 모듈과 함께 사용하여 엔터, Esc, 화살표 키 등 특수한 키보드 입력도 시뮬레이션할 수 있습니다.
- **CSS Selector 분석:** `button.Aside_searchButton__Xh65A`는 "button 태그이면서(`button`), class 이름이 Aside_searchButton__Xh65A인(`.Aside_searchButton__Xh65A`) 요소"를 의미합니다. 개발자 도구에서 해당 요소의 class 속성값을 복사하여 사용하면 됩니다.

---

## [Mission 3] 상호작용 자동화 실행 (재도전)

이제 수정된 코드로 다시 한번 미션을 수행해 주십시오.

> 1. `main.py` 파일을 위 코드로 수정한 뒤, 터미널에서 `python main.py`를 실행하십시오.
> 2. **예상 결과:**
>
> - 브라우저가 열리고 원티드 메인 페이지로 이동합니다.
> - **상단의 돋보기 아이콘이 자동으로 클릭**됩니다.
> - 나타난 검색창에 **'백엔드'가 자동으로 입력**되고 엔터가 눌러집니다.
> - 검색 결과 페이지로 이동한 뒤, **페이지가 끝까지 자동으로 스크롤**됩니다.
> - 모든 과정이 끝나면 브라우저가 자동으로 닫힙니다.