

Selenium 1

1. Install Selenim and Chrome Driver
2. Selenium의 기본 사용법

1. Install Selenim and Chrome Driver

패키지 설치

```
In [ ]: !pip install selenium
```

```
In [ ]: !pip install webdriver_manager
```

패키지 호출

```
In [7]: from selenium import webdriver
```

```
In [8]: from selenium.webdriver.chrome.service import Service as ChromeService  
from webdriver_manager.chrome import ChromeDriverManager
```

브라우저 열기

```
In [9]: driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()))
```

webdriver_manager의 ChromeDriverManager().install()은 다음과 같은 일련의 행동을 자동으로 처리해줍니다.

- 본인 크롬 버전 체크
- 본인에게 맞는 크롬 드라이버 다운로드
- 경로 설정 자동 처리

- 셀레니움은 <https://www.selenium.dev/> 웹페이지에서 자세한 내용과 사용법을 확인하실 수 있습니다.
- 셀레니움은 브라우저를 자동화하는 툴로 테스트를 목적으로 웹 어플리케이션을 자동화하는 목적으로 만들어졌지만 테스트에 국한되지 않고 사용되지는 않습니다.
- 웹 기반의 작업들을 자동화하여 업무 효율성을 높일 수 있으며 다양한 언어와 다양한 브라우저를 지원합니다.
- <https://www.selenium.dev/documentation/webdriver/browsers> 웹 페이지에서 지원하는 브라우저를 확인할 수 있으며, 공식 홈페이지에서 지원하는 OS와 지원하는 언어를 확인하실 수 있습니다.
- 셀레니움은 Firefox, Edge, Internet Explorer, Safari, Chrome 브라우저를 지원합니다.
- 지원하는 OS로는 Microsoft Windows, Apple OS X, Linux를 지원하며, 지원하는 언어는 C#, Haskell, Java, Javascript, Object-C, Perl, PHP, Python, R, Ruby를 지원합니다.

2.2 웹 드라이버 객체 만들기 및 페이지 이동

- driver의 get 함수를 이용하여 크롬 브라우저에서 페이지를 이동할 수 있습니다.
- 아래의 코드는 naver웹 페이지로 이동합니다.
- url은 http를 꼭 포함해야 합니다

```
In [10]: driver.get("https://www.naver.com/")
```

2.3 윈도우 사이즈 조절

- driver의 set_window_size 함수를 이용하여 윈도우 사이즈를 조절할 수 있습니다.
- 함수내의 첫번째 파라미터는 가로 사이즈를 나타내며 두번째 파라미터는 세로 사이즈를 나타냅니다.
- 사이즈의 단위는 픽셀입니다.

```
In [11]: # (가로, 세로) - 1024 * 768 픽셀의 사이즈로 윈도우 브라우저를 열어줍니다.  
driver.set_window_size(1024, 768)
```

2.4 브라우저의 스크롤 위치 이동

- 브라우저의 스크롤 위치 이동은 selenium의 webdriver에 따로 기능이 없기 때문에 javascript 코드를 실행 하여 브라우저 스크롤 위치를 이동시킬 수 있습니다.

- javascript의 window.scrollTo 함수의 첫번째 파라미터는 가로축 위치를 나타내며 아 래 코드와 같이 200으로 설정을 하면 왼쪽에서 200 픽셀만큼의 브라우저 스크롤 위치를 이동함을 의미합니다.
- 두번째 파라미 터는 세로축의 위치를 나타내며 아래의 코드와 같이 300으로 지정을 하면 위에서 아래로 300픽셀 만큼 스크롤이 내려감을 의미합니다.

```
In [12]: driver.execute_script("window.scrollTo(200, 300);")
```

2.5 Alert 다루기

- 크롤링이나 자동화를 하는 중간에 alert 메시지가 나오는 경우가 있습니다.
- 이럴때 크롤링이나 자동화를 계속 하게 하기 위해 alert 메시지를 다룰줄 알아야 합니다.

```
In [13]: # alert 체크
try:
    alert = driver.switch_to.alert
    print(alert.text)
except:
    print('alert 없음')
```

alert 없음

```
In [14]: driver.execute_script("alert('selenium test');")
```

```
In [15]: # alert 체크
try:
    alert = driver.switch_to.alert
    print(alert.text)
except:
    print('alert 없음')
```

selenium test

```
In [16]: # alert 확인 버튼 누르기
alert.accept()
```

2.7 버튼 클릭하기

- selenium driver의 css selector와 click 함수를 이용하여 웹페이지의 input 태그에 문자열을 입력할수 있습니다.

```
In [17]: from selenium.webdriver.common.by import By
```

```
In [18]: driver.find_element(By.CSS_SELECTOR, ".MyView-module__link_login__HpHMW").click()
```

2.8 윈도우 및 브라우저 종료하기

- selenium driver의 quit 함수를 이용하여 윈도우 창을 닫을수 있습니다

```
In [19]: # 브라우저 닫기  
driver.quit()
```