



Selenium

Contents

Unit 01 | 사용 이유

Unit 02 | 셀레늄 이란

Unit 03 | 활용법

Unit 04 | 참고

Unit 05 | 실습

Unit 01 | 사용 이유

기존 크롤링 과정 Review

1. URL을 가져온다.
2. readLines 로 읽는다.
3. 원하는 부분을 찾는다.

Unit 01 | 사용 이유

But,,

NAVER

아이디

비밀번호

로그인



로그인 상태 유지

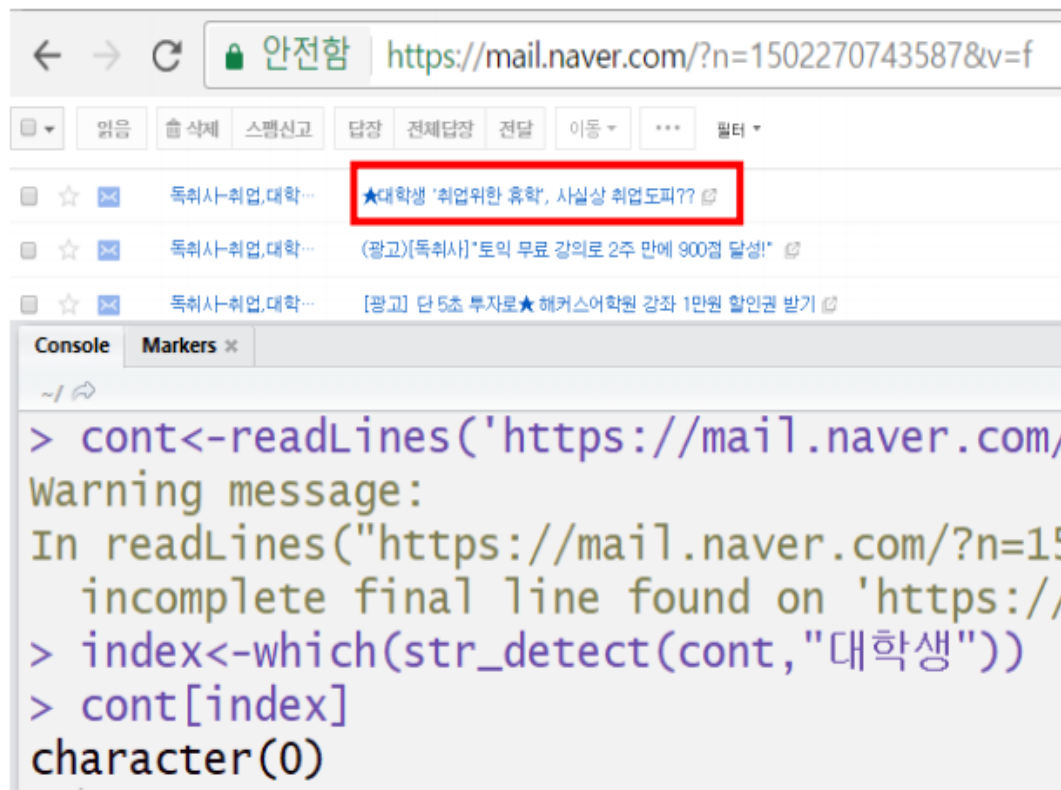
IP보안 ON

일회용 로그인 ?

로그인이 필요할때나
HTML Source 가 없는 경우가 있다.

자바스크립트를 많이 사용하는
웹 사이트는 웹 브라우저를 사용하지 않을
경우 동작을 사용할 수 없다.
그래서 request 모듈로는 대처할 수 없음

Unit 01 | 사용 이유



Why?

Source에 메일 관련된 내용이 없기 때문

Unit 02 | 셀레늄이란

셀레늄이란?

브라우저를 직접 동작시킨다는 것은 JavaScript를 이용해 비동기적으로 혹은 뒤늦게 불러와지는 콘텐츠들을 가져올 수 있다는 것이다.

requests에서 사용했던 `.text`의 경우 브라우저에서 '소스보기'를 한 것과 같이 동작하여, JS등을 통해 동적으로 DOM이 변화한 이후의 HTML을 보여주지 않는다.
반면 Selenium은 실제 웹 브라우저가 동작하기 때문에 JS로 렌더링이 완료된 후의 DOM결과물에 접근이 가능하다고 한다.
DOM (Document Object Model) : 문서(Html)를 구조화 한것

Unit 02 | 셀레늄이란

셀레늄이란?

브라우저를 직접 동작시킨다는 것은 JavaScript를 이용해 비동기적으로 혹은 뒤늦게 불러와지는 콘텐츠들을 가져올 수 있다는 것이다.

requests에서 사용했던 `.text`의 경우 브라우저에서 '소스보기'를 한 것과 같이 동작하여, JS등을 통해 동적으로 DOM이 변화한 이후의 HTML을 보여주지 않는다.
반면 Selenium은 실제 웹 브라우저가 동작하기 때문에 JS로 렌더링이 완료된 후의 DOM결과물에 접근이 가능하다고 한다.
DOM (Document Object Model) : 문서(Html)를 구조화 한것

Unit 02 | 셀레늄이란

Selenium : 주로 웹앱을 테스트하는데 이용하는 프레임워크

webdriver라는 API를 통해 브라우저를 제어

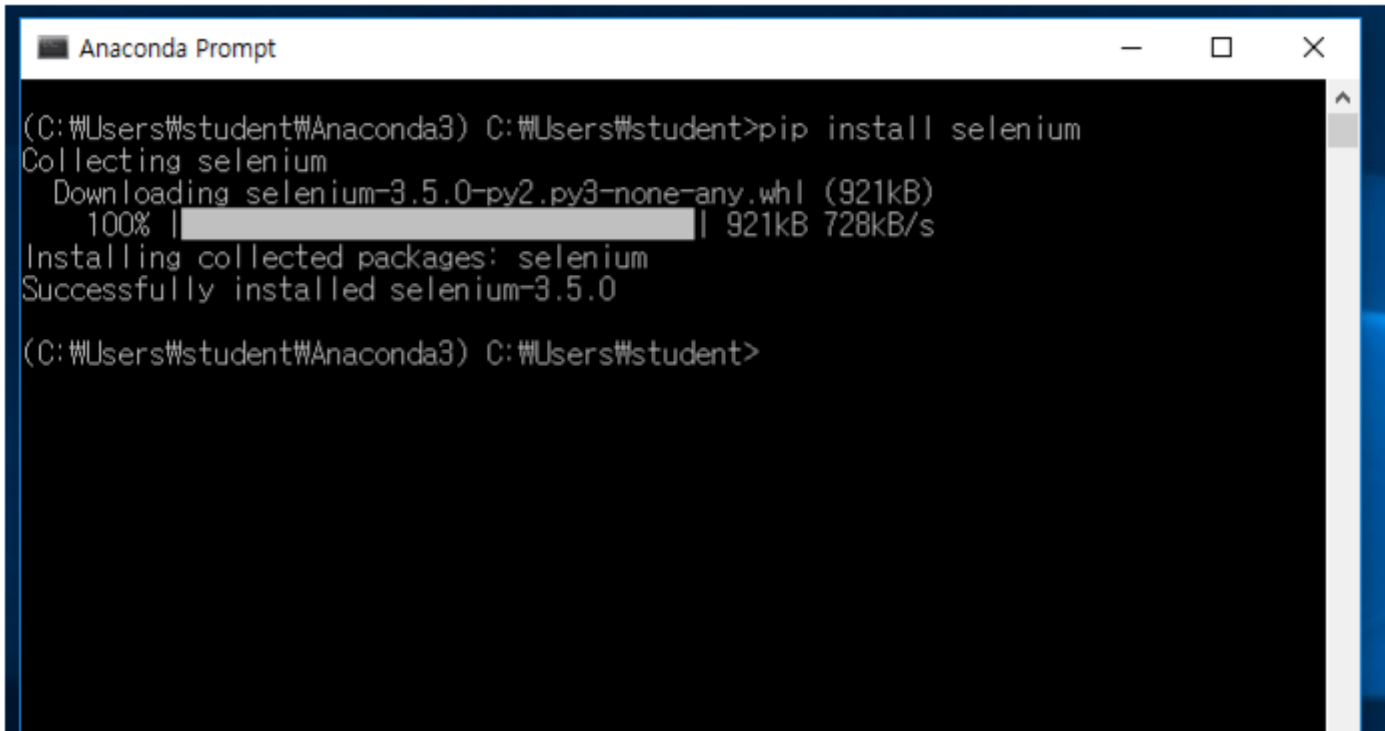
눈에 보이는 콘텐츠라면 모두 가져올 수 있다

직관적이나 속도가 느리다

Bs4와 적절히 섞어서 쓰면 좋은 도구

Unit 03 | 셀레늄 활용법

Setting



```
Anaconda Prompt
(C:\Users\student\Anaconda3) C:\Users\student>pip install selenium
Collecting selenium
  Downloading selenium-3.5.0-py2.py3-none-any.whl (921kB)
    100% |#####| 921kB 728kB/s
Installing collected packages: selenium
Successfully installed selenium-3.5.0

(C:\Users\student\Anaconda3) C:\Users\student>
```

CMD 창에서
1. `pip install selenium`
을 입력한다.

Unit 03 | 셀레늄 활용법

2. 아래의 경로에서 크롬드라이버 설치

<https://sites.google.com/a/chromium.org/chromedriver/downloads>

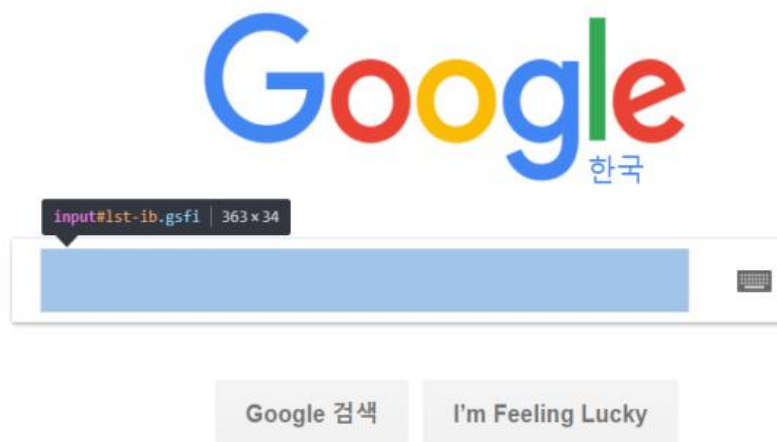
3. 설치 후

```
from selenium import webdriver
```

입력

Unit 03 | 셀레늄 활용법

Selenium에서는 class, id, name 을 먼저 찾아야 한다.
(element 라고 부른다)
세 가지 모두 같은 동작을 하고 다른점은 없다.



```
<div id="gs_l" style="position: relative;">  
  <input class="gsfi" id="lst-ib" maxlength="2048" name="q"  
    autocomplete="off" title="검색" type="text" value="" aria-label="검색"  
    aria-haspopup="false" role="combobox" aria-autocomplete="both"  
    style="border: none; padding: 0px; margin: 0px; height: auto; width:  
    100%; background: url("data:image/gif;base64,R0lGODlhAQABAID/  
    AMDAwAAAAACH5BAEAAAAALAAAAABAAEAAAIICRAEAOw%3D%3D") transparent;  
    position: absolute; z-index: 6; left: 0px; outline: none;" dir="ltr"  
    spellcheck="false">
```

```
In [7]: driver.find_element_by_class_name('gsfi').send_keys('검색창에 입력')
```

```
In [8]: driver.find_element_by_id('lst-ib').send_keys('검색창에 입력')
```

```
In [9]: driver.find_element_by_name('q').send_keys('검색창에 입력')
```

Unit 03 | 셀레늄 활용법

[element 찾는 함수]

find_element_by_id : id 값을 이용하여 돔 접근하는 함수

find_element_by_name : name 속성값을 찾아주는 함수

find_element_by_class_name : 클래스를 이용해 찾는 함수

나머지 함수는 그때그때 찾아가면서 활용

Unit 03 | 셀레늄 활용법

```
<div id="wrap">
  <div id="header_wrap" role="heading">...</div>
  <script type="text/javascript">...</script>
  <div id="container" role="main">
    <div id="content" class="pack_group">
      <h1 class="blind">000 검색결과 시작</h1>
      <div id="main_pack" class="main_pack">
        <div id="nx_related_keywords" class="sp_keyword section">...</div>
        <script type="text/javascript">...</script>
        <script>g_crt+="";</script>
        <script type="text/javascript">...</script>
        <script type="text/javascript"> function img_autoscroll_cli
tCR('img_noc.more',0,0); } </script>
        <link rel="stylesheet" type="text/css" href="https://ssl.ps
search/pc/2016/css/sp_image_170330.css">
      <div class="sp_image section" id="_sau_imageTab" style="ove
        <div class="section_head">...</div>
        <div class="photowall _photoGridWrapper">
          <div class="preview_viewer _indicator"></div>
          <div class="photo_grid_box" data-box-idx="0" style="wic
            <div class="img_noc_item" data-id="news2410002601626
```

Class = "section_head" 에
접근이 안될 경우

상위 element

Class = "sp_image_section"로 접근 해본다.

동일명의 element가 겹치는 경우가 있기 때문

Unit 03 | 셀레늄 활용법

Tutorial

1. 드라이버 경로 지정

```
In [22]: path = 'C:/Users\good\Desktop\chromedriver.exe'  
         driver=webdriver.Chrome(path)
```

Unit 03 | 셀레늄 활용법

2. 브라우저 실행

```
# naver에 접속한다
```

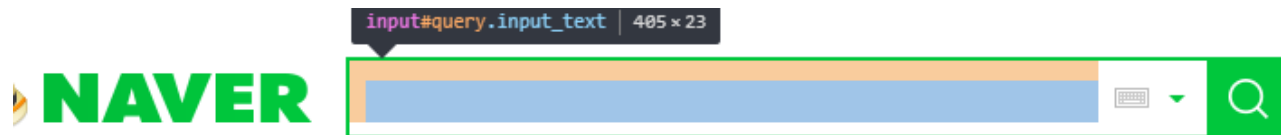
```
driver.get("https://naver.com")
```

**NAVER**

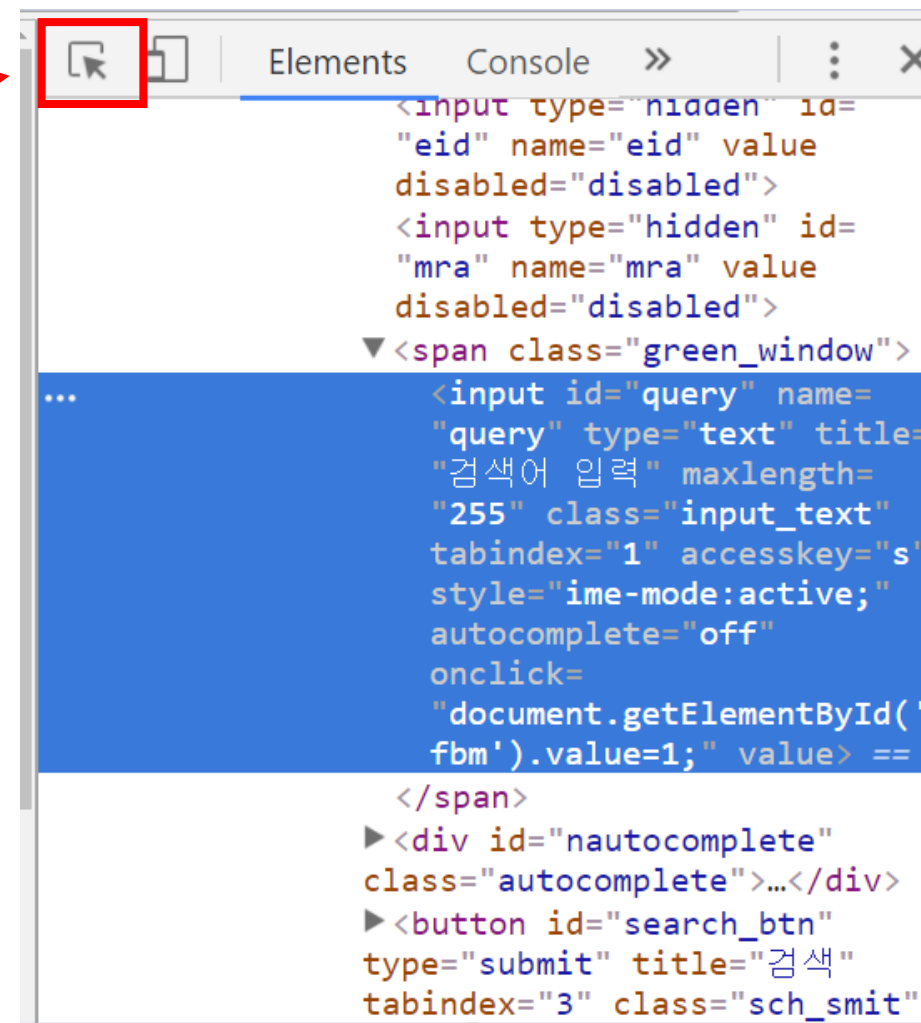
메일 카페 블로그 지식iN 쇼핑 Pay ▶TV 사전 뉴스 증권 부동산 지도 영화 뮤직 책 웹툰 | 더보기 ▾

Unit 03 | 셀레늄 활용법

3. Ctrl + Shift + I 눌러 Elements 확인 Ctrl + Shift + C 누르고 접근할 위치 커서 놓기

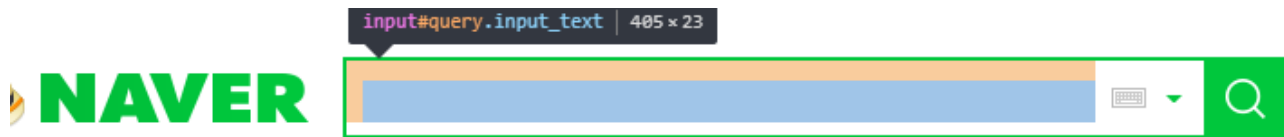


카페 블로그 지식iN 쇼핑 Pay ▶TV 사전 뉴스 증권 부동산 지도 영화 뮤직 책 웹툰 | 더보기 \

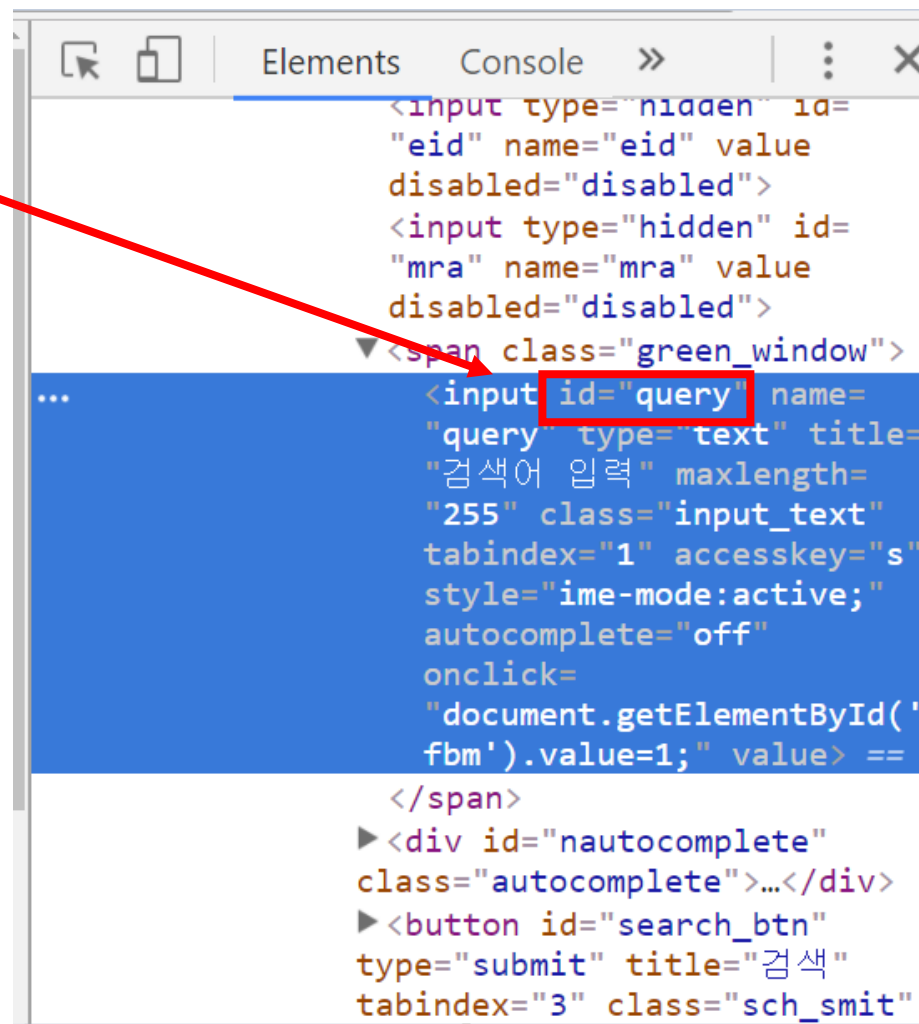


Unit 03 | 셀레늄 활용법

4. search = find_element_by_id('query')



카페 블로그 지식iN 쇼핑 Pay ▶TV 사전 뉴스 증권 부동산 지도 영화 뮤직 책 웹툰 | 더보기 \



Unit 03 | 셀레늄 활용법

사람이 키보드 누르는 것처럼 해주는 모듈
`from selenium.webdriver.common.keys import Keys`

5. 검색어 입력

```
In [28]: search.send_keys("손흥민")
```

NAVER

손흥민

손흥민 골

손흥민

손흥민

Unit 03 | 셀레늄 활용법

6. Enter 키

```
In [29]: search.submit()
```

NAVER

손흥민

손흥민 골

손흥민



NAVER

손흥민

통합검색

동영상

이미지

뉴스

실시간검색

카페

블로그

웹사이트

더

연관검색어 ?

토틀넘

토틀넘 본머스

본머스 토틀넘

토틀넘 손흥민

손흥민 연봉

이승우

손흥민 골

손흥민 해외반응

손흥민 유소영

가생이닷컴

이강인

인물 정보



내 프로필 수정

손흥민 축구선수

출생 1992년 7월 8일, 강원도 춘천

신체 183cm, 77kg

소속팀 **토틀넘 홉스퍼 FC** (FW 공격수, 7)가족 아버지 **손웅정**, 형 **손흥윤**학력 **동북고등학교** 중퇴관련정보 **프리미어리그 - 손흥민 경기 성적**사이트 **공식사이트**, **페이스북**

Unit 04 | Cheat Sheet

Selenium은 브라우저를 지정할 수 있다.

Wedriver.Firefox	파이어 폭스
Wedriver.Chrome	크롬
Wedriver.Ie	인터넷 익스플로러
Wedriver.Opera	오페라
Wedriver.PhantomJS	팬텀(화면없는 웹 브라우저)

Reference : 이삭

Unit 04 | Cheat Sheet

여러 요소 중 처음 찾아지는 요소를 추출

find_element_by_id(id)	Id 속성으로 요소 하나 추출	
find_element_by_name(name)	Name속성으로 요소 하나 추출	
find_element_by_class_name(name)	클래스 이름이 name에 해당하는 요소 하나 추출	find_elementS_by_class_name
find_element_by_partial_link(text)	링크의 자식요소에 포함돼 있는 텍스트로 요소 하나 추출	find_elementS_by_partial_link
find_element_by_tag_name(name)	태그이름이 name에 해당하는 요소 하나 추출	find_elementS_by_tag_name
find_element_by_link_text(text)	링크 텍스트로 요소 하나 추출	
find_element_by_xpath(query)	Xpath를 지정해 요소 하나 추출	find_elementS_by_xpath
find_element_by_css_selector(query)	css 선택자로요소 하나 추출	find_elementS_by_css_selector

Unit 04 | Cheat Sheet

Selenium으로 요소 조작하기

<code>clear()</code>	글자 입력란에 글자를 지움
<code>click()</code>	요소를 클릭
<code>get_attribute(name)</code>	Name에 해당하는 값을 추출
<code>is_displayed()</code>	요소가 화면에 출력되는지 확인
<code>is_selected()</code>	체크박스 등의 요소가 선택된 상태인지 확인
<code>is_enabled()</code>	요소가 활성화돼 있는지 확인
<code>screenshot(filename)</code>	스크린샷을 찍는다.
<code>send_keys(value)</code>	키를 입력한다.
<code>submit()</code>	입력 양식을 전송한다.
<code>value_of_css_property(name)</code>	Name에 해당하는 CSS속성의 값을 추출
<code>Id</code>	요소의 id속성
<code>location</code>	요소의 위치

Unit 04 | Cheat Sheet

parent	부모 요소
rect	크기와 위치정보를 가진 딕셔너리 자료형을 리턴
screenshot_as_base64	BASE64로 스크린샷을 추출
screenshot_as_png	PNG형식으로 스크린샷 추출
size	요소의 크기
tag_name	태그 이름
text	요소의 내부글자

send_key()로 키를 입력할 때 텍스트 이외에도 특수키를 입력할 수 있다.

```
from selenium.webdriver.common.keys import Keys
```

ARROW_DOWN / ARROW_UP / ARROW_RIGHT / ARROW_LEFT

BACKSPACE / DELETE / HOME / END / INSERT

ALT / COMMAND / CONTROL / SHIFT

ENTER / ESCAPE / SPACE / TAB

F1 / F2 / F3 / ... F12

Unit 04 | Cheat Sheet

Selenium 드라이버 조작

<code>add_cookie(cookie_dict)</code>	쿠키 값을 딕셔너리 형식으로 지정
<code>back()/ forward()</code>	이전 페이지 또는 다음페이지로 이동
<code>close()</code>	브라우저를 닫는다.
<code>current_url</code>	현재 URL을 추출
<code>delete_all_cookies()</code>	모든 쿠키를 제거
<code>delete_cookie(name)</code>	특정 쿠키를 제거
<code>excute(command, parmas)</code>	브라우저의 고유 명령어를 실행
<code>excute_async_script(script,*args)</code>	비동기 처리하는 자바스크립트를 실행
<code>excute_sript(script,*args)</code>	동기 처리하는 자바스크립트를 실행
<code>get(url)</code>	Url로 브라우저 이동
<code>get_cookie(name)</code>	특정 쿠키값을 추출
<code>get_cookies</code>	모든 쿠키 값을 딕셔너리 형식으로 추출
<code>get_log(type)</code>	로그를 추출(browser / drivrt / client / server)
<code>get_screenshot_as_base64()</code>	Base64형식으로 스크린샷을 추출

Unit 04 | Cheat Sheet

<code>get_screenshot_as_file(filename)</code>	스크린샷을 파일로 저장
<code>get_screenshot_as_png()</code>	PNG형식으로 스크린 샷을 추출
<code>get_window_position(windowHandle='current')</code>	브라우저의 위치를 추출
<code>get_window_size(windowHandle='current')</code>	크라우저의 크기를 추출
<code>implicitly_wait(sec)</code>	대기시간을 초 단위로 지정해 대기(보통은 time모듈 이용)
<code>quit()</code>	Selenium자체를 종료
<code>save_screenshot(filename)</code>	스크린 샷을 저장
<code>set_page_load_timeout(time_to_wait)</code>	페이지를 읽는 타임아웃 시간을 지정
<code>set_script_timeout(time_to_wait)</code>	스크립트의 타임아웃 시간을 지정
<code>set_window_position(x,y,windowHandle='current')</code>	브라우저의 위치를 지정
<code>set_window_size(가로, 세로, windowHandle='current')</code>	브라우저의 크기를 지정
<code>title</code>	현재페이지의 타이틀을 추출

대용량 자료를 다운받을 시
싱글코어를 이용하면 너무느리다

→ 멀티 쓰레드 or 멀티 프로세싱 사용

GIL을 도입하면

인터프리터의 구현이 쉬워집니다.

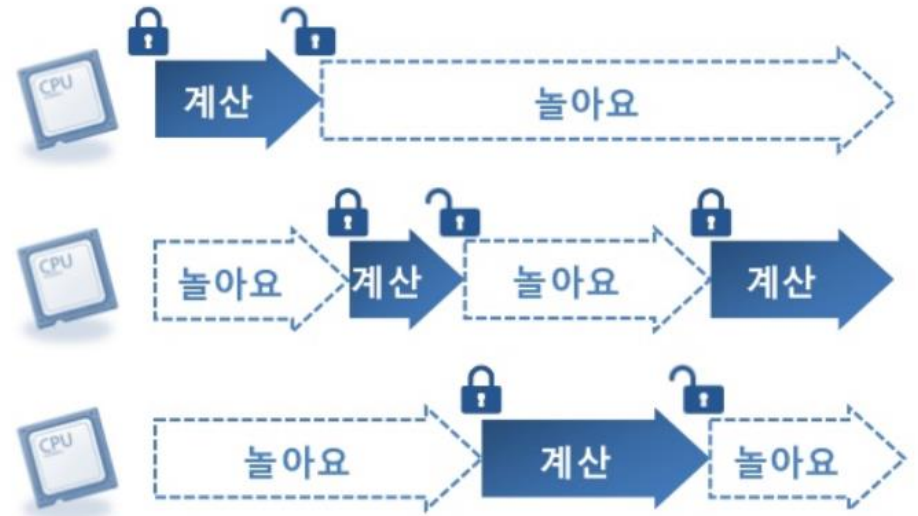
Garbage Collector 만들기도 좋습니다.

C/C++ 확장 모듈 만들기를 쉽게 해줍니다.

이 때문에 파이썬이 쉽게 퍼질 수 있었습니다.

GIL때문에 일어난 일

자물쇠는 하나뿐 → 한 CPU만 일(계산)한다!



파이썬에서는 Global Interpreter Lock 때문에
오히려 싱글쓰레드보다 속도저하가 일어난다

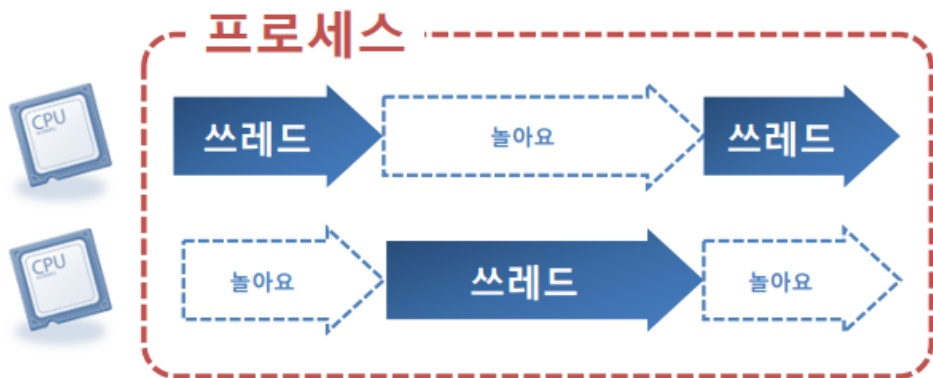
Unit 04 | 참고

쓰레드 대신 프로세스를 만들어 주는 모듈

Multiprocessing module

<http://docs.python.org/library/multiprocessing.html>

Unit 04 | 참고



multiprocessing 모듈은
쓰레드 대신 프로세스를 띄워줍니다.



Multiprocessing 은

쓰레드 쓰듯 프로세스를 쓸 수 있습니다.

```
th1 = Thread(target=do_work, args=(START, END, result))  
th1.start()  
th1.join()
```



```
pr1 = Process(target=do_work, args=(START, END, result))  
pr1.start()  
pr1.join()
```

Unit 04 | 참고

```
from multiprocessing import Process, Queue
def do_work(start, end, result):
    sum = 0
    for i in range(start, end):
        sum += i
    result.put(sum)
    return
if __name__ == "__main__":
    START, END = 0, 20000000
    result = Queue()
    pr1 = Process(target=do_work, args=(START, END/2, result))
    pr2 = Process(target=do_work, args=(END/2, END, result))
    pr1.start()
    pr2.start()
    pr1.join()
    pr2.join()
    result.put('STOP')
    sum = 0
    while True:
        tmp = result.get()
        if tmp == 'STOP': break
        else: sum += tmp
    print "Result : ", sum
```

쓰레드 2개

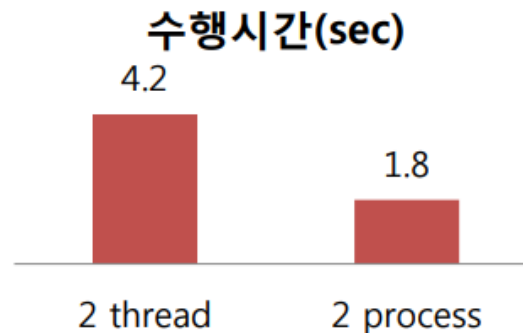
```
$ time python 2thread.py
Result : 199999990000000
```

real 0m4.225s

프로세스 2개

```
$ time python 2process.py
Result : 199999990000000
```

real 0m1.880s



성공!

Unit 05 | 실습

Jupyter notebook

Unit 05 | 참고 링크

1. 셀레니움 유튜브 강의

<https://www.youtube.com/watch?v=V69wc4Tmwjc&list=PLUY1IsOTtPeJNBuSweXS9pcSKbP4mr32S>

2. 구글 이미지 서치 API

<https://developers.google.com/image-search/>

3. 크롬 확장 플러그인 사용 (사이트의 모든 이미지 저장)
Bulk Download Images (ZIG)

<https://chrome.google.com/webstore/detail/zzllrr-imager-geek/gfjhimhkjmipphnaminnnnjpnlnneepk?hl=ko>

4. 파이썬으로 클라우드 하고 싶어요_분산기술Lab_하용호

https://www.slideshare.net/kthcorp/h32011c6pythonandcloud-111205023210phpapp02?from_m_app=ios



들어주셔서 감사합니다.