

# Robotic Process Automation

2023. 10. 05.

김현용

충북대학교 산업인공지능학과

---

RPA 개요

---

RPA와 파이썬

---

OpenPyXl 엑셀 자동화

---

웹의 개요

---

Open API

---

Web Scraping

---

Selenium 웹 자동화

---

PyAutoGui 화면 제어

---

## ■ RPA (Robotic Process Automation) 란?

- 사람이 하던 정형화되고 반복적인 업무를 자동화하는 것
- (소프트웨어/디지털) 로봇 또는 봇 → **사무자동화** vs. Robot = Programmable Manipulator or Vehicle
- 맥킨지(McKinsey)의 조사에 따르면 60%의 직업에서 “**직무활동 중 최소 1/3 이상을 자동화할 수 있다**”

사람의 노동을 디지털 노동으로 대체해  
업무의 효율을 높이고 비용을 최소화시킬 수 있는,  
**RPA(Robotic Process Automation)**

## ■ RPA의 역사

- 1990년대부터 단순하고 반복적인 업무를 줄이려는 시도가 있었고, 2000년대 초반에 등장한 용어 (BluePrism社)
- 미국, 유럽과 같은 선진국에서는 IT, 금융, 사무 분야에서 RPA가 대세로 떠오르고, 일본에서도 사무 생산성 향상을 위해 RPA 도입이 활발함
- 사무업무에 컴퓨터가 도입되면서
  - 응용프로그램 또는 매크로, 엑셀과 같은 도구를 통해 업무의 효율성 제고
  - 웹 스크레이핑, 워크플로우 자동화, 관리도구 및 인공지능과 같은 기술들이 등장하면서 고도화
  - 이러한 기술들을 통합하려는 노력에 의해 RPA라는 통합 솔루션이 등장



## ■ 물리적 로봇 vs 소프트웨어 로봇

- 전통적 산업현장에서 사용되는 물리적 로봇
- RPA에서 사용되는 소프트웨어 로봇

제조 (Smart Factory)



→ 물리적 로봇이 사람의 노동력을 대체

서비스 (RPA기반 Digital Worker)



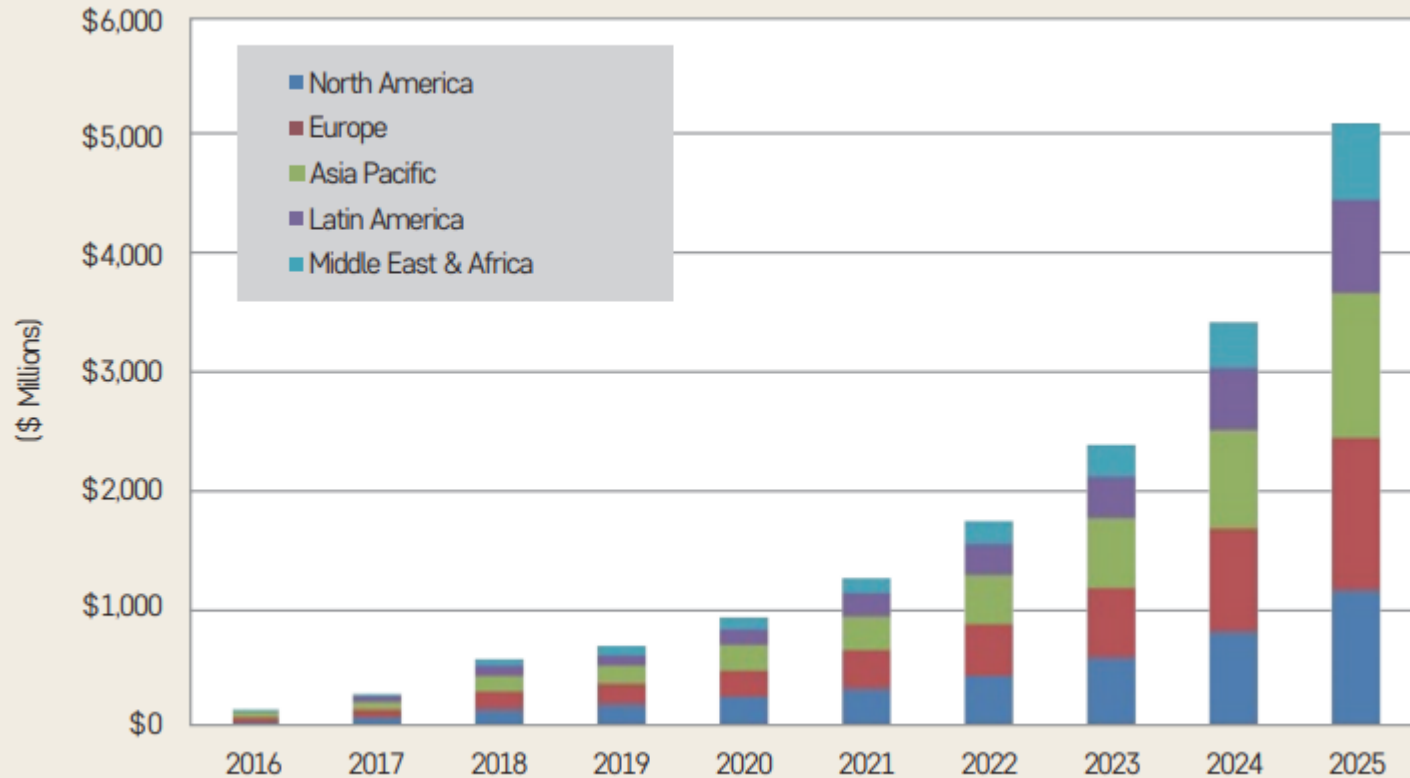
→ 소프트웨어 로봇이 사람의 단순/반복 업무를 대체

## ■ RPA는 필연

- 세계적 저성장 기조 : 투자 대비 효과가 가장 높은, 생산성은 높이고 시간은 줄이는 최적의 솔루션
- 노동인구의 감소 : 한국의 노동인구는 주요 국가 중에서 가장 큰 폭으로 감소하는 추세
- 주 52시간 근무제 시행 : 워라벨을 추구하기 위해 고질적인 장시간 근로 관행을 개선해야 함
- 4차 산업혁명의 도래 : 하드웨어와 소프트웨어의 발전과 보급 (open source화)
- 밀레니엄 세대의 등장
  - 기성세대가 직장에서 '의미'를 추구하는 것에 비해 밀레니엄 세대는 '워라벨'을 추구함

점유율	글로벌 RPA 기업
10~20%	  
5~10%	    
5% 미만	         

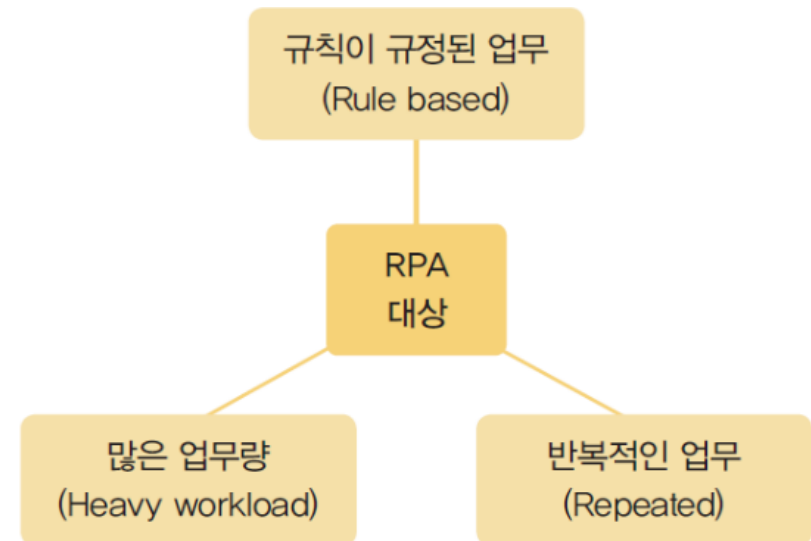
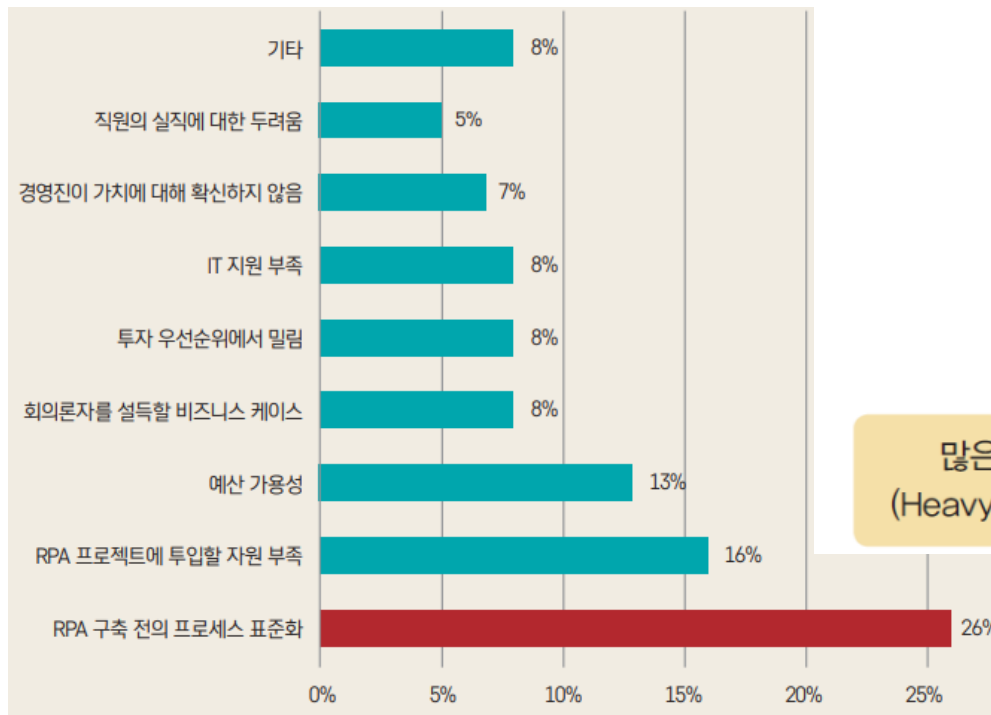
## ■ Tractica에서 전망한 지역별 RPA 매출 전망 (2016~2025)



※ 출처 : Tractica(2017.7.), Robotic Process Automation Market to Reach \$5.1 Billion by 2025

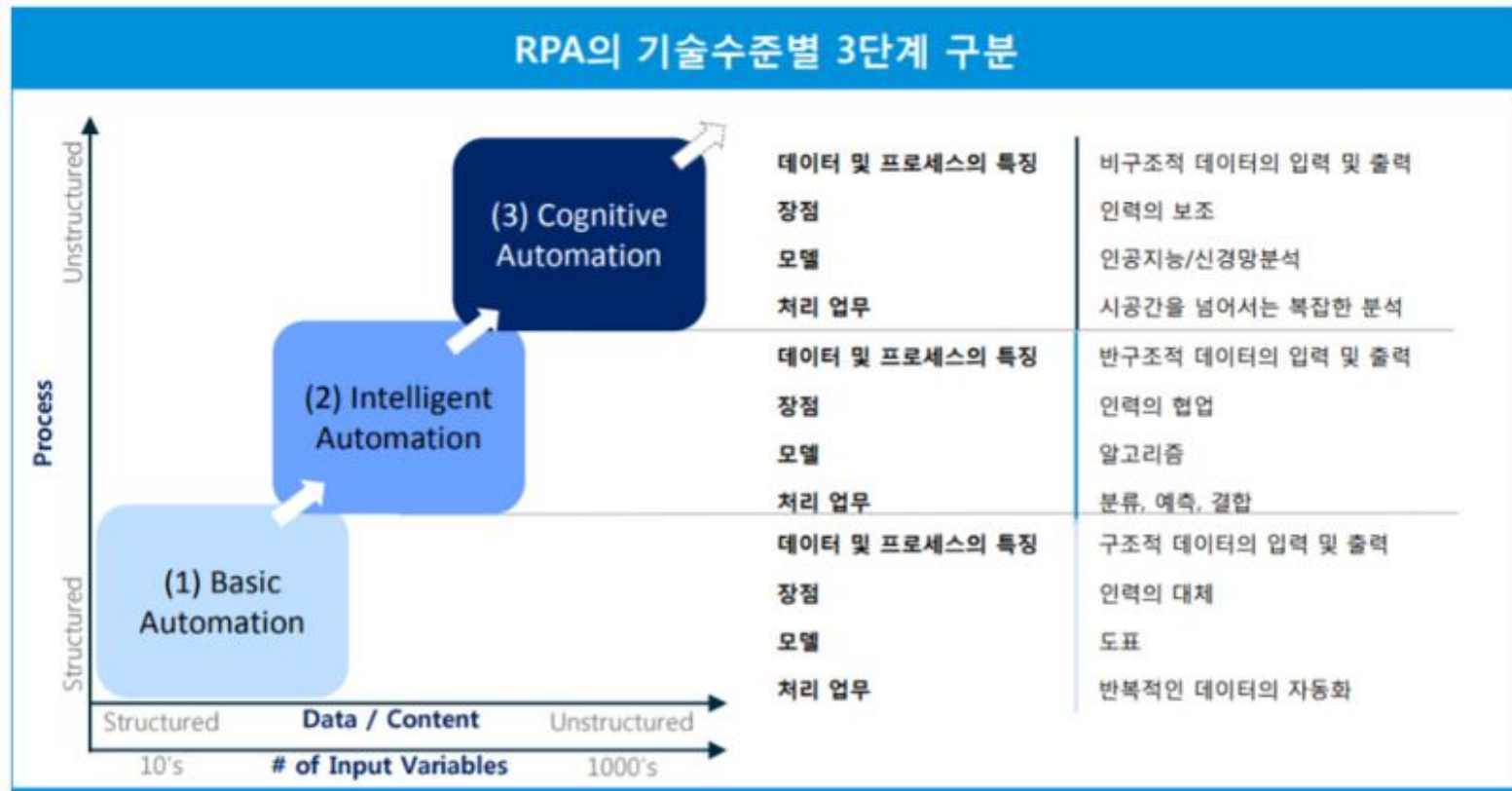
## ■ RPA의 대상

- 인간의 업무를 대신하거나 보조해주기 때문에 가상 지적 노동자 또는 디지털 노동자(digital worker)라고도 함
- 기술적으로 인간의 모든 업무를 자동화할 수 없기 때문에  
RPA 도입의 효과를 극대화할 수 있는 대상업무를 선정하는 것이 필요함
- RPA 도입 시의 주요 **애로사항**과 RPA 대상이 되는 3가지 주요 조건



## ■ RPA의 기술수준별 단계

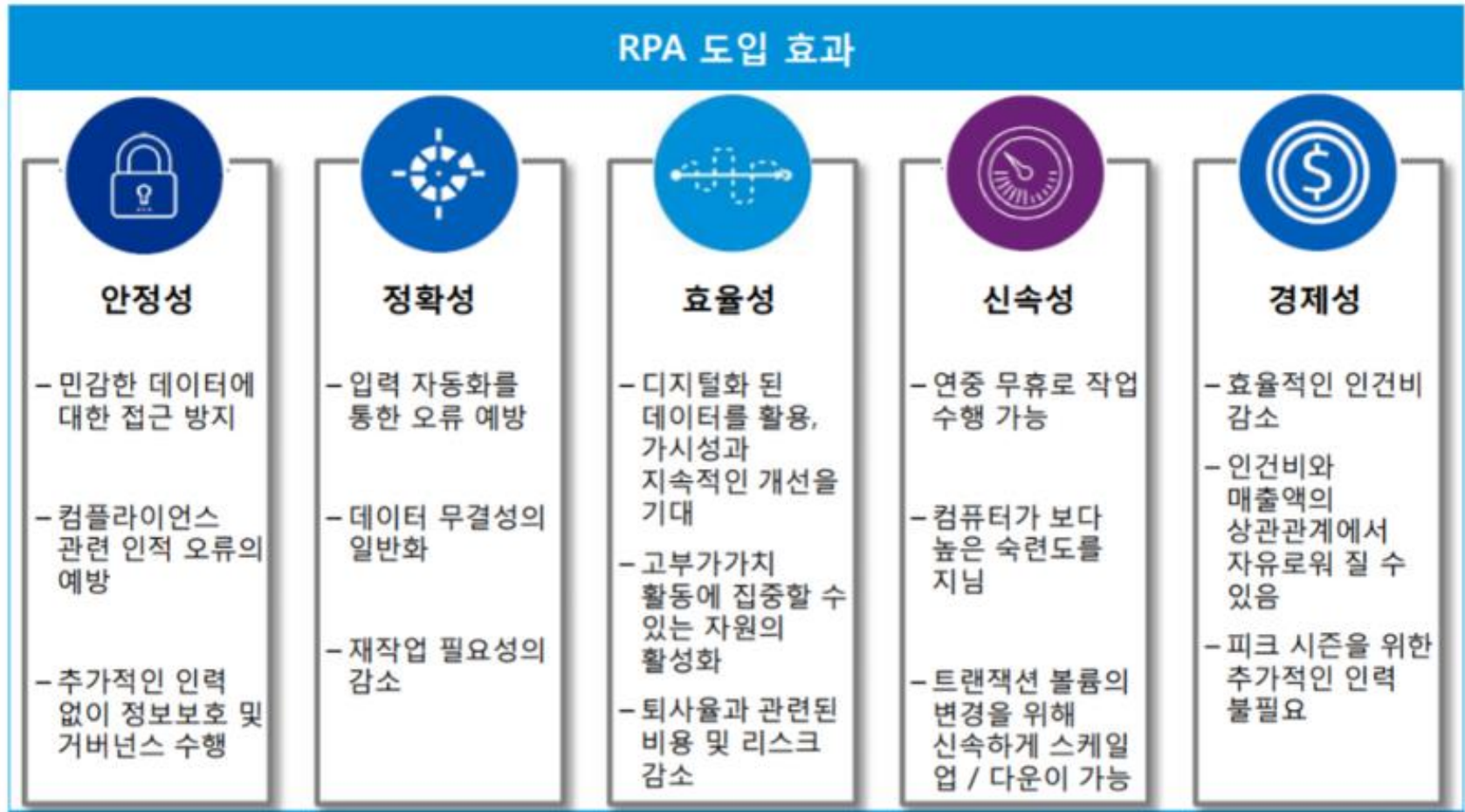
- 도입 초기 : 단순 업무, 반복 업무, 적은 예외
- 성숙기 : 비정형, 복잡한 업무, 인지로봇



Source: KPMG International(2016)



## ■ 도입 효과



Source: KPMG International(2016)

## ■ RPA를 위한 파이썬 모듈 → 프로그래머가 아닌 일반인도 가능

분야	파이썬 모듈	AI리터러시('2302)
엑셀 자동화	<b>openpyxl, win32com</b>	Day1
데이터 전처리, 분석	<b>numpy, pandas, matplotlib</b>	Day2
데이터베이스	sqlite	-
Web Scraping, Crawling	<b>requests, BeautifulSoup</b>	Day3
Web browser 자동화	<b>Selenium</b>	Day3
이메일 자동화	smtplib, EmailMessage	-
보고서 생성 자동화	Jinja2	-
모니터 화면제어(키보드,마우스)	<b>pyautogui</b>	Day3
자동화 통합관리	Robot Framework	-
작업 스케줄링(주기적인 실행)	schtasks(win), crontab(linux)	-
OCR (문자/숫자 인식)	tesseract	-
윈도우(GUI) 프로그래밍	pyQt	-

## ■ 관련 자료

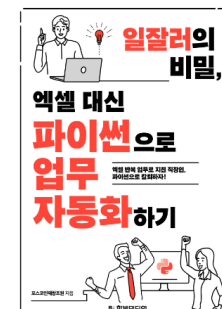
- 인생은 짧아요, 엑셀 대신 파이썬, 이승준, 유튜브
- 일잘러의 비밀, 엑셀 대신 파이썬으로 업무 자동화하기, 한빛미디어
  - 엑셀 반복 업무로 지친 직장인, 파이썬으로 칼퇴하자!
- 파이썬 자동화 교과서, Jpub
  - 업무 생산성을 3배 높이는 엑셀, 워드, 크롤링, 메일 자동화 기술
  - 정시 퇴근과 연봉 인상을 보장하는 업무 자동화 기술
- IT 비전공자를 위한 파이썬 업무 자동화 (RPA)
- 칼퇴하는 일잘러의 업무 스킬, 파이썬 업무 자동화 : 반복업무는 파이썬에게 맡기자~



## 삼성전자 반도체 부문 "문과 신입도 코딩 배워라"

반도체 산업/시사 / 캡틴 홍판판 / 2022. 1. 29. 19:40

삼성전자가 공채 신입사원들에게 직군에 상관없이 코딩을 비롯한 소프트웨어 교육을 실시합니다. 코로나19 이후 기업 환경의 디지털 전환이 가속되면서 모든 직원에게 기본적인 소프트웨어를 활용하는 역량을 요구하는 경영진의 의지가 반영됐습니다. 국내 기업 문화를 선도해온 삼성전자가 '모든 직원의 개발자화'를 추진함에 따라 다른 기업들의 인재 교육 시스템에도 연쇄적인 변화가 일어날 것으로 보입니다.



## ■ 엑셀파일 지원 파이썬 모듈

- xlswt
- OpenPyXL
- XlsxWriter
- PyExcelerate



<http://openpyxl.readthedocs.org>

## ■ openpyxl 모듈

- 엑셀 프로그램이 설치되어 있지 않아도 사용 가능
- 대용량, 이미지 지원 등 엑셀의 모든 기능이 다 있다.
- 문서화가 잘되어 있고, 최근까지 활발하게 업데이트가 되고 있다.
- `pip install openpyxl`

```
import openpyxl
```

```
openpyxl.load_workbook(fname)
```

## ■ win32com 모듈 (MS COM 방식)

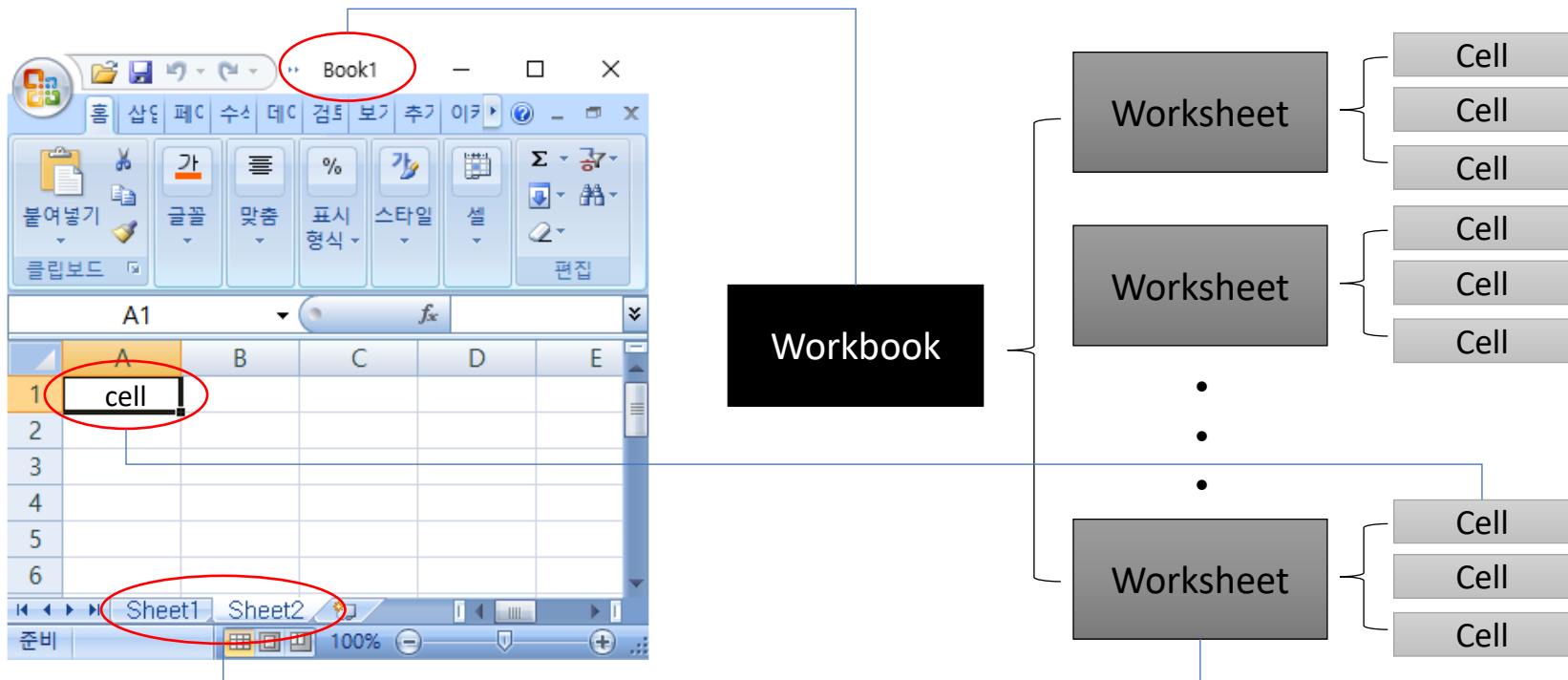
- 엑셀 프로그램이 설치되어 있어야 함
- 성능 우수, openpyxl에 비해 3배 정도 빠름
- COM(component object model) : MS에서 개발한 프로그래밍 언어와 상관없이 사용 가능한 기술

```
import win32com.client    # pip install pywin32
```

```
excel = win32com.client.Dispatch("Excel.Application")  
excel.Visible = False  
excel.Workbooks.Open(fname)  
excel.Quit()
```

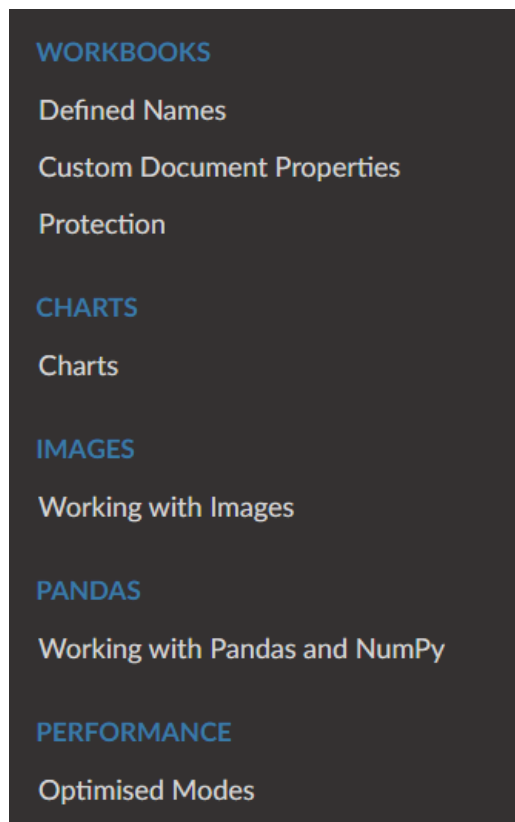
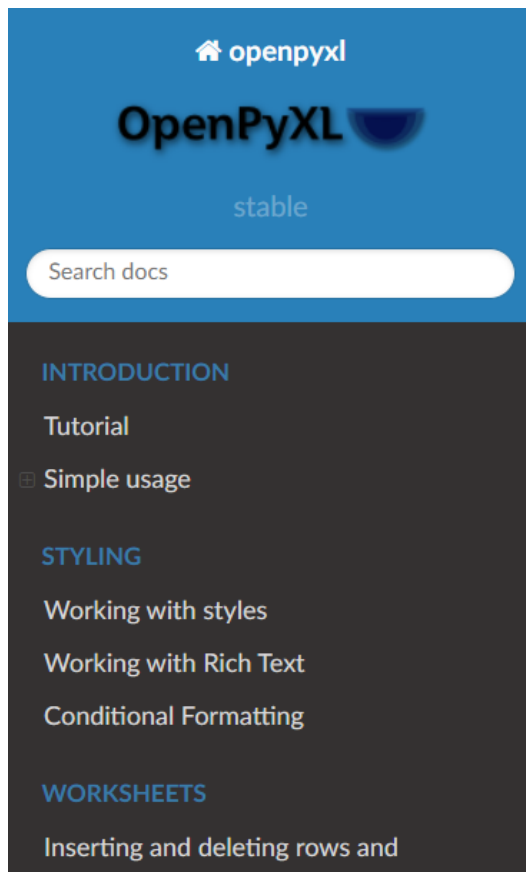
## ■ Workbook(파일), Worksheet(시트), Cell(셀)이라는 객체로 구성

- 엑셀 통합문서, 파일(\*.xlsx)을 workbook이라고 함
- workbook은 하나 이상의 worksheet로 구성됨
- 현재 활성화되어 있는 워크시트를 active sheet라고 함
- Worksheet는 row(행)과 column(열) 구조의 cell로 구성되고, cell에 데이터가 있음



## ■ OpenPyXL을 이용한 엑셀 자동화 실습

- OpenPyXL 사이트 : <https://openpyxl.readthedocs.io/en/stable/tutorial.html>
- 참고 사이트 : <https://goodthings4me.tistory.com/487>



## 목차는 다음과 같다

- 엑셀 파일 만들기
- 엑셀 시트 관리
- 엑셀 셀(cell) 관리
- 엑셀 파일, 셀 데이터 불러오기
- 셀 범위(cell range) 다루기
- 엑셀에서 값 찾기
- 엑셀에서 행, 열 삽입, 삭제, 이동
- 엑셀 차트(Chart) 다루기
- 엑셀 셀 스타일(Style) 다루기
- 엑셀 수식(함수) 활용해보기
- 엑셀에서 수식(데이터) 가져오기
- 엑셀 셀(Cell) 병합
- 엑셀에 이미지 추가

## ■ 엑셀문서(Workbook) 열기

## ■ 시트(Worksheet) 접근

- 시트 목록을 리스트 반환
- Active Sheet 가져오기
- 특정 Sheet 가져오기

```
import openpyxl
```

```
wb = openpyxl.load_workbook('재고장.xlsx')
```

```
wb.sheetnames # ['물류재고장', '입출고내역', '랙번호']
```

```
ws = wb['물류재고장']
```

```
ws = wb.worksheets[0]
```

```
ws = wb.active # 현재 활성화된 sheet를 가져옴
```

```
row = ws[2:5]
```

```
col = ws['A:C']
```

```
area = ws['A1:C3']
```

## ■ 셀 접근

```
ws['A1']
```

# 셀자체(객체)

```
ws['A1'].value
```

# 셀내용(값)

```
c = ws['A1']
```

```
c.value
```

# 셀의 값(내용) : '랙번호'

```
c.coordinate
```

# 셀의 좌표(위치) : A1

```
'Cell ' + c.coordinate + ' is ' + c.value
```

```
→ 'Cell A1 is 랙번호'
```

## ■ cell()함수를 이용하여 index로 접근 → 루프를 돌릴 때 유리함

- 행수 : ws.max\_row
- 열수 : ws.max\_column

# Cell 함수로 Cell에 접근하기

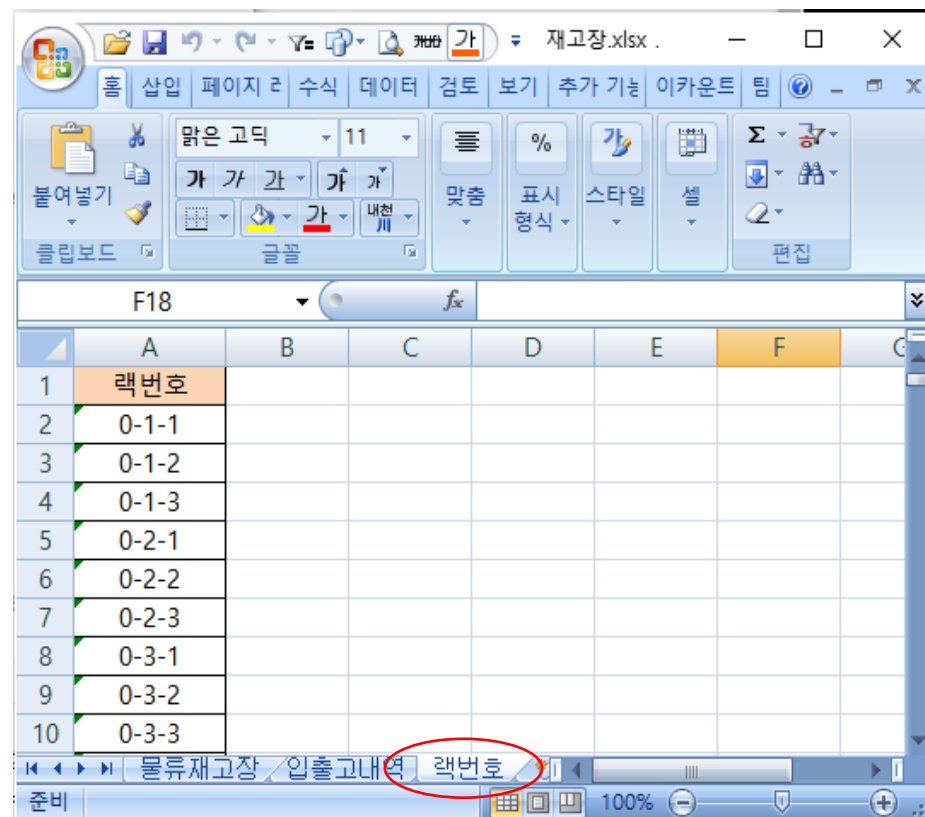
**ws.cell(row=1, column=1)**

ws.cell(1, 1).value

ws.cell(1, 1).coordinate

```
for i in range(1, 8, 2):
    print(i, ws.cell(i, 2).value)
```

1	랙번호
3	0-1-2
5	0-2-1
7	0-2-3





## ■ 새 문서 만들고 저장하기

- 워크북 생성 : 1개의 'Sheet'가 생성됨

```
from openpyxl import Workbook
```

```
wb = Workbook() # 새로운 엑셀파일 생성
```

```
wb.sheetnames # default로 'Sheet' 시트 생성
```

```
ws = wb.active # 현재 활성시트
```

```
ws.title = 'Stock' # sheetname 변경
```

```
wb.create_sheet('가나다')
```

# 마지막 위치에 '가나다' 시트 추가

```
wb.create_sheet('시트2', 1)
```

# 해당위치 뒤에 '시트2' 시트 추가

```
ws['A1'].value = 'Rack'
```

```
# ws['A1'] = 'Rack'
```

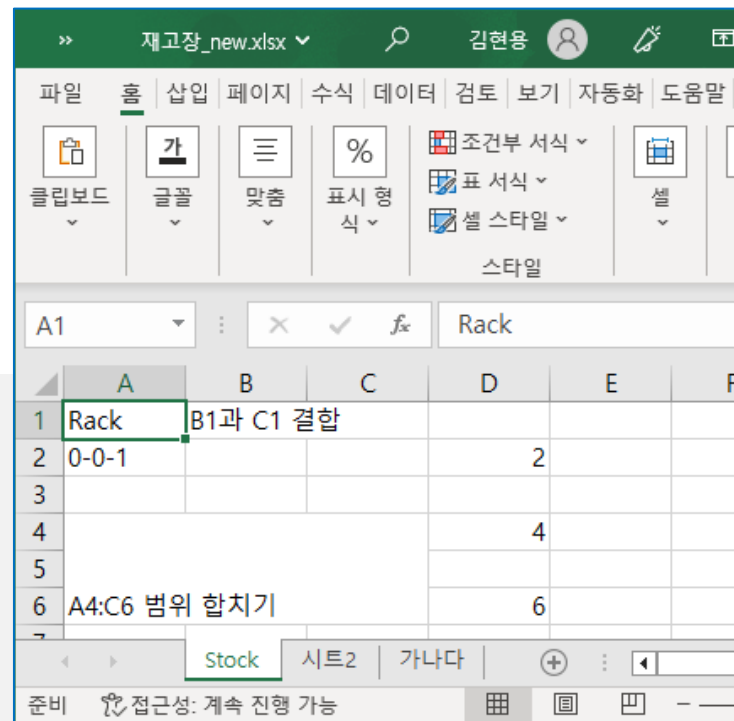
# value 생략 가능

```
ws.cell(1,1).value = 'Rack'
```

# 인덱스가 (1,1)부터 가능, value 생략불가

```
wb.save('재고장_new.xlsx')
```

# 저장



## ■ merge\_cells() 함수를 사용하여 셀 병합/해제

```
ws.merge_cells('B1:C1')
ws['B1'].value = 'B1과 C1 결합'
ws.merge_cells('A4:C6')
ws['A4'].value = 'A4:C6 범위 합치기'

ws.unmerge_cells('A4:C6')
```

## ■ 수식(formulas) 사용하기

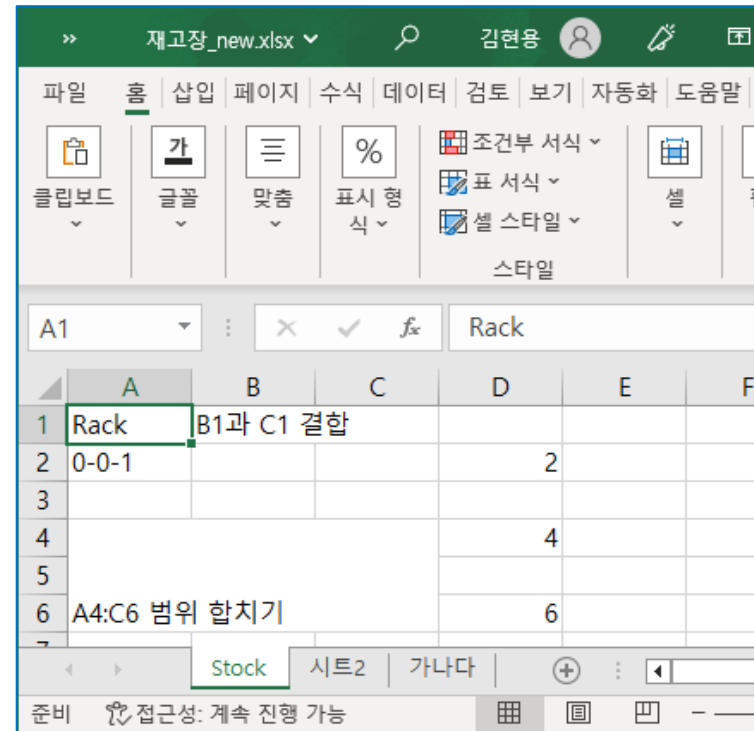
```
ws['D6'].value = '=SUM(D1:D5)'
```

## ■ 행, 열 삽입/삭제/이동

```
ws.insert_rows(8)      # 8번째 행 삽입 : insert_rows(idx, amount)
ws.insert_rows(8, 5)   # 8번째 행에서 아래로 5열 추가
ws.insert_cols(2, 3)   # B번째 열에서 우측에 3열 추가

ws.delete_rows(8)      # 8번째 행 삭제
ws.delete_cols(2, 3)   # B번째 열에서 우측 3열 삭제

ws.move_range('B1:C11', rows=0, cols=1) # 행은 그대로, 열을 1칸 이동
```



## ■ 엑셀파일을 읽어와서 2차원 리스트에 할당하기

```
from openpyxl import load_workbook

wb = load_workbook( ' 재고장.xlsx ' )
#ws = wb.active
ws = wb[ ' 물류재고장' ]

data = []
for row in range(ws.max_row):
    line = []
    for col in range(ws.max_column):
        line.append( ws.cell(row+1, col+1).value )
    data.append(line)
```

	A	B	C	D	E	F	G
1	랙번호	부품코드	부품명	공정	재고량	주문량	생산일자
2	0-1-2	SV0185	헤라 로지-사틴크림 50ML 용기 내용기 증착	증착	9,494	5,000	2018-04-01
3	0-2-2	SC0086	한울 어린숙수분진정크림 80ML(18한정 캡 외캡 무광코팅	코팅	4,475	4,000	2018-03-14
4	0-2-2	SI1017	한울 어린숙수분진정크림 80ML(18한정 캡 내캡 사출	사출	3,718	5,000	2018-04-16
5	0-9-3	1940400015	PUSH-PHONE CAP 태평양-35p 10각형(반투명)	완제품	7,000	2,000	2018-06-07
6	1-1-1	21051450018	kimchi keeper handle	완제품	160	500	2018-02-21
7	2-1-1	21051450017	HAPPY KIMCHI KEEPER PLAIN H	완제품	888	200	2018-05-30
8	1-1-1	7245842	아리따움 뽀얀미소발효클렌징크림 200ml 환판	완제품	1,000	1,000	2018-04-04
9	1-2-1	7334281	중APC)려함빛극손상케어트200ML(가느모발)스크류캡	완제품	741	300	2017-04-20
10	1-1-2	7233467	미장센 버블컬러산화제 60ML 캡(신)	완제품	15,000	5,000	2018-05-21
11	1-1-3	7338317	설화수 탄력크림(견 5ML(15AD) 캡	완제품	20,000	10,000	2018-06-01
12	1-1-6	1940400013	PUSH-PHONE CAP태평양-35파이 (10각큰캡)	완제품	5,000	3,000	2018-05-10

## ■ 숫자 서식 지정하기 <https://openpyxl.readthedocs.io/en/stable/styles.html#introduction>

- cell의 number\_format 속성 : 0 (zero 표시) # (zero 제거 suppression)

## ■ 폰트 설정 : Font 객체 – name, size, bold, italic, underline, strike(취소선), color

## ■ 셀 채우기 : PatternFill 객체 – patternType, fgColor, bgColor, ...

## ■ 텍스트 정렬

- **Alignment 객체** : horizontal, vertical, text\_rotation, wrap\_text, shrink\_to\_fit, indent  
horizontal = general / left / center / right / fill / centerContinuous / justify(양쪽맞춤) / distributed(균등분할)  
vertical = bottom / center / top / justify / distributed

```
font = Font(name='Calibri',  
            size=11,  
            bold=False,  
            italic=False,  
            vertAlign=None,  
            underline='none',  
            strike=False,  
            color='FF000000')
```

## ■ 테두리 지정하기

- **Side 객체** : style='thin/medium/thick/dotted/dashDot/slantDashDot', color
- **Border 객체** : left, right, top, bottom, diagonal, diagonal\_direction, outline, vertical, horizontal

```
cell.number_format = '#,##0'           # 천 단위 쉼표로 자릿수 표시  
cell.number_format = '#,##0.00'        # 소수점 이하 자릿수 표시  
cell(i, 5).font = Font(bold=True)  
cell.fill = PatternFill(patternType='solid', fgColor="AA8866")  
cell.alignment = Alignment(horizontal='distributed')  
side = Side(style='thick', color='00FF00')  
cell.border = Border(left=side, right=side, top=side, bottom=side)
```

## ■ 셀서식 지정하기

C19							
	A	B	C	D	E	F	G
1	랙번호	부품코드	부품명	공정	재고량	주문량	생산일자
2	0-1-2	SV0185	헤라 로지-사틴크림 50ML 용기 내용기 증착	증착	9,494	5,000	2018-04-01
3	0-2-2	SC0086	한울 어린숙수분진정크림 80ML(18한정 캡 외캡 무광코팅	코팅	4,475	4,000	2018-03-14
4	0-2-2	SI1017	한울 어린숙수분진정크림 80ML(18한정 캡 내캡 사출	사출	3,718	5,000	2018-04-16
5	0-9-3	1940400015	PUSH-PHONE CAP 태평양-35φ 10각형(반투명)	완제품	7,000	2,000	2018-06-07
6	1-1-1	21051450018	kimchi keeper handle	완제품	160	500	2018-02-21
7	2-1-1	21051450017	HAPPY KIMCHI KEEPER PLAIN H	완제품	888	200	2018-05-30



F8							1000
	A	B	C	D	E	F	G
1	랙번호	부 품 코 드	부 품 명	공 정	재 고 량	주 문 량	생 산 일 자
2	0-1-2	SV0185	헤라 로지-사틴크림 50ML 용기 내용기 증착	증착	9,494	5,000	2018-04-01
3	0-2-2	SC0086	한울 어린숙수분진정크림 80ML(18한정 캡 외캡 무광코팅	코팅	4,475	4,000	2018-03-14
4	0-2-2	SI1017	한울 어린숙수분진정크림 80ML(18한정 캡 내캡 사출	사출	3,718	5,000	2018-04-16
5	0-9-3	1940400015	PUSH-PHONE CAP 태평양-35φ 10각형(반투명)	완제품	7,000	2,000	2018-06-07
6	1-1-1	21051450018	kimchi keeper handle	완제품	<b>160</b>	500	2018-02-21
7	2-1-1	21051450017	HAPPY KIMCHI KEEPER PLAIN H	완제품	<b>888</b>	200	2018-05-30

```
import openpyxl
from openpyxl.styles import Alignment, PatternFill, Font, Border, Side

TITLE_CELL_COLOR = "AA8866" # 상수

wb = openpyxl.load_workbook('재고장.xlsx')
ws = wb['물류재고장']
# ws.append(['1-2-3', 'SI1234', '추가된 사출성형 품목', '사출', 6000, '2021-03-29'])

ws.freeze_panes = "C2" # 틀고정 : ws.freeze_panes = 'A1' , None (고정없음)

col_widths = {'A':8, 'B':13, 'C':50, 'D':10, 'E':10, 'F':15} # 열 크기 지정
for pos, width in col_widths.items():
    ws.column_dimensions[pos].width = width

# 행 높이 지정
for i in range(1, ws.max_row+1): # ws.row_dimensions[1].height = 30
    ws.row_dimensions[i].height = 20 #
    ws.cell(i, 5).number_format = '#,##0'
    if i != 1 and int(ws.cell(i, 5).value) < 1000:
        ws.cell(i, 5).font = Font(bold=True)
```

*# 폰트 지정*

```
font_header = Font(name='맑은 고딕', size=12, bold=True, color='FFFFFF')
for cols in ws['A1':'F1']:
    for cell in cols:
        cell.fill = PatternFill(patternType='solid', fgColor=TITLE_CELL_COLOR)
        cell.alignment = Alignment(horizontal='distributed')
        cell.font = font_header
```

*# 셀 테두리 지정*

```
side = Side(style='thin', color='000000')
border = Border(left=side, right=side, top=side, bottom=side)
for row in ws:
    for cell in row:
        cell.border = border
```

```
# ws.column_dimensions['A'].hidden = True
# # ws.column_dimensions['A'].hidden = False
# ws.row_dimensions[1].hidden = True
```

```
# 열 숨기기
# 숨긴 열 표시하기
# 행 숨기기
```

```
wb.save('재고장_new.xlsx')
wb.close()
```

## ■ 차트 그리기 절차 (<https://openpyxl.readthedocs.io/en/stable/charts/introduction.html>)

- 셀의 직사각형 선택영역으로부터 Reference 객체를 만든다.
- Chart 객체를 만든다.
- 워크시트 개체에 Chart 객체를 추가한다.

```
from openpyxl import Workbook
from openpyxl.chart import BarChart, Reference
```

```
wb = Workbook()
ws = wb.active
for i in range(10):
    ws.append([i])
```

```
data = Reference(ws, min_col=1, min_row=1, max_col=1, max_row=10)
```

```
# labels = Reference(ws, 1, 1, 1, 10)
```

```
chart = BarChart()
```

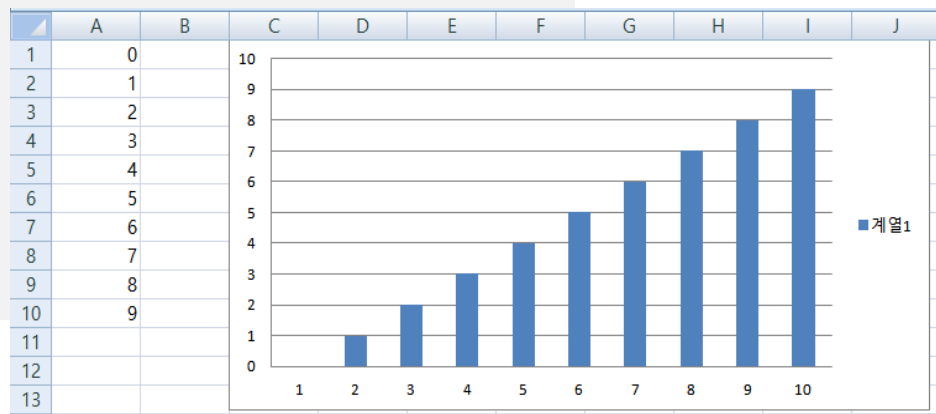
```
chart.add_data(data)
```

```
# chart.set_categories(labels)
```

```
ws.add_chart(chart, "C1")
```

```
wb.save("SampleChart.xlsx")
```

```
wb.close()
```





## 재고장.xlsx

	A	B	C	D	E	F	G
1	랙번호	부품코드	부품명	공정	재고량	주문량	생산일자
2	0-1-2	SV0185	헤라 로지-사틴크림 50ML 용기 내용기 증착	증착	9,494	5,000	2018-04-01
3	0-2-2	SC0086	한울 어린숙수분진정크림 80ML(18한정 캡 외캡 무광코팅	코팅	4,475	4,000	2018-03-14
4	0-2-2	SI1017	한울 어린숙수분진정크림 80ML(18한정 캡 내캡 사출	사출	3,718	5,000	2018-04-16
5	0-9-3	1940400015	PUSH-PHONE CAP 태평양-35φ 10각형(반투명)	완제품	7,000	2,000	2018-06-07
6	1-1-1	21051450018	kimchi keeper handle	완제품	160	500	2018-02-21
7	2-1-1	21051450017	HAPPY KIMCHI KEEPER PLAIN H	완제품	888	200	2018-05-30
8	1-1-1	7245842	아리따움 뽀오얀미소발효클렌징크림 200ml 환판	완제품	1,000	1,000	2018-04-04
9	1-2-1	7334281	중APC)려함빛극손상케어트200ML(가는모발)스크류캡	완제품	741	300	2017-04-20
10	1-1-2	7233467	미장센 버블컬러산화제 60ML 캡(신)	완제품	15,000	5,000	2018-05-21
11	1-1-3	7338317	설화수 탄력크림(건 5ML(15AD) 캡	완제품	20,000	10,000	2018-06-01
12	1-1-6	1940400013	PUSH-PHONE CAP태평양-35파이 (10각큰캡)	완제품	5,000	3,000	2018-05-10

```
import openpyxl
from openpyxl.chart import Reference, BarChart, LineChart, AreaChart, PieChart, RadarChart

wb = openpyxl.load_workbook('재고장.xlsx')
ws = wb['물류재고장']

# chart 데이터 참조범위 지정
data = Reference(ws, min_col=5, min_row=1, max_col=6, max_row=12)
labels = Reference(ws, 2, 2, 2, 12) # Reference(ws, 5, 2, 5, 12)
```

```
# 차트 종류 지정
# chart = BarChart()
# chart.type = 'bar'
# chart = LineChart()
# chart = AreaChart()
# chart = PieChart()
chart = RadarChart()

# 막대그래프
# 가로(bar), 세로(col) - 디폴트값
# 꺾은 선 그래프
# 영역형 그래프
# 원형 차트
# 방사형 차트
```

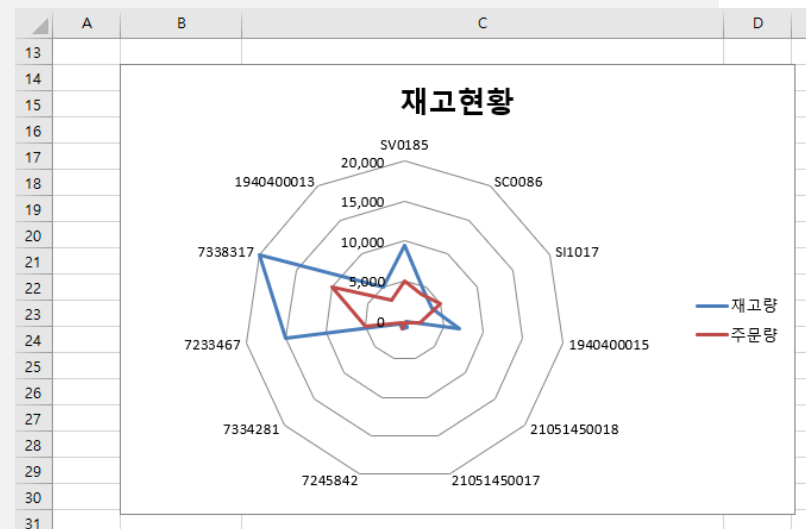
```
chart.grouping = 'stacked' # 누적
# chart.overlap = 100      # for barchart: 100
```

```
chart.add_data(data, titles_from_data=True) # title...=True: 첫 셀값을 계열값(범례)로 사용
chart.set_categories(labels)
```

```
chart.title = '재고현황'      # 차트 제목
chart.x_axis.title = '랙번호'  # x축 제목
chart.y_axis.title = '재고량'  # y축 제목

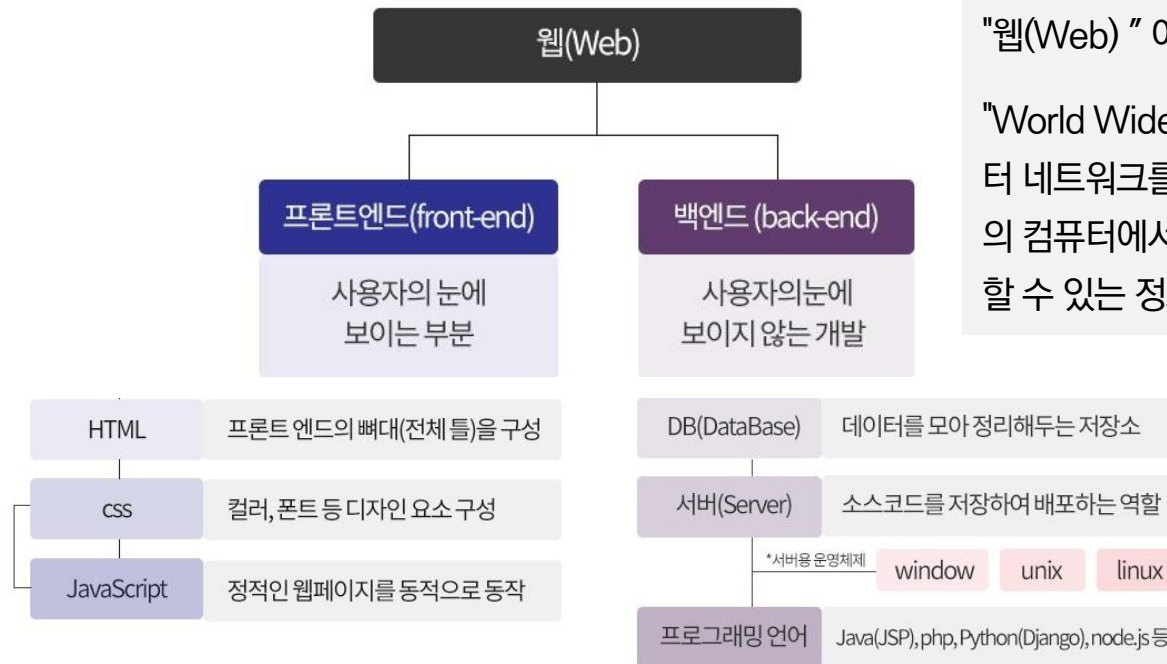
chart.height = 10             # 차트 가로 크기
chart.width = 20              # 차트 세로 크기
ws.add_chart(chart, 'B14')    # chart를 B14에 삽입
```

```
wb.save('재고장_radarchart.xlsx')
wb.close()
```



## ■ Web Programming

- WWW(world wide web) 통신규약에 따라 컴퓨터 간의 데이터를 주고받는 네트워크 프로그램 즉, 웹을 만드는 것
- 프론트엔드(front-end) : 주로 HTML, JavaScript(js), css 등을 이용하여 웹 페이지를 구성하고, 사용자가 이용할 수 있는 인터페이스와 상호작용하는 기능들을 구현
- 백엔드(back-end) : 프론트엔드에서 전달받은 데이터를 DB에 저장하고 처리하며, 인증, 보안 등의 기능을 구현
- 풀스택(full-stack) : 프론트엔드 + 백엔드 모두를 가리킴

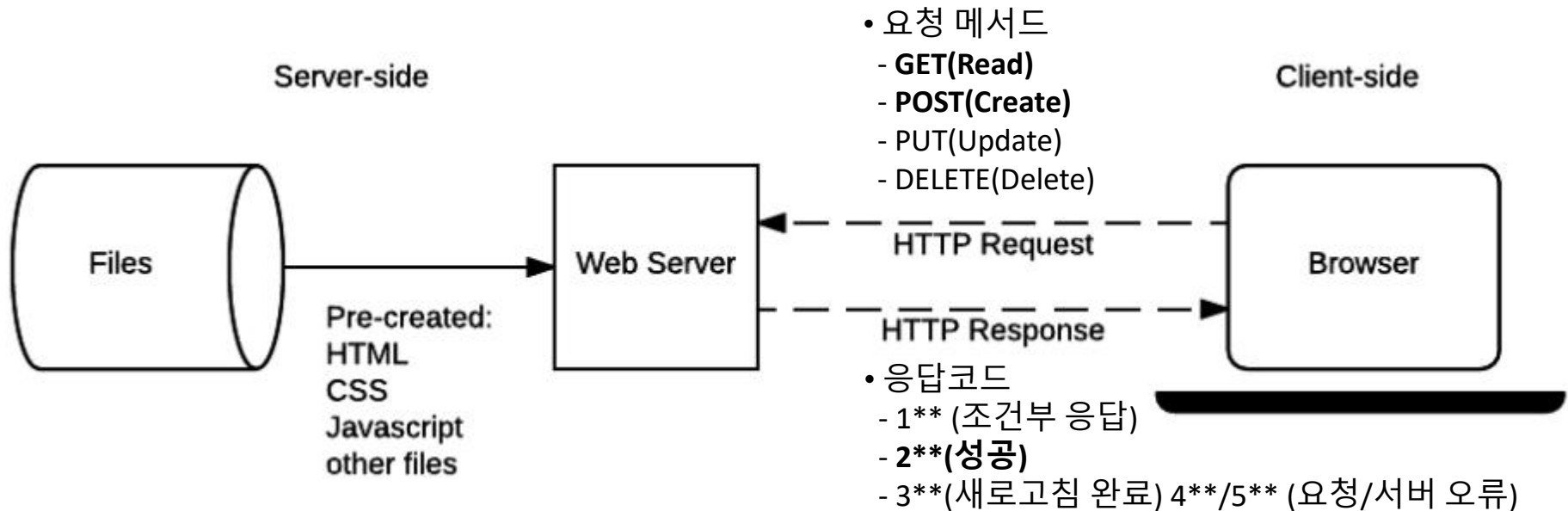


"웹(Web) " 이란?

"World Wide Web"의 약자로, 컴퓨터 네트워크를 통해 연결된 전 세계의 컴퓨터에서 정보를 공유하고 검색할 수 있는 정보 공간을 의미

## ■ HTTP(HyperText Transfer Protocol)

- 인터넷 상에서 서버와 클라이언트가 정보를 주고받는 데 사용되는 규약
- 클라이언트 (Client) 는 웹 브라우저를 통해 HTTP 요청(request) 메시지를 보내고,
- 웹 서버 (Server) 는 HTTP 응답(response) 메시지를 클라이언트에게 보냄
- URL(uniform resource locator) : 특정 웹페이지나 문서, 그림, 동영상 등의 파일에 접근하기 위한 주소  
scheme://host:port/path?query#fragment 의 구조로 되어 있음




## ■ HTML (HyperText Markup Language)

- 웹 페이지를 작성하기 위한 마크업 언어로써, 다양한 태그를 사용하여 웹 페이지의 구조와 내용을 정의
- HyperText : 일반 텍스트와 달리 링크를 클릭하면 다른 페이지로 이동하는 것처럼 서로 연결(링크)된 문자 데이터
- Markup 언어 : 다양한 문서나 데이터의 구조를 표현하는 언어

## ■ HTML 문서의 구성

- Tag가 쌍으로 존재, 예) <html> ... </html>, <td> ... </td>
- 속성 : 속성값, 예) border=1
- 텍스트



항목	2013	2014	2015
매출액	100	200	300

```
<html>
<table border=1>
  <tr>
    <td> 항목 </td>
    <td> 2013 </td>
    <td> 2014 </td>
    <td> 2015 </td>
  </tr>
  <tr>
    <td> 매출액 </td>
    <td> 100 </td>
    <td> 200 </td>
    <td> 300 </td>
  </tr>
</table>
</html>
```

## ■ XML (Extensible Markup Language)

- HTML과 같은 마크업 언어지만 데이터 저장과 전송을 위해 고안된 언어로 웹 이외의 환경에서도 사용 가능

HTML 태그	설명	예시
<!doctype html>	문서유형 지정. 이 페이지를 html로 해석하라	
<html>	웹페이지의 시작과 끝 나타냄	
<!-- 주석내용 -->	주석	
<head>	: 웹페이지의 정보 즉, <title>, <meta>, <style> 등이 포함	
<title>	문서 제목으로 웹브라우저의 제목 표시줄에 표시	
<meta>	문자 인코딩 및 문서 키워드, 요약 정보	<meta charset="utf-8">
<style>	스타일 정의	
<body>	브라우저에 실제로 표시되는 내용	
<h1> ~ <h6> 제목	제목(headline) 글씨로 h1이 가장 크다.	
<div> , <span>	구역분할 : division은 줄바꾸고, span은 줄바꿈 없음	
<a>	anchor, 웹 페이지나 외부 사이트 연결해주는 hyperlink를 만드는 태그. href 속성으로 링크 대상을 지정함	<a href="https://google.com"> 구글 검색</a>
<script>	실행가능한 코드(Javascript)를 삽입	
<img src='경로'>	이미지 삽입. src 속성을 사용하여 이미지 파일의 경로를 지정	
<p>	단락(paraphrase)을 구분하는 것으로 줄이 생김	
<li>	<ul>(unordered list)과 <ol>(ordered list) 안에서 각 항목을 추가할 때 사용	
<table>	표를 만듦. <tr> 태그로 행을 만들고 <td>로 셀을 추가	
 	줄바꿈(line-break), 닫는 태그는 없다.	
<input>	입력	

## ■ Open API (application programming interface)

- 개발한 프로그램을 외부인이 사용할 수 있도록 만든 인터페이스 → 쉽게 정보를 활용 가능
- 규모있는 웹 서버의 경우, 서비스를 일반인들이 활용할 수 있도록 Open API를 제공하고 있음
- 홈페이지 업그레이드 등의 변화에 대해서도 웹 스크래핑에 비해 안정적인 서비스를 받을 수 있음

## ■ 네이버 오픈 API 사용 절차 : 네이버 개발자(Developers) <https://developers.naver.com/main/>

- naver 개발자 센터에서 [애플리케이션 등록]을 하면서 발급받은 Client ID와 Secret값을 부여받아야 함.
- response는 key : value 쌍의 JSON 형식

내 애플리케이션

py-auto

애플리케이션 등록

API 제휴 신청

계정 설정

py-auto

개요

API 설정

멤버관리

로그인 통계

API 통계

Playground(Beta)

애플리케이션 정보

Client ID

CIMDxL67nDjpY40hmi5L

Client Secret

.....

보기

■ Papago 번역 (<https://developers.naver.com/docs/papago/papago-nmt-example-code.md>)

NAVER Developers

Products

Documents

Application

NAVER D2

Support

Forum

파파고

Papago 번역

개요

API 레퍼런스

구현 예제

언어 감지

한글 인명-로마자 변환

- Papago 번역 API 구현 라인 수
  - Java 88줄
  - PHP 27줄
  - Node.js 26줄
  - **python 18줄**
  - C# 35줄

Python ↗

```
import os
import sys
import urllib.request
client_id = "YOUR_CLIENT_ID" # 개발자센터에서 발급받은 Client ID 값
client_secret = "YOUR_CLIENT_SECRET" # 개발자센터에서 발급받은 Client Secret 값
encText = urllib.parse.quote("반갑습니다")
data = "source=ko&target=en&text=" + encText
url = "https://openapi.naver.com/v1/papago/n2mt"
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
response = urllib.request.urlopen(request, data=data.encode("utf-8"))
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)
```

- [GitHub에서 보기](#)



```

import urllib.request
import json
from myinfo import ID, PW

client_id = ID['naver_developer']          # 개발자센터에서 발급받은 Client ID 값
client_secret = PW['naver_developer']      # 개발자센터에서 발급받은 Client Secret 값
encText = urllib.parse.quote("반갑습니다. 빼앗긴 들에도 봄은 온다.")
data = "source=ko&target=en&text=" + encText
url = "https://openapi.naver.com/v1/papago/n2mt"
request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)

response = urllib.request.urlopen(request, data=data.encode("utf-8"))
rescode = response.getcode()

if(rescode==200):
    message = response.read()               # 2진 문자열
    str_message = message.decode('utf-8')  # json 문자열
    json_message = json.loads(str_message) # dict
    print('번역 결과 :', json_message['message']['result']['translatedText'])
else:
    print("Error Code:" + rescode)

```

```
번역 결과 : Nice to meet you. Spring comes even in the fields taken away.
```

## ■ 웹 스크래핑의 개요

- 용어 정의
  - 웹 스크래핑(Scraping) : 웹 사이트에 있는 특정 정보를 추출하는 기술
  - 웹 크롤링(Crawling) : 웹 사이트를 여기저기 이동하며 정보를 추출하는 기술
  - 웹에서 전문적으로 정보를 수집해주는 프로그램을 스크레이퍼, 크롤러, 스파이더 등이라고도 함
- 대량의 데이터를 얻을 수 있는 가장 효율적인 방법
- 웹 스크레이핑이 웹사이트의 저작권이나 서비스 이용약관을 위반하는지에 대한 법적인 문제가 발생할 수 있음



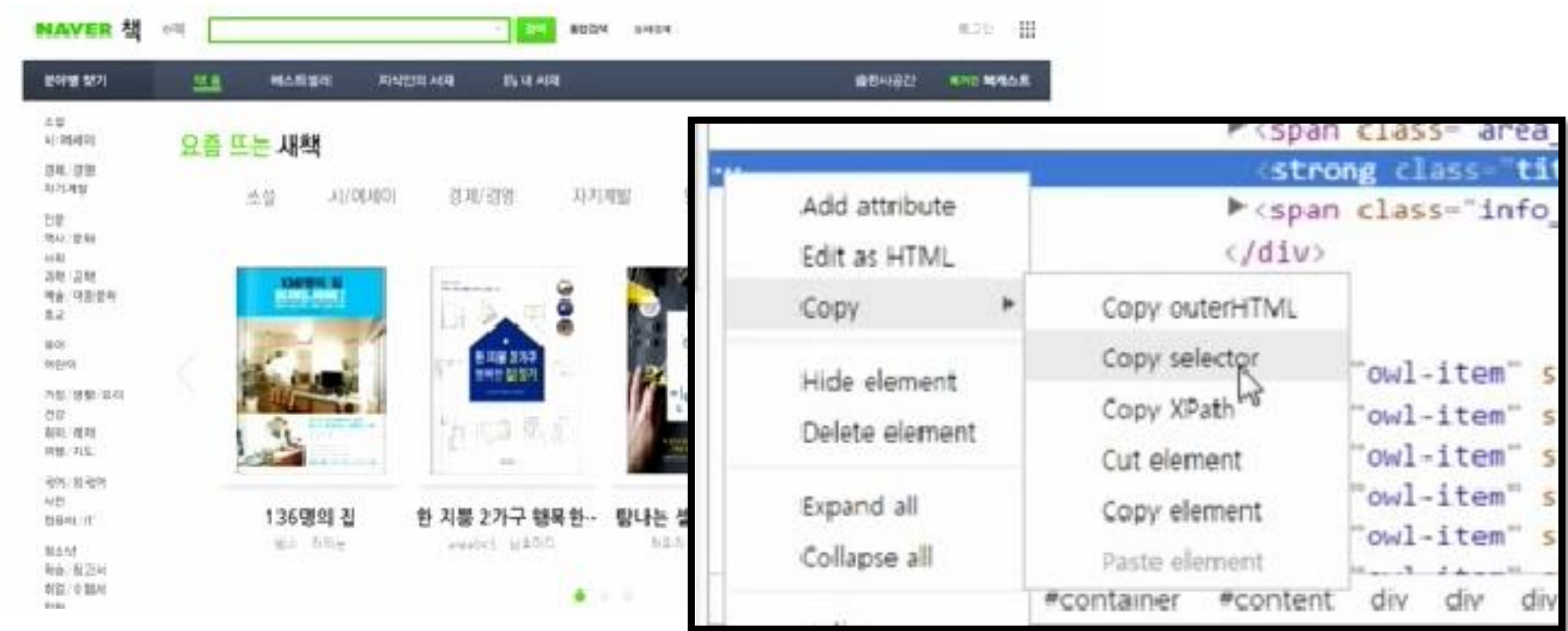
## ■ 웹 스크래핑의 절차

- 1단계 : URL 주소(도메인 이름, IP 주소)에 접속하여 HTML 문서를 가져옴
- 2단계 : HTML 문서를 파싱(parsing, 구문분석)하여 원하는 정보를 추출  
웹 페이지는 HTML 형식이기 때문에 문서의 구조를 분석하는 과정이 중요
- 3단계 : 원하는 정보를 저장
- 4단계 : 필요하다면 다른 페이지에서도 이 작업을 반복함 → Web crawling  
하이퍼링크라는 'a'라는 tag문자로 시작하는 문자열을 검색하여 다른 웹페이지로 이동



## ■ 개발자 도구 활용

- 웹 브라우저에서 먼저 원하는 데이터를 포함하는 태그를 찾기 위한 편리한 툴
- explorer : F12
- chrome : F12, 도구 더보기 > 개발자 도구
- 해당 html 소스 확인 > copy > copy selector / (full) XPath



## ■ requests 모듈

- HTTP 요청(request)을 쉽게 처리해주는 라이브러리
- 설치 : pip install requests <http://docs.python-requests.org/en/master/>
- 주요 메서드 : requests.get / post / put / delete



## ■ BeautifulSoup4 모듈

- HTML 및 XML 파일을 파싱하여 데이터를 추출하기 위한 라이브러리
- 설치 : pip install beautifulsoup4 → from bs4 import BeautifulSoup
- <https://www.crummy.com/software/BeautifulSoup/bs4/doc>
- 파싱방법 : html.parser(내장모듈), lxml(빠르고 우수함, third party 모듈)



## ■ soup.find(태그, 속성)와 find\_all(태그, 속성)

- 첫번째 요소와 모든 요소를 검색하는 함수
- HTML 페이지에서 원하는 태그를 다양한 속성에 따라 쉽게 필터링할 수 있음

- 복잡한 계층구조 예) `#idname > div > div > div:nth-child(1) > a > div > strong`  
 `#(id 속성): html에서 유일해서 첫번째 위치를 잡는 데 활용 (idname : id명)`

```
soup.find('div')                # Tag로 찾기
soup.find_all('div')
soup.find('a')
soup.find('a', href='#newsstand') # Tag와 Attributes로 찾기
soup.find('a', {'href': '#newsstand'})
soup.find(href='#newsstand')     # Attributes로 찾기
soup.find(id='wrap')
```

## ■ re 모듈 (내장모듈)

- 정규 표현식(regular expression)을 다루는 모듈
- 정규 표현식은 문자열에서 특정 패턴을 찾거나 원하는 형태로 변형할 때 사용
- re.findall(찾으려는 패턴, 찾을 대상) → 리스트 형태로 반환 `?(0 or 1), *(0 or many), +(1 or many), {n} (n번)`

```
data_list = re.findall('[+-]?[0-9]+[.0-9]*', '<span class="todaytemp">-9.5</span>')
float(data_list[0])    # data_list = ['-9.5']
```

## ■ HTML 요소 찾기

search.naver.com/search.naver?where=nexearch&sm=top\_hy&fbm=1&ie=utf8&query=충북+오창+날씨

The screenshot shows the Naver weather page for Ochang, Chungbuk. The page displays the current temperature as 14°C and a forecast for the next few days. The browser's developer tools are open, showing the HTML structure and the CSS styles for the selected element.

**div.main\_info** 399 x 193

- Color #000000
- Font 13px -apple-system, BlinkMacSystemFont, ...
- Padding 42px 0px 25px 60px
- ACCESSIBILITY
- Name
- Role generic
- Keyboard-focusable

**HTML Structure:**

```
<div class="api_title_area">...</div>
<div class="api_cs_wrap">
  <div class="blind">...</div>
  <div class="weather_box">
    <div class="weather_area_mainArea">
      <div class="sort_box_areaSelectLayer">...</div>
      <div class="main_tab">...</div>
      <div class="today_area_mainTabContent" style="display:block">
        <div class="main_info">
          <span class="ico_state ws1"></span>
          <div class="info_data">
            <p class="info_temperature">
              <span class="todaytemp">14</span>
              <span class="tempmark">...</span>
            </p>
            <ul class="info_list">...</ul>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

**Styles:**

- background-color: #000000;
- font-size: 13px;
- padding: 42px 0px 25px 60px;

**Forecast:**

시간대별 날씨

14° 13° 11° 9° 7°

# [실습] 네이버 기온 가져오기 (#2/2)

[search.naver.com/search.naver?where=nexearch&sm=top\\_h ty&fbm=1&ie=utf8&query=충북+오창+날씨](https://search.naver.com/search.naver?where=nexearch&sm=top_h ty&fbm=1&ie=utf8&query=충북+오창+날씨)

```
import requests
import re                                     # 정규식(regular expression) : 패턴을 이용한 검색
from bs4 import BeautifulSoup
from pprint import pprint

def get_today_temperature(location):
    location = '충북 오창'
    url = 'https://search.naver.com/search.naver?where=nexearch&sm=top_h ty&fbm=1&ie=utf8&query='
    query = location + ' 날씨'

    res = requests.get(url + query)
    print(res)                               # <Resopnxe [200]>
    html = res.text                           # 페이지 소스 보기

    soup = BeautifulSoup(html, 'html.parser')
    data = soup.find_all('div', {'class': 'weather_info'})
    print(len(data))                          # 5
    pprint(data)
    text_temp = data[0].find('strong').text    # '현재 온도-1.0°'
    # text_temp = soup.find_all('div', {'class': 'weather_info'})[0].find('strong').text # chain rule
    str_temp = re.findall('[+-]?[0-9]+[.0-9]*', text_temp) # 정규식
    temp = float(str_temp[0])                  # ['-1.4']
    return temp
```

```
if __name__ == '__main__':
    loc = '충북 오창'
    temp = get_today_temperature(loc)
    print('{}의 현재 온도는 {:.1f}C입니다.'.format(loc, temp))
```

<https://selenium-python.readthedocs.io/>



## ■ Selenium 소개

- 웹 브라우저 자동화를 지원하는 모듈 · 웹상의 업무를 자동화
- Selenium을 사용하려면 익스플로러, 크롬 등의 웹 브라우저를 구동하는 webdriver가 필요함
- 설치 : `pip install selenium`

## ■ Chrome driver 설치 (다음장 참고)

- <https://sites.google.com/chromium.org/driver/>
- 설치 후 경로가 지정된 Python이나 프로젝트 폴더에 저장

## ■ Chrome Browser 실행

```
from selenium import webdriver  
import time
```

```
driver = webdriver.Chrome()  
driver.get('https://www.google.com/')  
time.sleep(10)  
driver.quit()
```



## ■ 크롬 버전 확인 : 도움말 > Chrome 정보









Chrome이 최신 버전입니다.

버전 109.0.5414.120(공식 빌드) (64비트)

## ■ ChromeDriver 다운로드 (<https://sites.google.com/chromium.org/driver/>)

### Current Releases

- If you are using Chrome version 110, please download [ChromeDriver 110.0.5481.30](#)
- If you are using Chrome version 109, please download [ChromeDriver 109.0.5414.74](#)
- If you are using Chrome version 108, please download [ChromeDriver 108.0.5359.71](#)
- For older version of Chrome, please see below for the version of ChromeDriver that supports it.

	Name	Last modified	Size	ETag
	<a href="#">Parent Directory</a>		-	
	<a href="#">chromedriver_linux64.zip</a>	2023-01-11 05:40:13	6.96MB	245747a6d5f2b7da02c465941f43e0e8
	<a href="#">chromedriver_mac64.zip</a>	2023-01-11 05:40:17	8.72MB	5b5bab24847bb5a6d714985967f2a87a
	<a href="#">chromedriver_mac_arm64.zip</a>	2023-01-11 05:40:20	7.95MB	d5c6a6edadb4c3647df677b4350c7721
	<a href="#">chromedriver_win32.zip</a>	2023-01-11 05:40:23	6.79MB	b447a4cb23af51da8272f6d867b9b1c7
	<a href="#">notes.txt</a>	2023-01-11 05:40:30	0.00MB	c0c31f9a056a92b1ad304580f25cbf81

## ■ 압축 풀면 chromedriver.exe가 나오는데, 현재 폴더나 python 폴더에 저장

- driver = webdriver.Chrome(executable\_path=r'C:\Chrome\chromeomdriver.exe')

## ■ 웹 요소 찾기(Locating web elements) : find\_element/elements (단수/복수)

- By 클래스로 특성을 지정

- ID : 가장 쉽고 안정적인(변화가 적은) 속성
- Name : ID 다음으로 안정적인 속성
- XPath (XML Path Language)
- XML 문서의 요소를 검색하는 질의어
- 웹 요소를 검색하는 강력한 방법이지만 ID, Name, Link Text를 우선적으로 적용할 것.
- 왜냐하면 웹 요소들의 구조변경에 영향을 받기 때문
- Link Text : Hyperlinks에 적용 예) 이곳으로 이동
- Tag Name : 한 웹페이지에 같은 태그명이 많기 때문에 보통 단독으로 사용하지는 않음
- Class Name, CSS Selector ...

```
find_element(By.ID, "id")
find_element(By.NAME, "name")
find_element(By.XPATH, "xpath")
find_element(By.LINK_TEXT, "link text")
find_element(By.TAG_NAME, "tag name")
find_element(By.CLASS_NAME, "class name")
find_element(By.CSS_SELECTOR, "css selector")
```

```
from selenium.webdriver.common.by import By

driver.find_element(By.XPATH, '//button[text()="Some text"]')
driver.find_elements(By.XPATH, '//button')
```

■ 마우스 제어 : `click()`, `click_and_hold()`, `double_click()`, `drag_and_drop()`

■ 키보드 제어 : `send_keys()`, `key_down()`, `key_up()`, `clear()`

- 특수키를 입력하기 위해서는 `Keys` 클래스의 import가 필요함

`Keys.Enter`, `BACKSPACE`, `SHIFT`, `CONTROL`, `ALT`, `F1~F12`, `NUMPAD0~9`, `ARROW_UP` 등

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

driver = webdriver.Chrome()
driver.get('https://google.com')

copy_css_selector = 'div > div.a4blc > input'
driver.find_element(By.CSS_SELECTOR, copy_css_selector).send_keys('파이썬')
driver.find_element(By.CSS_SELECTOR, copy_css_selector).send_keys(Keys.ENTER)

# driver.find_element_by_class_name("LC20lb").click() # 첫번째 검색 결과
# driver.find_element_by_css_selector('.LC20lb').text
driver.find_elements_by_class_name('LC20lb')[2].click() # 3번째 검색 결과
```

## ■ Wait 방법

- 웹페이지가 완전히 로딩되지 않아서 발생하는 문제 → `ElementNotVisibleException` 에러 등
- **Implicit Waits(암묵적 대기)** : 설정시간 동안 기다림 (default값은 0초) · 설정하면 모든 코드에 적용  
대기시간 전에 완료되면 더 기다리지 않고 다음 라인 실행
- **Explicit Waits(명시적 대기)** : 명시적으로 어떤 조건이 될 때까지 기다림  
찾을 태그를 튜플로 지정 : `(By.ID, 'com_code')` – NAME, TAG\_NAME, CLASS\_NAME 등도 가능  
timeout으로 조건이 성립하지 않을 때 기다릴 시간 설정

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
```

```
driver = webdriver.Chrome('chromedriver')
driver.implicitly_wait(time_to_wait=10) # seconds
driver.get(url='https://www.google.com/')
try:
    element = WebDriverWait(driver, 30).until(
        EC.presence_of_element_located((By.CLASS_NAME, 'gLfYf'))
    ) # 최대 30초 동안 0.5초마다, 요소가 존재하는지 확인, 인자는 튜플로 전달
finally:
    driver.quit()
```

```
EC.title_is(...)
EC.title_contains(...)
EC.presence_of_element_located(...)
EC.visibility_of_element_located(...)
EC.visibility_of(...)
EC.presence_of_all_elements_located(...)
EC.text_to_be_present_in_element(...)
EC.text_to_be_present_in_element_value(...)
EC.frame_to_be_available_and_switch_to_it(...)
EC.invisibility_of_element_located(...)
EC.element_to_be_clickable(...)
EC.staleness_of(...)
EC.element_to_be_selected(...)
EC.element_located_to_be_selected(...)
EC.element_selection_state_to_be(...)
EC.element_located_selection_state_to_be(...)
EC.alert_is_present(...)
```

## ■ 개발자 도구로 태그 찾기

The image shows a Google search page with the Chrome DevTools developer tools open. The search input field is highlighted with a red circle, and its HTML structure is shown in the Elements panel. The input field is an `input` element with the class `gLFyf`. The HTML structure is as follows:

```
<div class="SDkEP">
  <div class="iblp" jsname="uFMOof">...</div>
  <div jscontroller="vZr2rb" class="a4bIc" jsname="gLFyf"
    jsaction="h5M12e;input:d3sQLd;blur:jI3wzf">
    <div class="YacQv" jsname="vdLsw"></div>
    <input class="gLFyf" jsaction="paste:puy29d;" maxlength="2048"
      name="q" type="text" aria-autocomplete="both" aria-haspopup="false"
      autocapitalize="off" autocomplete="off" autocorrect="off"
      autofocus role="combobox" spellcheck="false" title="검색"
      value="" aria-label="검색" data-ved="0ahUKEWjox7iUspn9AhVLON4KHwFNAukQ39UDCAy"></div>
  </div>
</div>
```

The Styles panel shows the default styles for the `.gLFyf` class:

```
element.style {
}

.gLFyf {
  background-color: transparent;
  border: none;
  margin: 0;
  padding: 0;
  color: rgba(0,0,0,.87);
  word-wrap: break-word;
  outline: none;
  display: flex;
  -webkit-tap-highlight-color: transparent;
  margin-top: -37px;
}

.gLFyf, .YacQv {
  height: 34px;
  font-size: 16px;
  flex: 1 00%;
}
```

```
import time
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait # Implicitly wait
from selenium.webdriver.support import expected_conditions as EC # Explicitly wait
from selenium.webdriver.chrome.options import Options

# 웹 브라우저 옵션 설정
opts = Options()
# opts.add_argument('--headless') # 브라우저 숨기기
opts.add_argument( ' --start-maximized ' ) # 창크기 최대화
# opts.add_argument( ' --window-size=1920,1080' ) # 창크기 지정

driver = webdriver.Chrome(options=opts)
driver.implicitly_wait(time_to_wait=5) # 암묵적 대기(초)
# driver.maximize_window() # driver.set_window_size(1300, 900)

# 태그는 상황에 따라 달라지므로 직접 프로그램을 실행시킨 후 F12로 확인할 것
try: # name='q' 태그를 찾을 때까지 최대 10초를 기다림
    driver.get('https://google.com')
    WebDriverWait(driver, 30).until(EC.presence_of_element_located((By.NAME, 'q')))
```

*# iframe 스위칭 & 건너뛰기 클릭 (삽입)*

```
elem = driver.find_element(By.NAME, "q")
# elem = driver.find_element(By.CLASS_NAME, "gLfyf")
elem.clear()           # 안정성을 위해 입력필드를 지움
elem.send_keys('파이썬');
time.sleep(1)
elem.send_keys(Keys.ENTER)    # elem.submit()
```

```
# driver.find_element_by_class_name("LC20lb").click()    # 첫번째 검색 결과 (구버전)
# driver.find_element_by_css_selector('.LC20lb').text    # 태그 검색 확인 (구버전)
elems = driver.find_elements(By.CLASS_NAME, 'LC20lb')   # 모든 결과
```

```
print(elems[1].text, end="")    # 두번째 검색 제목 출력
elems[1].click()
time.sleep(5)
print(driver.current_url)
driver.back()    # driver.forward()
```

```
except Exception as e:
    print(e)
```

```
finally:
    time.sleep(5)
    driver.quit()    # driver.close()
```

*# iframe 스위칭 & 건너뛰기 클릭*

```
iframes = driver.find_elements(By.TAG_NAME, 'iframe')
for iframe in iframes:
    print(iframe.get_attribute('name'))
```

```
driver.switch_to.frame('callout') # driver.switch_to.frame(0)
driver.find_element(By.CLASS_NAME, "M6CB1c.rr4y5c").click()
driver.switch_to.default_content()
```

## ■ PyAutoGui

<https://pyautogui.readthedocs.io/en/latest/>



- 프로그래밍 방식으로 마우스 및 키보드를 제어하는 모듈
- 화면의 해상도, 위치가 바뀌면 제어가 어려움
- 설치 : `pip install pyautogui`

## ■ 마우스 제어 함수(1)

*# 화면 전체 크기 확인하기*

```
screen_size = pyautogui.size()
```

```
type(screen_size)
```

```
screen_size[0]
```

```
screen_size.width
```

```
screen_size[1]
```

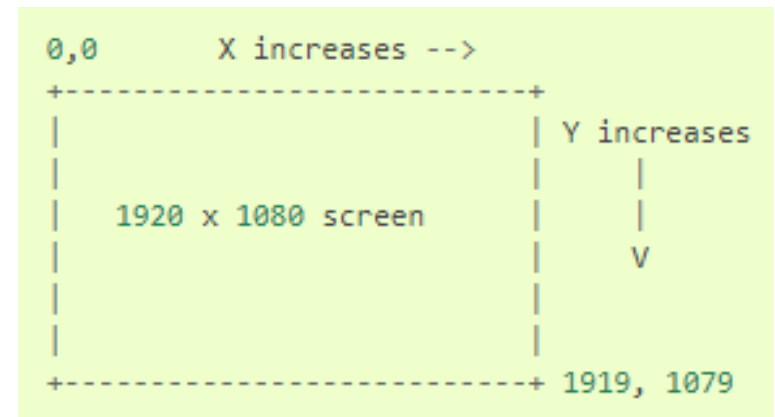
```
screen_size.height
```

```
screen_width, screen_height = pyautogui.size()
```

*# 마우스 커서 좌표 객체 얻기*

```
mouse_x, mouse_y = pyautogui.position()
```

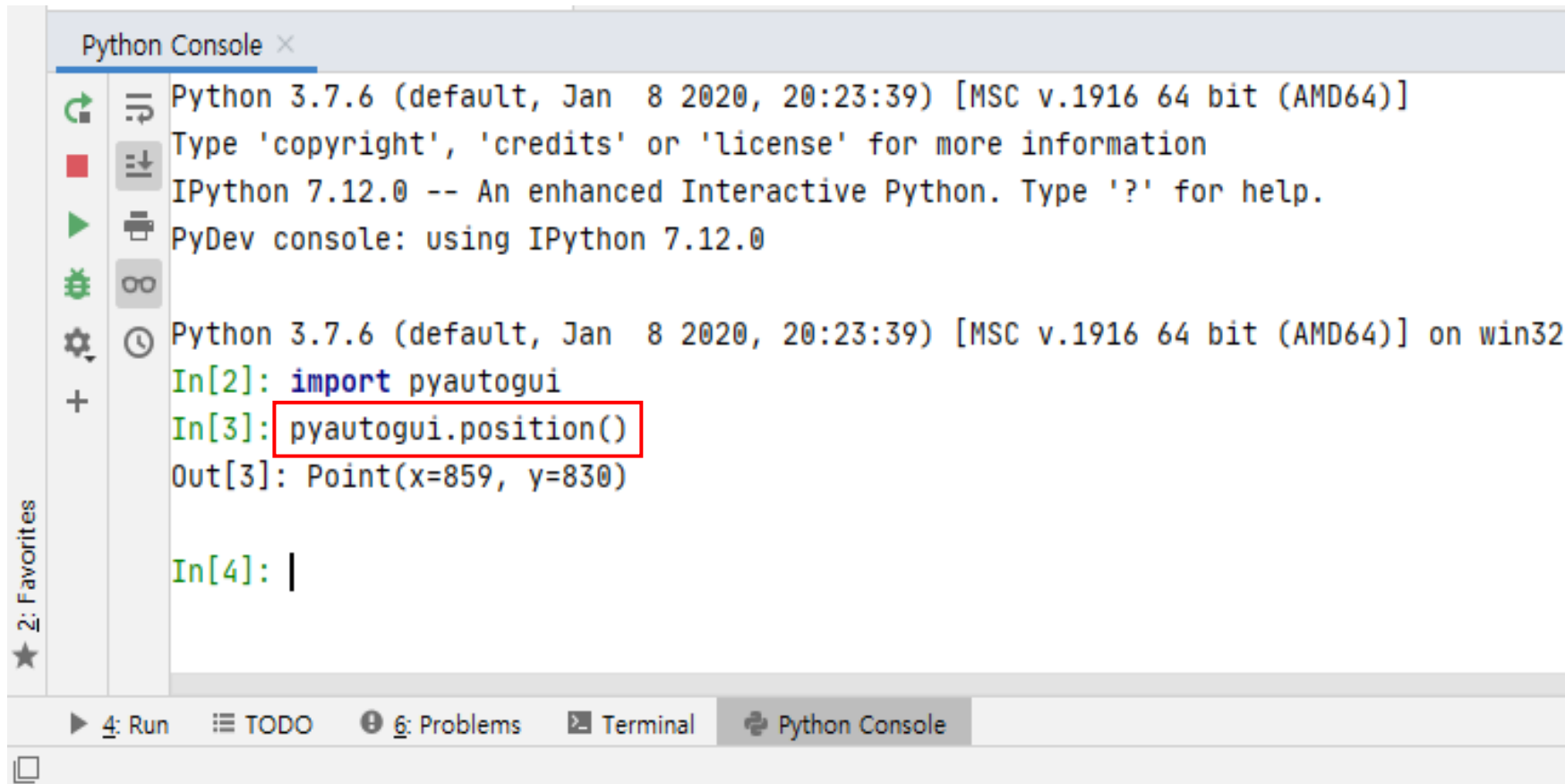
```
pyautogui.position()
```





## ■ 마우스 커서 좌표값 찾기

- position 메서드 사용



```
Python Console x
Python 3.7.6 (default, Jan 8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.12.0 -- An enhanced Interactive Python. Type '?' for help.
PyDev console: using IPython 7.12.0

Python 3.7.6 (default, Jan 8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)] on win32
In[2]: import pyautogui
In[3]: pyautogui.position()
Out[3]: Point(x=859, y=830)

In[4]: |
```

## ■ 마우스 제어 함수

*# 마우스 이동 (x 좌표, y 좌표, 이동시간)*

```
pyautogui.moveTo(100, 150)  
pyautogui.moveTo(100, 150, 10)  
pyautogui.moveRel(100, 150)
```

*# 마우스 클릭*

```
pyautogui.click()  
pyautogui.click(x=1417, y=15)  
pyautogui.click(x=1417, y=15, clicks=2, interval=2) # 2번, 2초 간격  
pyautogui.doubleClick()  
pyautogui.click(button='right') # 우클릭
```

*# 드래그하기*

```
pyautogui.moveTo(x=1736, y=427)  
pyautogui.dragRel(100, 40, 3, button='left') # xoffset, yoffset, duration  
pyautogui.dragTo(1900, 550, 10, button='left')
```

*# 스크롤하기*

```
pyautogui.moveTo(x=100, y=300)  
pyautogui.scroll(100) # scroll up 100 "clicks"  
pyautogui.scroll(-50, x=100, y=250) # (100, 200)으로 이동후 scroll down 50 "clicks"  
pyautogui.hscroll(100) # scroll right 100 "clicks"
```

주의!

## ■ 키보드 제어 함수

- pyautogui는 한글을 지원하지 않기 때문에 한글을 입력하려면, pyperclip 모듈을 통해서 한글을 복사한 후 입력해야 함.

```
# 문자 타이핑! (한글 미지원)
pyautogui.write('hello world!')
pyautogui.write('hello world!', interval=0.25) # 각 문자를 0.25간격으로 타이핑

# 한글 타이핑
pyperclip.copy("안녕하세요") # 클립보드에 텍스트를 복사
pyautogui.hotkey('ctrl', 'v') # 붙여넣기

# 문자가 아닌 키보드 자판 <Shift>, <Ctrl> 키 입력
pyautogui.press('shift') # <shift> 키
pyautogui.press('esc')
pyautogui.press(['left', 'left', 'left']) # 왼쪽 방향키 세번 입력
pyautogui.press('enter', presses=3, interval=3) # <enter> 키, 3회, 3초간격

# Press() = keyDown() + keyUp()
pyautogui.keyDown('ctrl') # <ctrl> 키 누른 상태
pyautogui.press('c') # <c> key 입력
pyautogui.keyUp('ctrl') # <ctrl> 키를 뗌

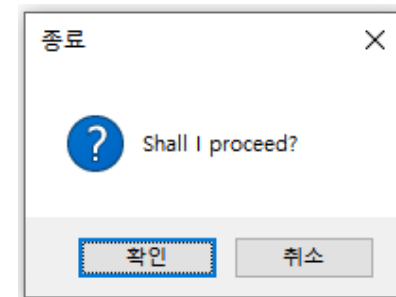
# 여러 키를 조합하여 입력할 경우
pyautogui.hotkey('ctrl', 'c') # <ctrl> + <c>
```

## ■ 다이얼로그(대화) 상자

**alert(text="", title="", button='OK')**

`pyautogui.alert('This is an alert box.')`

→ 확인을 누르면 'OK' 반환



**confirm(text="", title="", buttons=['OK', 'Cancel'])**

`pyautogui.confirm('Shall I proceed?','종료')`

→ 'Cancel'

`pyautogui.confirm('Enter option.', buttons=['A', 'B', 'C'])`

→ 'B'



**prompt(text="", title="" , default="")**

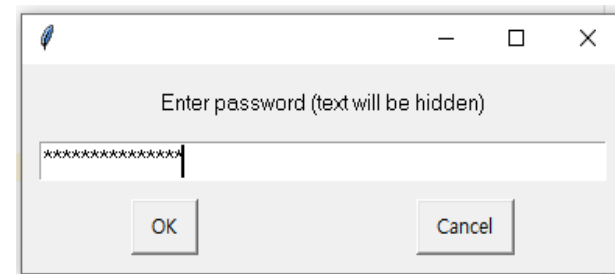
`pyautogui.prompt('What is your name?')`

→ 'AI'

**password(text="", title="", default="", mask='\*')**

`pyautogui.password('Enter password (text hidden)')`

→ 'abcd1234'



## ■ 구글 사이트에서 '파이썬' 검색하기

```
import time
import subprocess
import pyautogui as pg
import pyperclip
```

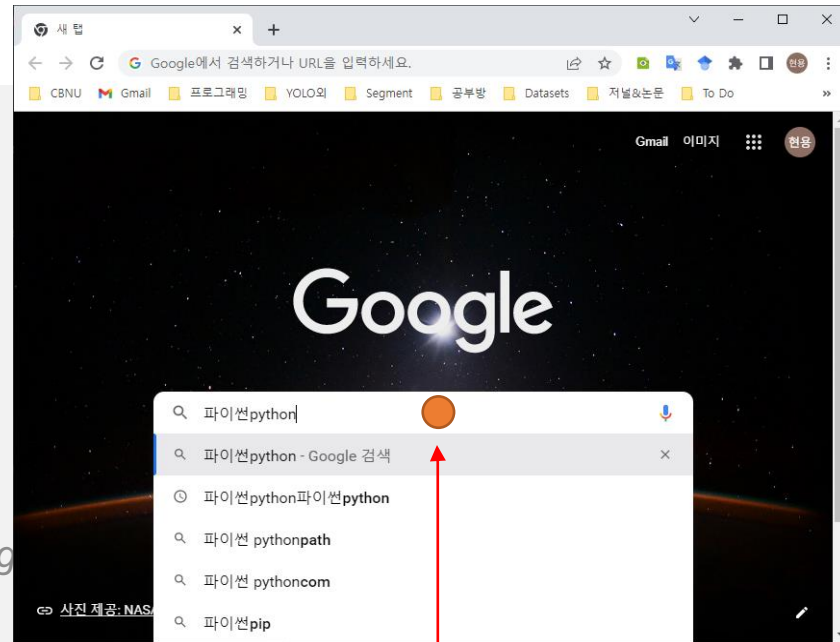
# 한영 입력 함수

```
def clip_write(txt):
    pyperclip.copy(txt)
    pyautogui.hotkey('ctrl', 'v') # pyautog
```

# Chrome을 실행한 후 'Google' 페이지로 이동

```
app_path = r"C:\Program Files\Google\Chrome\Application\chrome.exe";
subprocess.Popen(app_path)
time.sleep(3)
```

```
pg.click(x=866, y=473)
clip_write('파이썬python')
time.sleep(3)
pg.press('enter')
```





# Q & A

---

감사합니다

