Python data handling

TEAMLAB director
최성철

boostcamp AI Tech

우리가 처리하는 데이터 저장 방식들

CSV, 웹(html)
XML, JSON

Comma separate Values

Comma Separate Value

- CSV, 필드를 쉼표(,)로 구분한 텍스트 파일
- 엑셀 양식의 데이터를 프로그램에 상관없이 쓰기 위한 데이터 형식이라고 생각하면 쉬움
- 탭(TSV), 빈칸(SSV) 등으로 구분해서 만들기도 함
- 통칭하여 character-separated values (CSV) 부름
- 엑셀에서는 “다름 이름 저장” 기능으로 사용 가능

엑셀로 CSV 파일 만들기

Comma Separate Values

1) 파일 다운로드 – https://bit.ly/2KGjLxR

2) 파일 열기

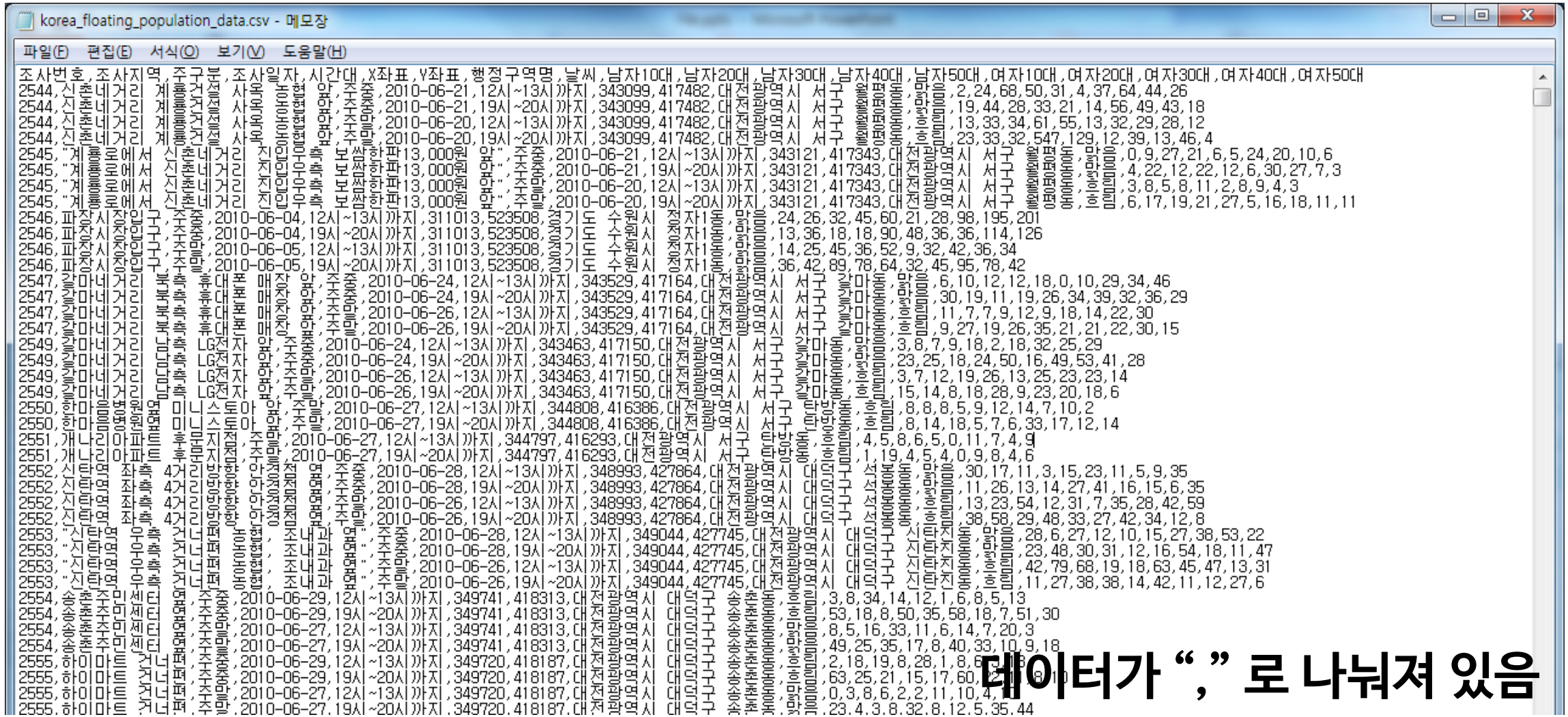
3) 파일 → 다른 이름으로 저장

→ CSV(쉼표로 분리) 선택 후 → 파일명 입력

4) 엑셀 종료 후 Notepad로 파일 열어보기

엑셀로 CSV 파일 만들기

Comma Separate Values



데이터가 “,” 로 나눠져 있음

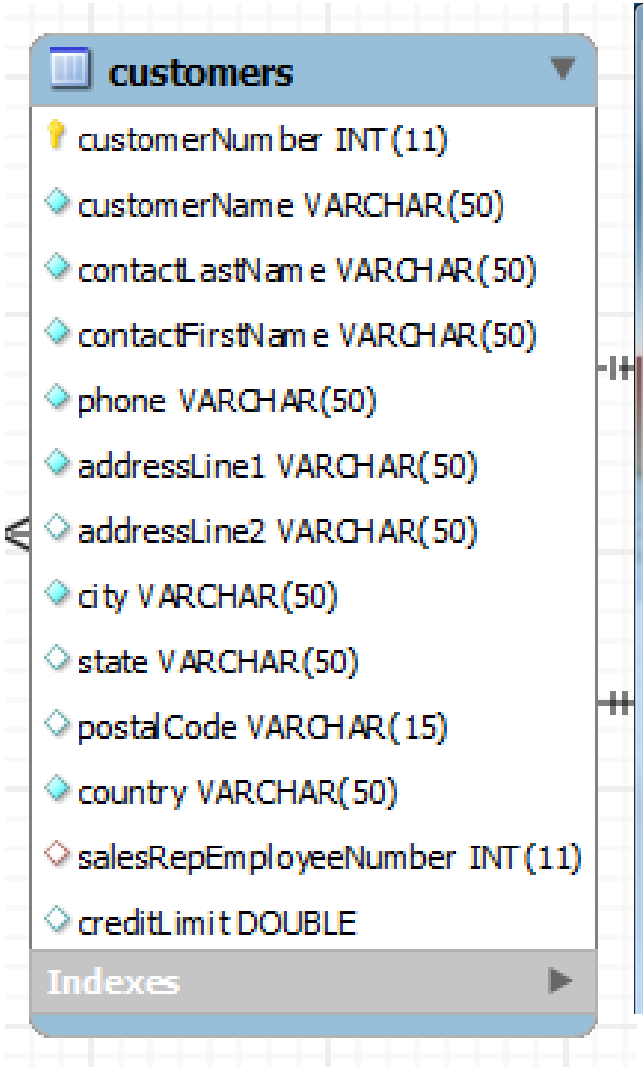
파이썬으로 CSV 파일 읽기/쓰기

Comma Separate Values

- Text 파일 형태로 데이터 처리 예제
- 예제 데이터: customer.csv (https://bit.ly/3psoUZb)
- 일반적 textfile을 처리하듯 파일을 읽어온 후, 한 줄 한 줄씩 데이터를 처리함

파이썬으로 CSV 파일 읽기/쓰기

Comma Separate Values



customers.csv - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
customerNumber,customerName,contactLastName,contactFirstName,phone,addressLine1,addressLine2,city,state,postalCode,country,salesRepEmployeeNumber,creditLimit
103,"Atelier graphique",Schmitt,"Carine ",40.32.2555,"54, rue Royale",NULL,Nantes,NULL,44000,France,1370,21000
112,"Signal Gift Stores",King,Jean,7025551838,"8489 Strong St.",NULL,"Las Vegas",NV,83030,USA,1166,71800
114,"Australian Collectors, Co.",Ferguson,Peter,"03 9520 4555","636 St Kilda Road","Level 3",Melbourne,Victoria,3004,Australia,1611,117300
119,"La Rochelle Gifts",Labrune,"Janine ",40.67.8555,"67, rue des Cinquante Otages",NULL,Nantes,NULL,44000,France,1370,118200
121,"Baane Mini Imports",Bergulfsen,"Jonas ","07-98 9555","Erling Skakkes gate 78",NULL,Stavern,NULL,4110,Norway,1504,81700
124,"Mini Gifts Distributors Ltd.",Nelson,Susan,4155551450,"5677 Strong St.",NULL,"San Rafael",CA,97562,USA,1165,210500
125,"Havel & Zbyszek Co",Piestrzeniewicz,"Zbyszek ","(26) 642-7555","ul. Filtrowa 68",NULL,Warszawa,NULL,01-012,Poland,NULL,0
128,"Blauer See Auto, Co.",Keitel,Roland,"+49 69 66 90 2555","Lyonerstr. 34",NULL,Frankfurt,NULL,60528,Germany,1504,59700
129,"Mini Wheels Co.",Murphy,Julie,6505555787,"5557 North Pendale Street",NULL,"San Francisco",CA,94217,USA,1165,64600
131,"Land of Toys Inc.",Lee,Kwai,2125557818,"897 Long Airport Avenue",NULL,NYC,NY,10022,USA,1323,114900
141,"Euro+ Shopping Channel",Freyre,"Diego ","(91) 555 94 44","C/ Moralzarzal, 86",NULL,Madrid,NULL,28034,Spain,1370,227600
144,"Volvo Model Replicas, Co",Berglund,"Christina ","0921-12 3555","Berguvsvägen  8",NULL,Luleå,NULL,"S-958 22",Sweden,1504,53100
145,"Danish Wholesale Imports",Petersen,"Jytte ","31 12 3555","Vinbæltet 34",NULL,Kobenhavn,NULL,1734,Denmark,1401,83400
146,"Saveley & Henriot, Co.",Saveley,"Mary ",78.32.5555,"2, rue du Commerce",NULL,Lyon,NULL,69004,France,1337,123900
148,"Dragon Souveniers, Ltd.",Natividad,Eric,"+65 221 7555","Bronz Sok.","Bronz Apt. 3/6 Tesvikiye",Singapore,NULL,079903,Singapore,1621,103800
151,"Muscle Machine Inc",Young,Jeff,2125557413,"4092 Furth Circle","Suite 400",NYC,NY,10022,USA,1286,138500
157,"Diecast Classics Inc.",Leong,Kelvin,2155551555,"7586 Pompton St.",NULL,Allentown,PA,70267,USA,1216,100600
161,"Technics Stores Inc.",Hashimoto,Juri,6505556809,"9408 Furth Circle",NULL,Burlingame,CA,94217,USA,1165,84600
166,"Handji Gifts& Co",Victorino,Wendy,"+65 224 1555","106 Linden Road Sandown","2nd Floor",Singapore,NULL,069045,Singapore,1612,97900
167,"Herkku Gifts",Oeztan,Veysel,"+47 2267 3215","Brehmen St. 121","PR 334 Sentrum",Bergen,NULL,"N 5804","Norway  ",1504,96800
168,"American Souvenirs Inc",Franco,Keith,2035557845,"149 Spinnaker Dr.","Suite 101","New Haven",CT,97823,USA,1286,0
169,"Porto Imports Co.","de Castro","Isabel ","(1) 356-5555","Estrada da saúde n. 58",NULL,Lisboa,NULL,1756,Portugal,NULL,0
171,"Daedalus Designs Imports",Rancé,"Martine ",20.16.1555,"184, chaussée de Tournai",NULL,Lille,NULL,59000,France,1370,82900
172,"La Corne D'abondance, Co.",Bertrand,Marie,"(1) 42.34.2555","265, boulevard Charonne",NULL,Paris,NULL,75012,France,1337,84300
173,"Cambridge Collectables Co.",Tseng,Jerry,6175555555,"4658 Baden Av.",NULL,Cambridge,MA,51247,USA,1188,43400
175,"Gift Depot Inc.",King,Julie,2035552570,"25593 South Bay Ln.",NULL,Bridgewater,CT,97562,USA,1323,84300
177,"Osaka Souveniers Co.",Kentary,Mory,"+81 06 6342 5555","1-6-20 Dojima",NULL,Kita-ku,Osaka,"  530-0003",Japan,1621,81200
181,"Vitachrome Inc.",Frick,Michael,2125551500,"2678 Kingston Rd.","Suite 101",NYC,NY,10022,USA,1286,76400
186,"Toys of Finland, Co.",Karttunen,Matti,"90-224 8555","Keskuskatu 45",NULL,Helsinki,NULL,21240,Finland,1501,96500
187,"AV Stores, Co.",Ashworth,Rachel,"(171) 555-1555","Fauntleroy Circus",NULL,Manchester,NULL,"EC2 5NT",UK,1501,136800
189,"Clover Collections, Co.",Cassidy,Dean,"+353 1862 1555","25 Maiden Lane","Floor No. 4",Dublin,NULL,2,Ireland,1504,69400
198,"Auto-Moto Classics Inc.",Taylor,Leslie,6175558428,"16780 Pompton St.",NULL,Brickhaven,MA,58339,USA,1216,23000
201,"UK Collectables, Ltd.",Devon,Elizabeth,"(171) 555-2282","12, Berkeley Gardens Blvd",NULL,Liverpool,NULL,"WX1 6LT",UK,1501,92700
202,"Canadian Gift Exchange Network",Tamuri,"Yoshi ","(604) 555-3392","1900 Oak St.",NULL,Vancouver,BC,"V3F 2K1",Canada,1323,90300
204,"Online Mini Collectables",Barajas,Miguel,6175557555,"7635 Spinnaker Dr.",NULL,Brickhaven,MA,58339,USA,1188,68700
205,Toys4GrownUps.com,Young,Julie,6265557265,"78934 Hillside Dr.",NULL,Pasadena,CA,90003,USA,1166,90700
206,"Asian Shopping Network, Co",Walker,Brydey,"+612 9411 1555","Suntec Tower Three","8 Temasek",Singapore,NULL,038988,Singapore,NULL,0
209,"Mini Caravy",Citeaux,"Frédérique ",88.60.1555,"24, place Kléber",NULL,Strasbourg,NULL,67000,France,1370,53800
211,"King Kong Collectables, Co.",Gao,Mike,"+852 2251 1555","Bank of China Tower","1 Garden Road","Central Hong Kong",NULL,NULL,"Hong Kong",1621,58600
216,"Enaco Distributors",Saavedra,"Eduardo ","(93) 203 4555","Rambla de Cataluña, 23",NULL,Barcelona,NULL,08022,Spain,1702,60300
219,"Boards & Toys Co.",Young,Mary,3105552373,"4097 Douglas Av.",NULL,Glendale,CA,92561,USA,1166,11000
223,"Natürlich Autos",Kloss,"Horst ",0372-555188,"Taucherstraße 10",NULL,Cunewalde,NULL,01307,Germany,NULL,0
227,"Heintze Collectables",Ibsen,Palle,"86 21 3555","Smagsloget 45",NULL,Århus,NULL,8200,Denmark,1401,120800
233,"Québec Home Shopping Network",Fresnière,"Jean ","(514) 555-8054","43 rue St. Laurent",NULL,Montréal,Québec,"H1J 1C3",Canada,1286,48700
237,"ANG Resellers",Camino,"Alejandra ","(91) 745 6555","Gran Vía, 1",NULL,Madrid,NULL,28001,Spain,NULL,0
239,"Collectable Mini Designs Co.",Thompson,Valarie,7605558146,"361 Furth Circle",NULL,"San Diego",CA,91217,USA,1166,105000
240,giftsbymail.co.uk,Bennett,"Helen ","(198) 555-8888","Garden House","Crowther Way 23",Cowes,"Isle of Wight","PO31 7PJ",UK,1501,93900
242,"Alpha Cognac",Roulet,"Annette ",61.77.6555,"1 rue Alsace-Lorraine",NULL,Toulouse,NULL,31000,France,1370,61100
247,"Messner Shopping Network",Messner,"Renate ",069-0555984,"Magazinweg 7",NULL,Frankfurt,NULL,60528,Germany,NULL,0
249,"Amica Models & Co.",Accorti,"Paolo ",011-4988555,"Via Monte Bianco 34",NULL,Torino,NULL,10100,Italy,1401,113000
250,"Lyon Souveniers","Da Silva",Daniel,"+33 1 46 62 7555","27 rue du Colonel Pierre Avia",NULL,Paris,NULL,75508,France,1337,68100
256,"Auto Associés & Cie.",Tonini,"Daniel ",30.59.8555,"67, avenue de l'Europe",NULL,Versailles,NULL,78000,France,1370,77900
259,"Toms Spezialitäten, Ltd",Pfalzheim,"Henriette ",0221-5554327,"Mehrheimerstr. 369",NULL,Köln,NULL,50739,Germany,1504,120400
260,"Royal Canadian Collectables, Ltd.",Lincoln,"Elizabeth ","(604) 555-4555","23 Tsawassen Blvd.",NULL,Tsawassen,BC,"T2F 8M4",Canada,1323,89600
273,"Franken Gifts, Co",Franken,"Peter ",089-0877555,"Berliner Platz 43",NULL,München,NULL,80805,Germany,NULL,0
276,"Anna's Decorations, Ltd","O'Hara,Anna","02 9936 8555","201 Miller Street","Level 15","North Sydney",NSW,2060,Australia,1611,107800
```

CSV 파일 읽기 예제

Comma Separate Values

```
line_counter = 0    #파일의 총 줄수를 세는 변수
data_header = []   #data의 필드값을 저장하는 list
customer_list = [] #cutomer 개별 List를 저장하는 List

with open ("customers.csv") as customer_data: #customer.csv 파일을 customer_data 객체에 저장

    while True:
        data = customer_data.readline() #customer.csv에 한줄씩 data 변수에 저장
        if not data: break  #데이터가 없을 때, Loop 종료
        if line_counter==0:     #첫번째 데이터는 데이터의 필드
            data_header = data.split(",")  #데이터의 필드는 data_header List에 저장, 데이터 저장시 “,”로 분리
        else:
            customer_list.append(data.split(",")) #일반 데이터는 customer_list 객체에 저장, 데이터 저장시 “,”로 분리
        line_counter += 1

print("Header :\t", data_header)   #데이터 필드 값 출력
for i in range(0,10):      #데이터 출력 (샘플 10개만)
    print ("Data",i,":\t\t",customer_list[i])
print (len(customer_list))     #전체 데이터 크기 출력
```

×

CSV 파일 쓰기 예제

Comma Separate Values

```
line_counter = 0
data_header = []
employee = []
customer_USA_only_list = []
customer = None

with open ("customers.csv", "r") as customer_data:
    while 1:
        data = customer_data.readline()
        if not data:
            break
        if line_counter==0:
            data_header = data.split(",")
        else:
            customer = data.split(",")
            if customer[10].upper() == "USA":  #customer 데이터의 offset 10번째 값
                customer_USA_only_list.append(customer)     #즉 country 필드가 "USA" 것만
        line_counter+=1           #sutomer_USA_only_list에 저장

print ("Header :\t", data_header)
for i in range(0,10):
    print ("Data :\t\t",customer_USA_only_list[i])
print (len(customer_USA_only_list))

with open ("customers_USA_only.csv", "w") as customer_USA_only_csv:
    for customer in customer_USA_only_list:
        customer_USA_only_csv.write(",".join(customer).strip('\n')+"\n")
        #cutomer_USA_only_list 객체에 있는 데이터를 customers_USA_only.csv 파일에 쓰기
```

CSV 객체로 CSV 처리

Comma Separate Values

- Text 파일 형태로 데이터 처리시 문장 내에 들어가 있는 “,” 등에 대해 전처리 과정이 필요
- 파이썬에서는 간단히 CSV파일을 처리하기 위해 csv 객체를 제공함
- 예제 데이터: korea_foot_traffic_data.csv (from http://www.data.go.kr)
- 예제 데이터는 국내 주요 상권의 유동인구 현황 정보
- 한글로 되어 있어 한글 처리가 필요

CSV 객체로 CSV 처리

- 시간대/조사일자/행정구역/날씨 등을 기준으로 연령별/성별 유동인가 해당 지역에 몇 명인가 표시

```
조사번호,조사지역,주구분,조사일자,시간대,X좌표,Y좌표,행정구역명,날씨,남자10대,남자20대,남자30대,남자40대,남자50대,여자10대,여자20대,여자30대,여자40대,여자50대
2544,신촌네거리 계룡건설 사옥 농협 앞,주중,2010-06-21,12시~13시까지,343099,417482,대전광역시 서구 월평동,맑음,2,24,68,50,31,4,37,64,44,26
2544,신촌네거리 계룡건설 사옥 농협 앞,주중,2010-06-21,19시~20시까지,343099,417482,대전광역시 서구 월평동,맑음,19,44,28,33,21,14,56,49,43,18
2544,신촌네거리 계룡건설 사옥 농협 앞,주말,2010-06-20,12시~13시까지,343099,417482,대전광역시 서구 월평동,흐림,13,33,34,61,55,13,32,29,28,12
2544,신촌네거리 계룡건설 사옥 농협 앞,주말,2010-06-20,19시~20시까지,343099,417482,대전광역시 서구 월평동,흐림,23,33,32,547,129,12,39,13,46,4
2545,"계룡로에서 신촌네거리 진입우측 보쌈한판13,000원 앞",주중,2010-06-21,12시~13시까지,343121,417343,대전광역시 서구 월평동,맑음,0,9,27,21,6,5,24,20,10,6
2545,"계룡로에서 신촌네거리 진입우측 보쌈한판13,000원 앞",주중,2010-06-21,19시~20시까지,343121,417343,대전광역시 서구 월평동,맑음,4,22,12,22,12,6,30,27,7,3
2545,"계룡로에서 신촌네거리 진입우측 보쌈한판13,000원 앞",주말,2010-06-20,12시~13시까지,343121,417343,대전광역시 서구 월평동,흐림,3,8,5,8,11,2,8,9,4,3
2545,"계룡로에서 신촌네거리 진입우측 보쌈한판13,000원 앞",주말,2010-06-20,19시~20시까지,343121,417343,대전광역시 서구 월평동,흐림,6,17,19,21,27,5,16,18,11,11
2546,파장시장입구,주중,2010-06-04,12시~13시까지,311013,523508,경기도 수원시 정자1동,맑음,24,26,32,45,60,21,28,98,195,201
2546,파장시장입구,주중,2010-06-04,19시~20시까지,311013,523508,경기도 수원시 정자1동,맑음,13,36,18,18,90,48,36,36,114,126
2546,파장시장입구,주말,2010-06-05,12시~13시까지,311013,523508,경기도 수원시 정자1동,맑음,14,25,45,36,52,9,32,42,36,34
2546,파장시장입구,주말,2010-06-05,19시~20시까지,311013,523508,경기도 수원시 정자1동,맑음,36,42,89,78,64,32,45,95,78,42
2547,갈마네거리 북측 휴대폰 매장 앞,주중,2010-06-24,12시~13시까지,343529,417164,대전광역시 서구 갈마동,맑음,6,10,12,12,18,0,10,29,34,46
2547,갈마네거리 북측 휴대폰 매장 앞,주중,2010-06-24,19시~20시까지,343529,417164,대전광역시 서구 갈마동,맑음,30,19,11,19,26,34,39,32,36,29
2547,갈마네거리 북측 휴대폰 매장 앞,주말,2010-06-26,12시~13시까지,343529,417164,대전광역시 서구 갈마동,흐림,11,7,7,9,12,9,18,14,22,30
2547,갈마네거리 북측 휴대폰 매장 앞,주말,2010-06-26,19시~20시까지,343529,417164,대전광역시 서구 갈마동,흐림,9,27,19,26,35,21,21,22,30,15
2549,갈마네거리 남측 LG전자 앞,주중,2010-06-24,12시~13시까지,343463,417150,대전광역시 서구 갈마동,맑음,3,8,7,9,18,2,18,32,25,29
2549,갈마네거리 남측 LG전자 앞,주중,2010-06-24,19시~20시까지,343463,417150,대전광역시 서구 갈마동,맑음,23,25,18,24,50,16,49,53,41,28
2549,갈마네거리 남측 LG전자 앞,주말,2010-06-26,12시~13시까지,343463,417150,대전광역시 서구 갈마동,흐림,3,7,12,19,26,13,25,23,23,14
2549,갈마네거리 남측 LG전자 앞,주말,2010-06-26,19시~20시까지,343463,417150,대전광역시 서구 갈마동,흐림,15,14,8,18,28,9,23,20,18,6
2550,한마음병원옆 미니스토아 앞,주말,2010-06-27,12시~13시까지,344808,416386,대전광역시 서구 탄방동,흐림,8,8,8,5,9,12,14,7,10,2
2550,한마음병원옆 미니스토아 앞,주말,2010-06-27,19시~20시까지,344808,416386,대전광역시 서구 탄방동,흐림,8,14,18,5,7,6,33,17,12,14
2551,개나리아파트 후문지점,주말,2010-06-27,12시~13시까지,344797,416293,대전광역시 서구 탄방동,흐림,4,5,8,6,5,0,11,7,4,9
2551,개나리아파트 후문지점,주말,2010-06-27,19시~20시까지,344797,416293,대전광역시 서구 탄방동,흐림,1,19,4,5,4,0,9,8,4,6
2552,신탄역 좌측 4거리방향 안경점 옆,주중,2010-06-28,12시~13시까지,348993,427864,대전광역시 대덕구 석봉동,맑음,30,17,11,3,15,23,11,5,9,35
2552,신탄역 좌측 4거리방향 안경점 옆,주중,2010-06-28,19시~20시까지,348993,427864,대전광역시 대덕구 석봉동,맑음,11,26,13,14,27,41,16,15,6,35
2552,신탄역 좌측 4거리방향 안경점 옆,주말,2010-06-26,12시~13시까지,348993,427864,대전광역시 대덕구 석봉동,흐림,13,23,54,12,31,7,35,28,42,59
2552,신탄역 좌측 4거리방향 안경점 옆,주말,2010-06-26,19시~20시까지,348993,427864,대전광역시 대덕구 석봉동,흐림,38,58,29,48,33,27,42,34,12,8
2553,"신탄역 우측 건너편 농협, 조내과 옆",주중,2010-06-28,12시~13시까지,349044,427745,대전광역시 대덕구 신탄진동,맑음,28,6,27,12,10,15,27,38,53,22
2553,"신탄역 우측 건너편 농협, 조내과 옆",주중,2010-06-28,19시~20시까지,349044,427745,대전광역시 대덕구 신탄진동,맑음,23,48,30,31,12,16,54,18,11,47
2553,"신탄역 우측 건너편 농협, 조내과 옆",주말,2010-06-26,12시~13시까지,349044,427745,대전광역시 대덕구 신탄진동,흐림,42,79,68,19,18,63,45,47,13,31
2553,"신탄역 우측 건너편 농협, 조내과 옆",주말,2010-06-26,19시~20시까지,349044,427745,대전광역시 대덕구 신탄진동,흐림,11,27,38,38,14,42,11,12,27,6
2554,송촌주민센터 옆,주중,2010-06-29,12시~13시까지,349741,418313,대전광역시 대덕구 송촌동,흐림,3,8,34,14,12,1,6,8,5,13
2554,송촌주민센터 옆,주중,2010-06-29,19시~20시까지,349741,418313,대전광역시 대덕구 송촌동,흐림,53,18,8,50,35,58,18,7,51,30
2554,송촌주민센터 옆,주말,2010-06-27,12시~13시까지,349741,418313,대전광역시 대덕구 송촌동,맑음,8,5,16,33,11,6,14,7,20,3
2554,송촌주민센터 옆,주말,2010-06-27,19시~20시까지,349741,418313,대전광역시 대덕구 송촌동,맑음,49,25,35,17,8,40,33,10,9,18
2555,하이마트 건너편,주중,2010-06-29,12시~13시까지,349720,418187,대전광역시 대덕구 송촌동,흐림,2,18,19,8,28,1,8,6,3,13
```

CSV 객체 활용

Comma Separate Values

```
import csv
reader = csv.reader(f,
                    delimiter=',', quotechar='"',
                    quoting=csv.QUOTE_ALL)
```

Attribute	Default	Meaning
delimiter	,	글자를 나누는 기준
lineterminator	₩r₩n	줄 바꿈 기준
quotechar	"	문자열을 둘러싸는 신호 문자
quoting	QUOTE_MINIMAL	데이터 나누는 기준이 quotechar에 의해 둘러싸인 레벨

https://docs.python.org/ko/3.8/library/csv.html

CSV 객체 활용

Comma Separate Values

```
import csv
seoung_nam_data = []
header = []
rownum = 0

with open("korea_floating_population_data.csv","r", encoding="cp949") as p_file:
    csv_data = csv.reader(p_file) #csv 객체를 이용해서 csv_data 읽기
    for row in csv_data:    #읽어온 데이터를  한 줄씩 처리
        if rownum == 0:
            header = row #첫 번째 줄은 데이터 필드로 따로 저장
        location = row[7]
        # " 행정구역 " 필드 데이터 추출, 한글 처리로 유니코드 데이터를 cp949로 변환
        if location.find(u"성남시") != -1:
            seoung_nam_data.append(row)
        # " 행정구역 " 데이터에 성남시가 들어가 있으면 seoung_nam_data List에 추가
        rownum +=1

with open("seoung_nam_floating_population_data.csv","w", encoding="utf8") as s_p_file:
    writer = csv.writer(s_p_file, delimiter='\t', quotechar="'", quoting=csv.QUOTE_ALL)
    # csv.writer를 사용해서 csv 파일 만들기 delimiter 필드 구분자
    # quotechar는 필드 각 데이터는 묶는 문자, quoting는 묶는 범위
    writer.writerow(header)        #제목 필드 파일에 쓰기
    for row in seoung_nam_data:
        writer.writerow(row)    #seoung_nam_data에 있는 정보 list에 쓰기
```

유동 인구 데이터 중 성남 데이터만 수집

Web

하루 중 가장 많이 시간을 보내는 곳

인터넷 = Web

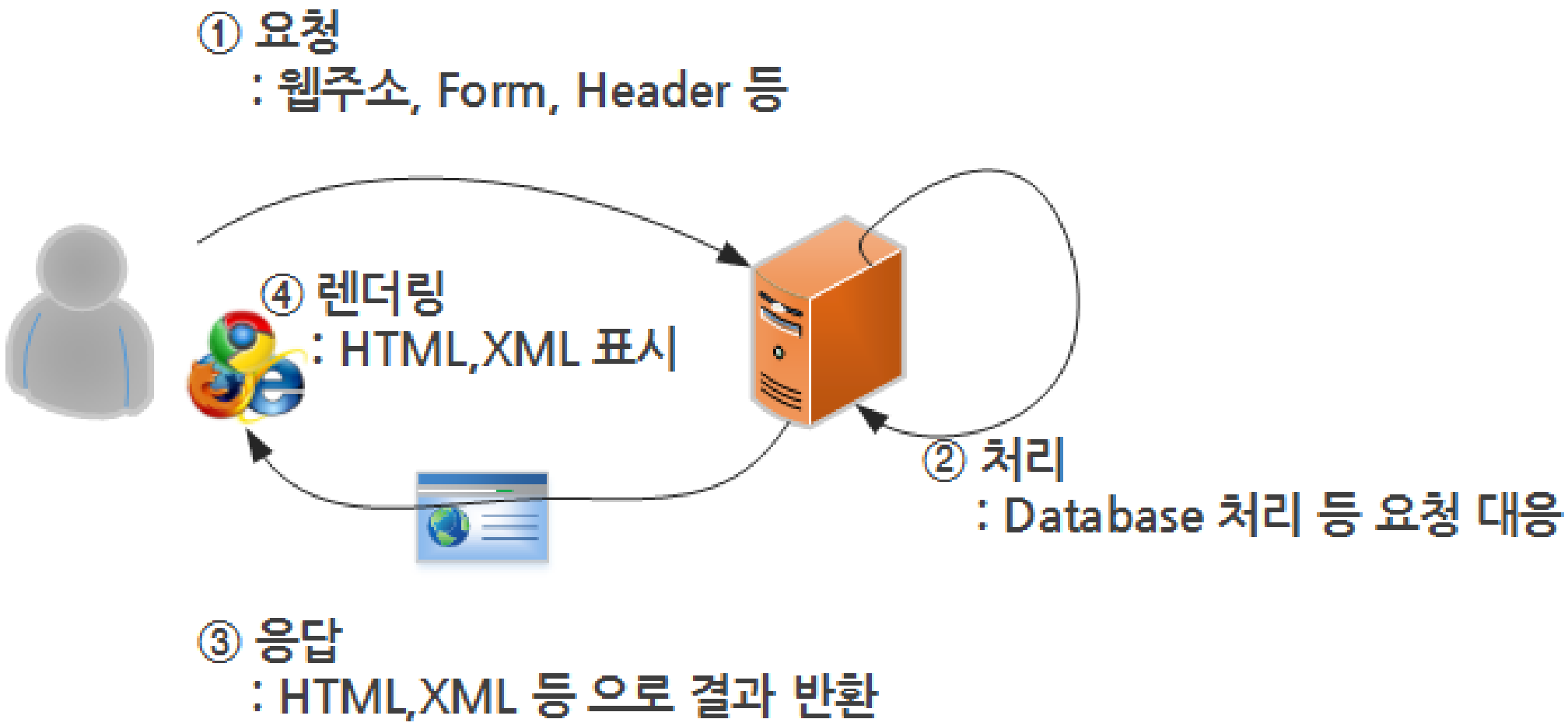
Web – 우리가 늘 쓰는 그 것

- World Wide Web(WWW), 줄여서 웹이라고 부름
- 우리가 늘 쓰는 인터넷 공간의 정식 명칭
- 팀 버너스리에 의해 1989년 처음 제안되었으며,

  원래는 물리학자들간 정보 교환을 위해 사용됨
- 데이터 송수신을 위한 HTTP 프로토콜 사용,

  데이터를 표시하기 위해 HTML 형식을 사용

Web은 어떻게 동작하는가?



HTML(Hyper Text Markup Language)

- 웹 상의 정보를 구조적으로 표현하기 위한 언어
- 제목, 단락, 링크 등 요소 표시를 위해 Tag를 사용
- 모든 요소들은 꺾쇠 괄호 안에 둘러 쌓여 있음

  〈title〉 Hello, World 〈/title〉 # 제목 요소, 값은 Hello, World
- 모든 HTML은 트리 모양의 포함관계를 가짐
- 일반적으로 웹 페이지의 HTML 소스파일은

  컴퓨터가 다운로드 받은 후 웹 브라우저가 해석/표시

HTML(Hyper Text Markup Language)


```
<!doctype html>
<html>
  <head>
    <title>Hello HTML</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

HTML 구조

<html> – <head> – <title>
– <body> – <p>

Element, Attribute Value 이루어짐

<tag attribute1= " att_value1" attribute2="
att_value1 ">
보이는 내용(Value)
</tag>

Source: http://ko.wikipedia.org/wiki/HTML

왜 웹을 알아야 하는가?

- 정보의 보고, 많은 데이터들이 웹을 통해 공유됨

  환율정보: https://finance.naver.com/

  날씨정보 : http://goo.gl/nwi8WE

  미국 특허정보: http://bit.ly/3pxFkjb

- HTML도 일종의 프로그램, 페이지 생성 규칙이 있음

  : 규칙을 분석하여 데이터의 추출이 가능

- 추출된 데이터를 바탕으로 하여 다양한 분석이 가능

정규식 (regular expression)

- 정규 표현식, regexp 또는 regex 등으로 불림
- 복잡한 문자열 패턴을 정의하는 문자 표현 공식
- 특정한 규칙을 가진 문자열의 집합을 추출

```
010-0000-0000    ^\d{3}\-\d{4}\-\d{4}$
203.252.101.40   ^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$
```

정규식 for HTML Parsing

- 주민등록 번호, 전화번호, 도서 ISBN 등 형식이 있는 문자열을 원본 문자열로부터 추출함

- HTML역시 tag를 사용한 일정한 형식이 존재하여 정규식으로 추출이 용이함

- 관련자료: http://www.nextree.co.kr/p4327/

정규식 for HTML Parsing

- 주민등록 번호, 전화번호, 도서 ISBN 등 형식이 있는 문자열을 원본 문자열로부터 추출함

- HTML역시 tag를 사용한 일정한 형식이 존재하여 정규식으로 추출이 용이함

- 관련자료: http://www.nextree.co.kr/p4327/

정규식 for HTML Parsing

- 문법 자체는 매우 방대, 스스로 찾아서 하는 공부 필요
- 필요한 것들은 인터넷 검색을 통해 찾을 수 있음
- 기본적인 것을 공부 한 후 넓게 적용하는 것이 중요

```
이메일 :        ^[a-zA-Z0-9]+@[a-zA-Z0-9]+$  or
                ^[_0-9a-zA-Z-]+@[0-9a-zA-Z-]+(.[_0-9a-zA-Z-]+)*$
휴대폰 :        ^01(?:0|1|[6-9]) - (?:₩d{3}|₩d{4}) - ₩d{4}$
일반전화 :      ^₩d{2,3} - ₩d{3,4} - ₩d{4}$
주민등록번호 :  ₩d{6} ₩- [1-4]₩d{6}
IP 주소 :       ([0-9]{1,3}) ₩. ([0-9]{1,3}) ₩. ([0-9]{1,3}) ₩. ([0-9]{1,3})
```

http://bit.ly/2WNudWK

정규식 연습장 활용하기

1) 정규식 연습장(http://www.regexr.com/) 으로 이동

2) 테스트하고 싶은 문서를 Text 란에 삽입

3) 정규식을 사용해서 찾아보기

정규식 기본 문법 #1

문자 클래스 [ ] : [와] 사이의 문자들과 매치라는 의미

예) [abc] ← 해당 글자가 a,b,c중 하나가 있다.
"a", "before", "deep" , "dud", "sunset"

"-"를 사용 범위를 지정할 수 있음

예) [a-zA-z] - 알파벳 전체, [0-9] - 숫자 전체

https://wikidocs.net/4308

정규식 기본 문법 – 메타 문자

정규식 표현을 위해 원래 의미 X, 다른 용도로 사용되는 문자

```
. ^ $ * + ? { } [ ] \ | ( )
```

- **. – 줄바꿈 문자인 ₩n를 제외한 모든 문자와 매치** a[.]b
- *** – 앞에 있는 글자를 반복해서 나올 수 있음**

  tomor*ow tomorrow tomoow tomorrrrow
- **+ – 앞에 있는 글자를 최소 1회 이상 반복**

https://wikidocs.net/4308

정규식 기본 문법 – 메타 문자

정규식 표현을 위해 원래 의미 X, 다른 용도로 사용되는 문자

```
. ^ $ * + ? { } [ ] \ | ( )
```

{m.n} – 반복 횟수를 지정 `{1,} , {0,} {1,3}`

`203.252.101.40` `[0-9]{1,3}` `\d{1,3}`

? – 반복 횟수가 1회 `01[01]?-[0-9]{4}-[0-9]{4}`

| – or `(0|1){3}` **^ – not**

https://wikidocs.net/4308

정규식 추출 연습

① 정규식 연습장(http://www.regexr.com/) 으로 이동

② 구글 USPTO Bulk Download 데이터페이지 소스 보기 클릭

③ 소스 전체 복사후 정규식 연습장 페이지에 붙여넣기

④ 상단 Expression 부분을 수정해가며 “Zip”로 끝나는 파일명만 추출

⑤ Expression에 (http) (.+) (zip) 를 입력

http://www.google.com/googlebooks/uspto-patents-grants-text.html

정규식 in 파이썬

- re 모듈을 import 하여 사용 : import re
- 함수: search – 한 개만 찾기, findall – 전체 찾기
- 추출된 패턴은 tuple로 반환됨
- 연습 – 특정 페이지에서 ID만 추출하기 https://bit.ly/3rxQFS4
- ID 패턴: [영문대소문자|숫자] 여러 개, 별표로 끝남
  "([A-Za-z0-9]+₩*₩*₩*)“ 정규식

정규식 in 파이썬


```
import re
import urllib.request

url = "https://bit.ly/3rxQFS4"
html = urllib.request.urlopen(url)
html_contents = str(html.read())
id_results = re.findall(r"([A-Za-z0-9]+\*\*\*)", html_contents)
#findall 전체 찾기, 패턴대로 데이터 찾기

for result in id_results:
    print (result)
```

정규식 in 파이썬


```
import urllib.request # urllib 모듈 호출
import re

url = "http://www.google.com/googlebooks/uspto-patents-grants-text.html"
#url 값 입력
html = urllib.request.urlopen(url)  # url 열기
html_contents = str(html.read().decode("utf8"))
# html 파일 읽고, 문자열로 변환

url_list = re.findall(r"(http)(.+)(zip)", html_contents)
for url in url_list:
    print("".join(url))  # 출력된 Tuple 형태 데이터 str으로 join
```

정규식 in 파이썬 for html


```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2020년 01월 17일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 61,300 전일대비 상승 600 플러스 0.99 퍼센트</dd>
    <dd>전일가 60,700</dd>
    <dd>시가 61,900</dd>
    <dd>고가 62,000</dd>
    <dd>상한가 78,900</dd>
    <dd>저가 61,000</dd>
    <dd>하한가 42,500</dd>
    <dd>거래량 15,451,576</dd>
    <dd>거래대금 949,344백만</dd>
</dl>
```

이 데이터는 어떻게 뽑을까?

정규식 in 파이썬 for html


```
<dl class="blind">
    <dt>종목 시세 정보</dt>
    <dd>2020년 01월 17일 16시 10분 기준 장마감</dd>
    <dd>종목명 삼성전자</dd>
    <dd>종목코드 005930 코스피</dd>
    <dd>현재가 61,300 전일대비 상승 600 플러스 0.99 퍼센트</dd>
    <dd>전일가 60,700</dd>
    <dd>시가 61,900</dd>
    <dd>고가 62,000</dd>
    <dd>상한가 78,900</dd>
    <dd>저가 61,000</dd>
    <dd>하한가 42,500</dd>
    <dd>거래량 15,451,576</dd>
    <dd>거래대금 949,344백만</dd>
</dl>
```

① <dl class="blind"> ~~~~ </dl> 에 있는
② <dd> ~~~~ </dd> 정보를 추출하면 됨

정규식 in 파이썬 for html

① ⟨dl class="blind"⟩ ~~~~ ⟨/dl⟩
(₩⟨dl class=₩"blind₩"₩⟩)([₩s₩S]+?)(₩⟨₩/dl₩⟩)
⟨dl class에서 시작해서 / 사이에 아무 글자나 있고 / ⟨/dl⟩ 로 끝내기

② ⟨dd⟩ ~~~~ ⟨/dd⟩ 정보를 추출하면 됨
(₩⟨dd₩⟩)([₩s₩S]+?)(₩⟨₩/dd₩⟩)
⟨dd⟩ 에서 시작해서 / 사이에 아무 글자나 있고 / ⟨/dl⟩ 로 끝내기

① 를 먼저 찾고 ① 안에 ②를 차례대로 찾으면 됨

정규식 in 파이썬 for html


```
import urllib.request
import re

url = "http://finance.naver.com/item/main.nhn?code=005930"
html = urllib.request.urlopen(url)
html_contents = str(html.read().decode("ms949"))

stock_results = re.findall("(\<dl class=\"blind\"\>)([\s\S]+?)(\<\/dl\>)", html_contents)
samsung_stock = stock_results[0] # 두 개 tuple 값중 첫번째 패턴
samsung_index = samsung_stock[1] # 세 개의 tuple 값중 두 번째 값
                                                    # 하나의 괄호가 tuple index가 됨
index_list= re.findall("(\<dd\>)([\s\S]+?)(\<\/dd\>)", samsung_index)

for index in index_list:
    print (index[1]) # 세 개의 tuple 값중 두 번째 값
```

eXtensible Markup Language

XML이란

- 데이터의 구조와 의미를 설명하는
  TAG(MarkUp)를 사용하여 표시하는 언어

- TAG와 TAG사이에 값이 표시되고,
  구조적인 정보를 표현할 수 있음

- HTML과 문법이 비슷, 대표적인 데이터 저장 방식

XML이란

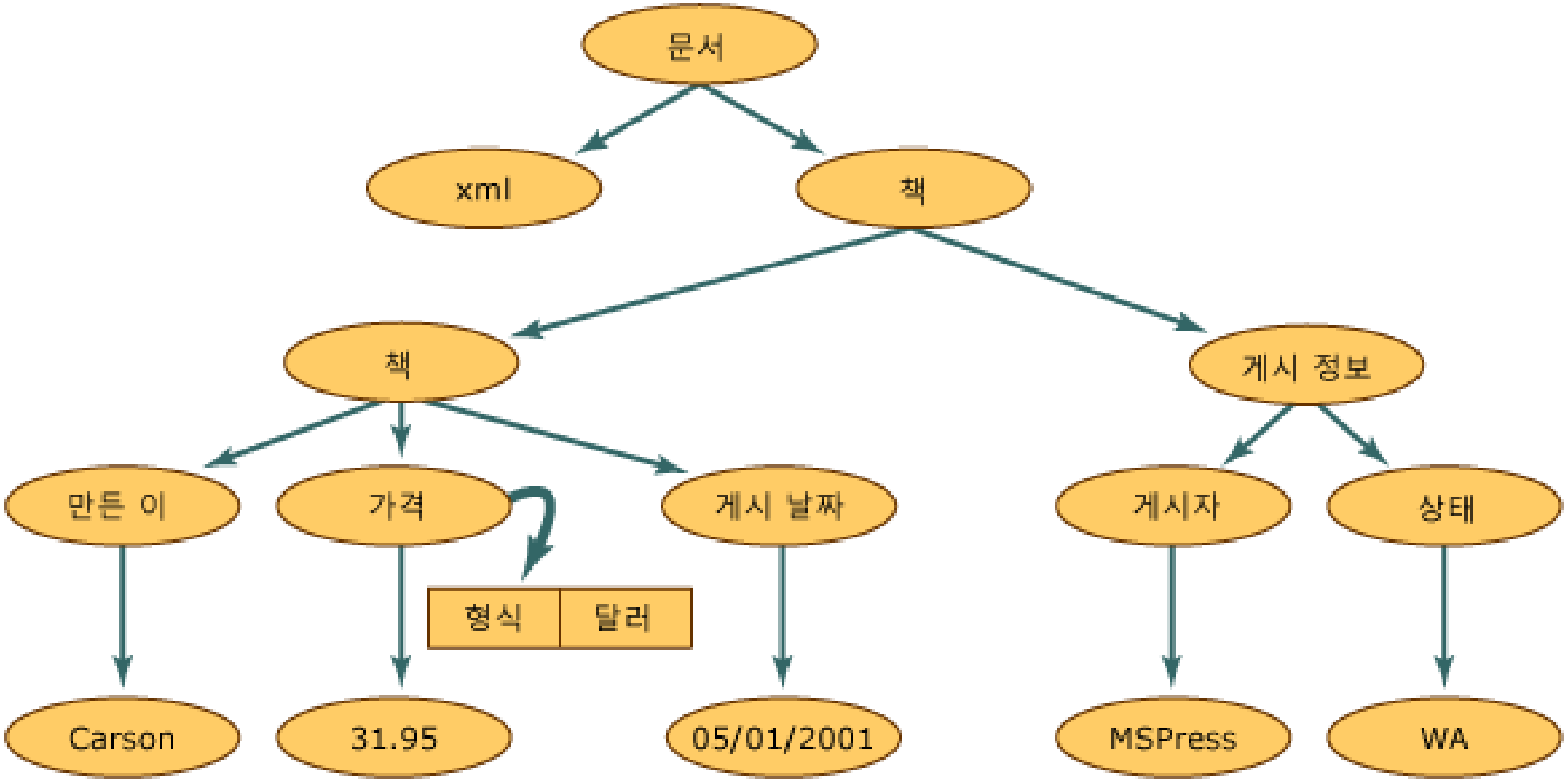
- 정보의 구조에 대한 정보인 스키마와 DTD 등으로 정보에 대한 정보(메타정보)가 표현되며, 용도에 따라 다양한 형태로 변경가능

- XML은 컴퓨터(예: PC ↔ 스마트폰)간에 정보를 주고받기 매우 유용한 저장 방식으로 쓰이고 있음

XML 예제


```
<?xml version="1.0"?>
<고양이>
  <이름>나비</이름>
  <품종>샴</품종>
  <나이>6</나이>
  <중성화>예</중성화>
  <발톱 제거>아니요</발톱 제거>
  <등록 번호>Izz138bod</등록 번호>
  <소유자>이강주</소유자>
</고양이>
```

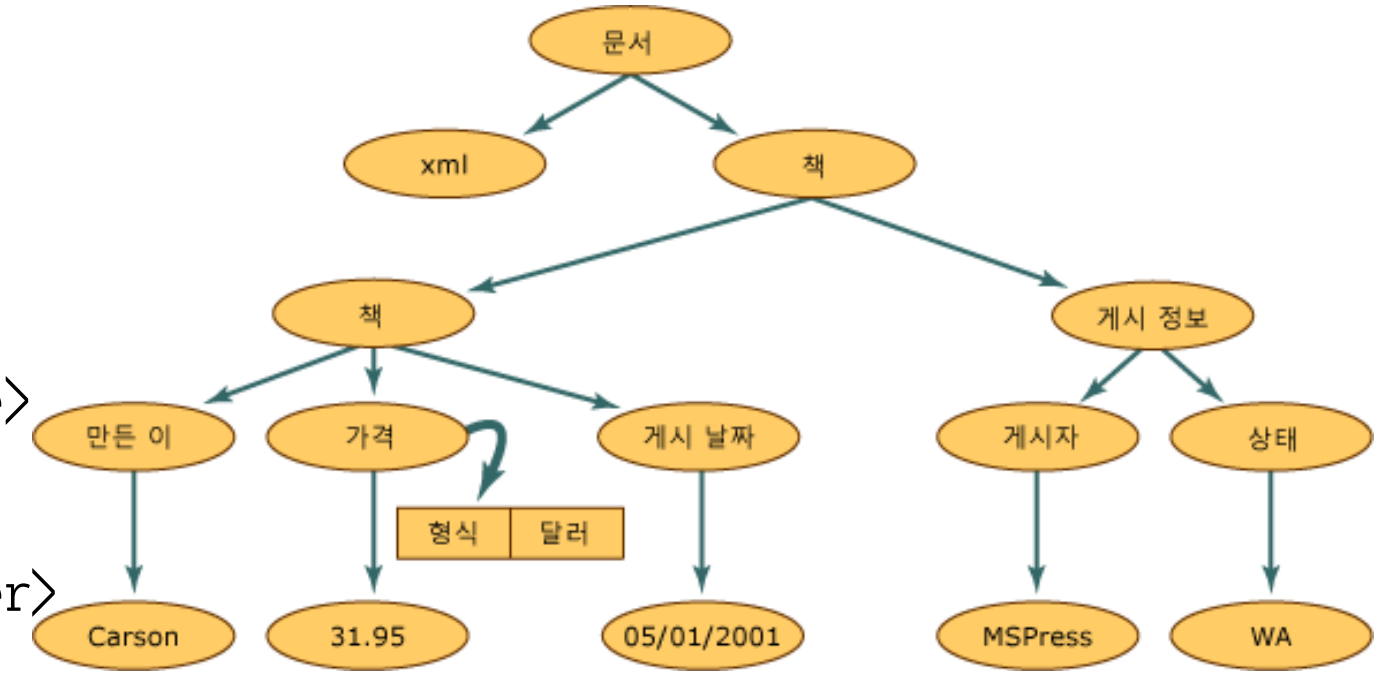
XML 예제



http://goo.gl/7m015w

XML 형태로 만들어 보기


```
<?xml version="1.0"?>
  <books>
    <book>
        <author>Carson</author>
        <price
format="dollar">31.95</price>
        <pubdate>05/01/2001</pubdate>
    </book>
    <pubinfo>
        <publisher>MSPress</publisher>
        <state>WA</state>
    </pubinfo>
  </books>
```



http://goo.gl/7m015w

XML Parsing in Python

- XML도 HTML과 같이 구조적 markup 언어
- 정규표현식으로 Parsing이 가능함
- 그러나 좀 더 손쉬운 도구들이 개발되어 있음
- 가장 많이 쓰이는 parser인 beautifulsoup으로 파싱

BeautifulSoup

- HTML, XML등 Markup 언어 Scraping을 위한 대표적인 도구
- https://www.crummy.com/software/BeautifulSoup/
- lxml 과 html5lib 과 같은 Parser를 사용함
- 속도는 상대적으로 느리나 간편히 사용할 수 있음

BeautifulSoup

	Python 2.7		Python 3.2	
Parser	Speed (KB/s)	Success rate	Speed (KB/s)	Success rate
Beautiful Soup 3.2 (SGMLParser)	211	100%	-	-
html5lib (BS3 treebuilder)	253	99%	-	-
Beautiful Soup 4.0 + lxml	255	100%	2140	96%
html5lib (lxml treebuilder)	270	99%	-	-
Beautiful Soup 4.0 + html5lib	271	98%	-	-
Beautiful Soup 4.0 + HTMLParser	299	59%	1705	57%
html5lib (simpletree treebuilder)	332	100%	-	-
HTMLParser	5194	52%	3918	57%
lxml	17925	100%	14258	96%

https://www.crummy.com/2012/01/22/0

beautifulsoup 설치

- conda 가상 환경으로 lxml과 beautifulsoup 설치

```
activate python_mooc
conda install lxml
conda install -c anaconda beautifulsoup4=4.5.1
```

beautifulsoup 모듈 사용

- 모듈 호출

```
from bs4 import BeautifulSoup
```

- 객체 생성

```
soup  = BeautifulSoup(books_xml, "lxml")
```

- Tag 찾는 함수 find_all 생성

```
soup.find_all("author")
```

beautifulsoup 모듈 사용

- find_all: 정규식과 마찬가지로 해당 패턴을 모두 반환
- **find('invention-title')**

Tag 네임 = title

- get_text(): 반환된 패턴의 값 반환 (태그와 태그 사이)

⟨invention-title id="d2e43"⟩
Adjustable shoulder device for hard upper torso suit
⟨/invention-title⟩

http://goo.gl/aeKMGS, http://goo.gl/lKhFzh 참고

beautifulsoup Example

- 데이터 다운로드 받기

https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/books.xml

```
from bs4 import BeautifulSoup

with open("books.xml", "r", encoding="utf8") as books_file:
    books_xml = books_file.read()  # File을 String으로 읽어오기

soup  = BeautifulSoup(books_xml, "lxml") # lxml Parser를 사용해서 데이터 분석

# author가 들어간 모든 element 추출
for book_info in soup.find_all("author"):
    print (book_info)
    print (book_info.get_text())
```

beautifulsoup 예제 데이터

- 미국 특허청 (USPTO) 특허 데이터는 XML로 제공됨

- 해당 데이터중 등록번호 “08621662” 인

  “Adjustable shoulder device for hard upper torso suit” 분석

  참고: http://www.google.com/patents/US20120260387

  ```
  https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/US08621662-20140107.XML
  ```

- XML 데이터를 Beautiful Soup을 통해 데이터 추출

beautifulsoup 예제 데이터

- 데이터 다운로드 받기

https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/US08621662-20140107.XML

```
import urllib.request
from bs4 import BeautifulSoup

with open("US08621662-20140107.XML", "r", encoding="utf8") as patent_xml:
    xml = patent_xml.read()  # File을 String으로 읽어오기

soup = BeautifulSoup(xml, "lxml") #lxml parser 호출

#invention-title tag 찾기
invention_title_tag = soup.find("invention-title")
print (invention_title_tag.get_text())
```

beautifulsoup 예제 데이터

- 특허의 출원번호, 출원일, 등록번호, 등록일, 상태, 특허명을 추출

```
<publication-reference>    등록 관련 정보
<document-id>
<country>US</country>
<doc-number>08621662</doc-number> 등록번호
<kind>B2</kind>    상태
<date>20140107</date>        등록일자
</document-id>
</publication-reference>

<application-reference appl-type="utility"> 출원 관련 정보
<document-id>
<country>US</country>
<doc-number>13175987</doc-number> 출원 번호
<date>20110705</date>        출원일
</document-id>
</application-reference>
```

beautifulsoup 예제 데이터

HTML

```
publication_reference_tag = soup.find("publication-reference")
p_document_id_tag = publication_reference_tag.find("document-id")
p_country = p_document_id_tag.find("country").get_text()
p_doc_number = p_document_id_tag.find("doc-number").get_text()
p_kind = p_document_id_tag.find("kind").get_text()
p_date = p_document_id_tag.find("date").get_text()
```

```
application_reference_tag = soup.find("application-reference")
a_document_id_tag = publication_reference_tag.find("document-id")
a_country = p_document_id_tag.find("country").get_text()
a_doc_number = p_document_id_tag.find("doc-number").get_text()
a_date = p_document_id_tag.find("date").get_text()
```

```
<publication-reference>
<document-id>
<country>US</country>
<doc-number>08621662</doc-number>
<kind>B2</kind>
<date>20140107</date>
</document-id>
</publication-reference>

<application-reference appl-
type="utility">
<document-id>
<country>US</country>
<doc-number>13175987</doc-number>
<date>20110705</date>
</document-id>
</application-reference>
```

[연습] ipg140107.xml 분석

- ipa110106.xml 파일은 11년 첫째주에 나온 출원 특허를 모은 파일

  https://s3.ap-northeast-2.amazonaws.com/teamlab-gachon/ipa110106.XML

- 개별 특허들을 나눠서 CSV 형태로 저장 하는 문제

- 개별 특허 시작은 *<?xml version="1.0"*시작함

- 분할된 특허 문서로 부터 특허의 등록번호, 등록일자, 출원 번호, 출원 일자, 상태, 특허 제목을 추출하여 CSV로 만들 것

JavaScript Object Notation

JSON Overview

- JavaScript Object Notation
- 원래 웹 언어인 Java Script의 데이터 객체 표현 방식
- 간결성으로 기계/인간이 모두 이해하기 편함
- 데이터 용량이 적고, Code로의 전환이 쉬움
- 이로 인해 XML의 대체제로 많이 활용되고 있음

JSON Overview


```
{
        "title": "Example Schema",
        "type": "object",
        "properties": {
                "firstName": {
                        "type": "string"
                },
                "lastName": {
                        "type": "string"
                },
                "age": {
                        "description": "Age in years",
                        "type": "integer",
                        "minimum": 0
                }
        },
        "required": ["firstName", "lastName"]
}
```

Sample JSON Schema

Python의 Dict Type과 유사,

key:value 쌍으로 데이터 표시

https://www.flickr.com/photos/xmodulo/26106186415

JSON Overview


```
<?xml version="1.0" encoding="UTF-8" ?>    <employees>
<name>Shyam</name>
<email>shyamjaiswal@gmail.com</email>    </employees>
<employees>
<name>Bob</name>
<email>bob32@gmail.com</email>
</employees>    <employees>
<name>Jai</name>
<email>jai87@gmail.com</email>
</employees>
```

```
{"employees":[
  {"name":"Shyam",
"email":"shyamjaiswal@gmail.com"},
{"name":"Bob",
"email":"bob32@gmail.com"},
{"name":"Jai",
"email":"jai87@gmail.com"} ]
}
```

JSON in Python

- json 모듈을 사용하여 손 쉽게 파싱 및 저장 가능
- 데이터 저장 및 읽기는 dict type과 상호 호환 가능
- 웹에서 제공하는 API는 대부분 정보 교환 시 JSON 활용
- 페이스북, 트위터, Github 등 거의 모든 사이트
- 각 사이트 마다 Developer API의 활용법을 찾아 사용

JSON Read

- JSON 파일의 구조를 확인 → 읽어온 후 → Dict Type처럼 처리

JSON Data

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

```
import json

with open("json_example.json", "r", encoding="utf8") as f:
    contents = f.read()
    json_data =  json.loads(contents)
    print(json_data["employees"])
```

JSON Write

- Dict Type으로 데이터 저장 → josn모듈로 Write

```
import json

dict_data = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

with open("data.json", "w") as f:
    json.dump(dict_data, f)
```

JSON Write

- Dict Type으로 데이터 저장 → josn모듈로 Write

```
import json

dict_data = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

with open("data.json", "w") as f:
    json.dump(dict_data, f)
```

Twitter 데이터 가져오기

- Twitter에서 제공하는 Developer API를 사용하여 트위터 데이터 수집

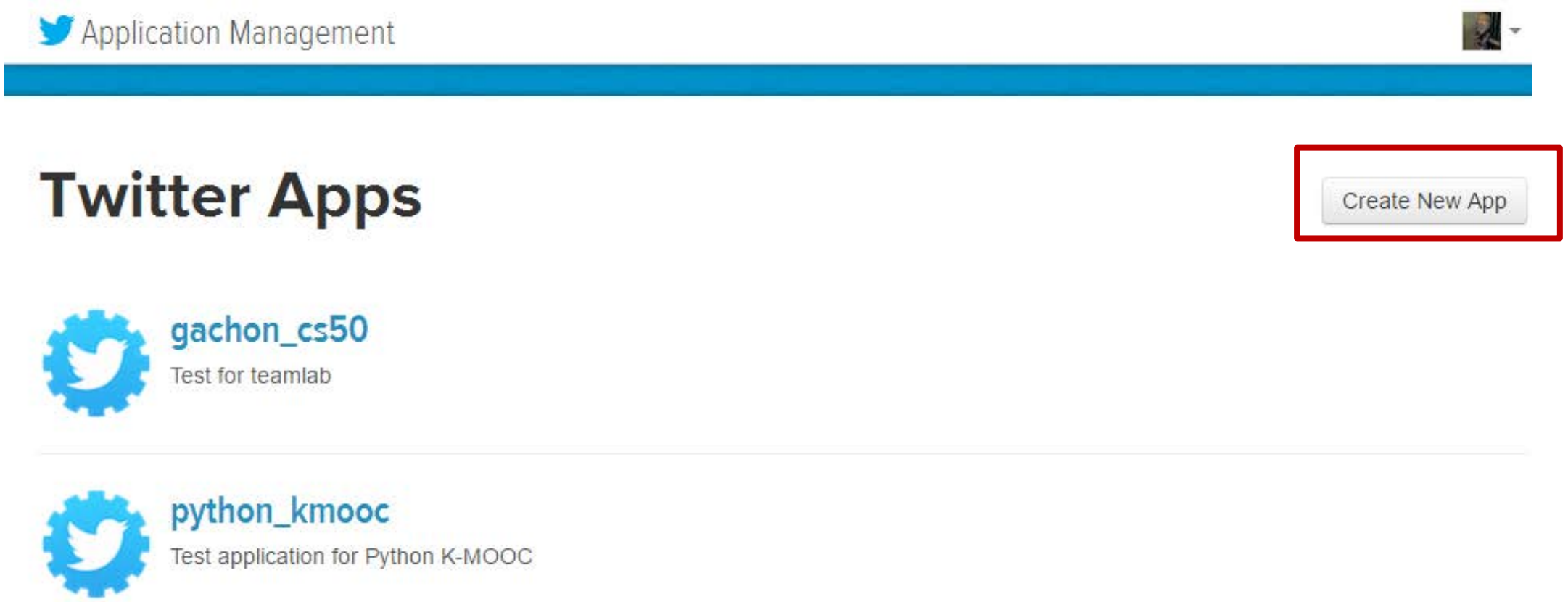
- 수집되는 데이터 형태는 JSON 형태로 제공함

- developer.twitter.com Oauth 인증으로 데이터를 주고 받을 수 있음

- 다양한 기능을 이해하기 위해 API 문서의 공부가 필요
  developer.twitter.com/en/docs

Twitter 데이터 가져오기

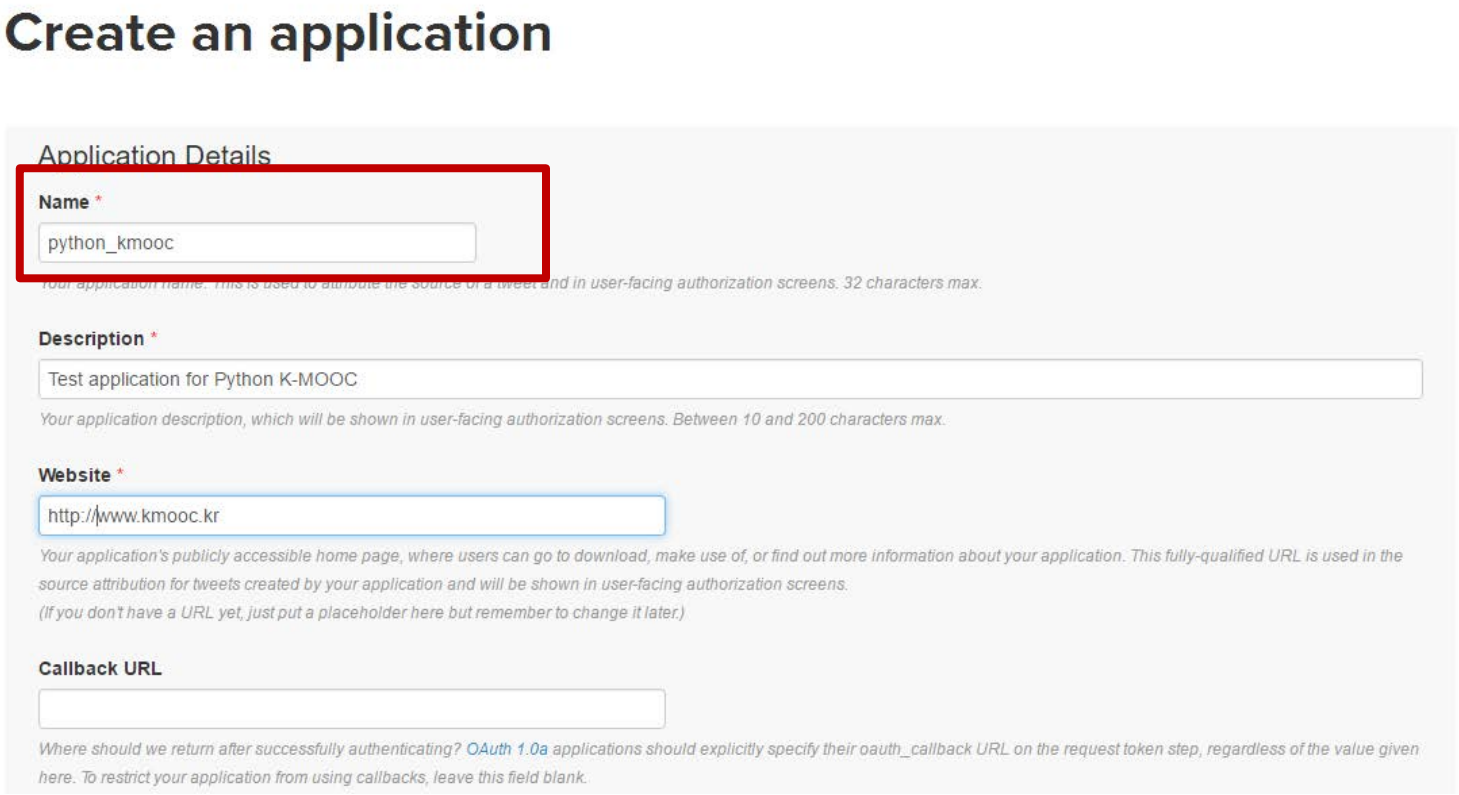
- 트위터 가입후 Twitter App 생성 developer.twitter.com/en/apps



Twitter 데이터 가져오기

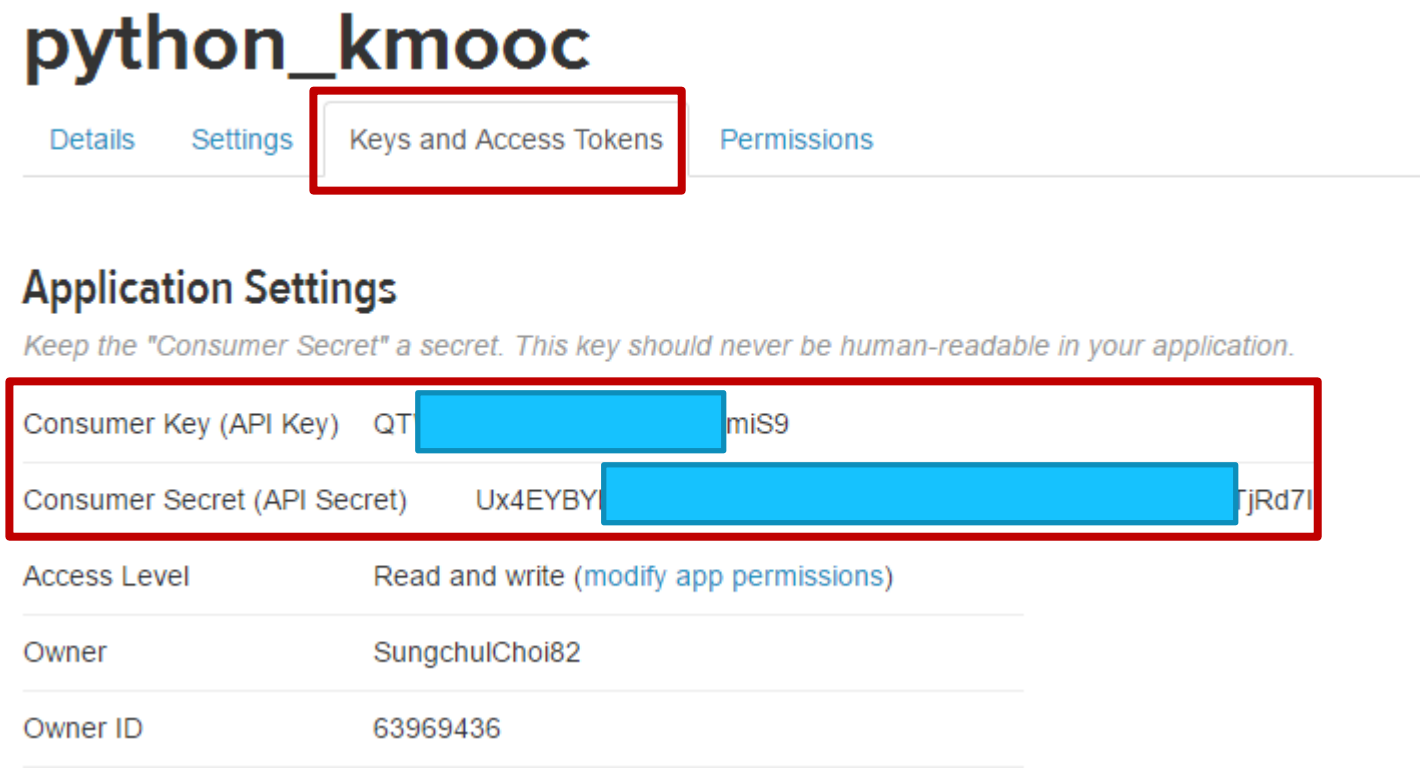
- 트위터 App 정보 입력

developer.twitter.com/en/apps



Twitter 데이터 가져오기

- Keys와 Access Tokens로 가서 API Key 값 확인



Twitter 데이터 가져오기

- Keys와 Access Tokens로 가서 API Key 값 확인

API 사용을 위한 모듈 설치

- conda 가상 환경으로 requests 와 oauthlib 설치

```
acitvate python_mooc
conda install requests
pip install requests-oauthlib
```

- oauth 접속 권한 받기

```
import requests
from requests_oauthlib import OAuth2

consumer_key = '확인한 consumer_key'
consumer_secret = '확인한c onsumer_secret'
access_token = '확인한 access_token'
access_token_secret = '확인한 access_token_secret'

oauth = OAuth2(client_key=consumer_key, client_secret=consumer_secret,
               resource_owner_key=access_token,
resource_owner_secret=access_token_secret)
```

- 특정 계정의 타임라인 데이터 가져오기

```
# Twitter REST api // screen_name 은 트위터 계정명
url =
'https://api.twitter.com/1.1/statuses/user_timeline.json?screen_name={0}'.format('naver_d2')
r = requests.get(url=url,auth=oauth)
statuses =  r.json()

for status in statuses:
    print (status['text'], status['created_at'])
```

End of Document
Thank You.