데이터 분석 기초

- 웹 크롤링
  - 원하는 정보의 태그 추출
  - 음원 사이트 크롤링
  - 크롤링 정보 합하기

2021년 4월 15일

Beautifulsoup을 이용한 정보 찾기

HTML 정보 찾기(2) - 상위 구조 활용

- **태그 속성만으로 찾기 어려운 경우 사용**
- **어떤 부모 태그 아래 있는지 등의 정보를 추가하여 검색**
- **한 단계 아래를 지정할 때는 > 기호 사용**
  - 상위 태그 : 부모 태그, 하위 태그 : 자식 태그
- **1개 이상의 여러 단계 아래를 지정 시 띄어쓰기(빈 칸) 사용**
  - 상위 태그 : 부모 태그, 하위 태그 : 자손 태그
- **태그 구조로 위치 찾기 예**
  - 바나나 검색

    tags_name = soup.select('span.name')

```
[<span class="name"> 바나나 </span>, <span class="name"> 파인애플 </span>]
```

  - 파인애플 제외 바나나 찾기
    - 바나나가 속한 부모 태그 정보 추가

    tags_banana1 =

    soup.select('#fruits1 > span.name')

```
[<span class="name"> 바나나 </span>]
```

변수 html, soup

```
<html>
    <head>
    </head>
    <body>
        <h1> 우리동네시장</h1>
            <div class = 'sale'>
                <p id='fruits1' class='fruits'>
                    <span class = 'name'> 바나나 </span>
                    <span class = 'price'> 3000원 </span>
                    <span class = 'inventory'> 500개 </span>
                    <span class = 'store'> 가나다상회 </span>
                    <a href = 'https://map.kakao.com/' > 홈페이지 </a>
                </p>
            </div>
            <div class = 'prepare'>
                <p id='fruits2' class='fruits'>
                    <span class ='name'> 파인애플 </span>
                </p>
            </div>
    </body>
</html>
```

Beautifulsoup을 이용한 정보 찾기

- **태그 구조로 위치 찾기**
  - 바나나 검색 다른 예

```
tags_banana2 = soup.select('div.sale > #fruits1 > span.name')
tags_banana3 = soup.select('div.sale span.name')
print(tags_banana2)
print(tags_banana3)
```

변수 html, soup

```
<html>
    <head>
    </head>
    <body>
        <h1> 우리동네시장</h1>
            <div class = 'sale'>
                <p id='fruits1' class='fruits'>
                    <span class = 'name'> 바나나 </span>
                    <span class = 'price'> 3000원 </span>
                    <span class = 'inventory'> 500개 </span>
                    <span class = 'store'> 가나다상회 </span>
                    <a href = 'https://map.kakao.com/' > 홈페이지 </a>
                </p>
            </div>
            <div class = 'prepare'>
                <p id='fruits2' class='fruits'>
                    <span class ='name'> 파인애플 </span>
                </p>
            </div>
    </body>
</html>
```

```
[<span class="name"> 바나나 </span>]
[<span class="name"> 바나나 </span>]
```

- tags_banana2
  - 상위태그1(div.sale) 바로 아래에 있는
  - 상위태그2(#fruits1) 바로 아래에 있는
  - 태그(span.name) 찾음
- tags_banana3
  - 상위태그1(div.sale) 바로 아래에 있는 태그 뿐 아니라 몇 단계 아래의 태그 중에서 태그 정보(span.name) 찾음
- > 와 빈칸을 이용한 태그 찾기

Beautifulsoup을 이용한 정보 찾기

정보 가져오기(1) - 태그 그룹에서 하나의 태그 선택

- soup.select('조건') : 조건에 해당하는 모든 태그 찾음 => 그룹 형태로 결과 확인
- 태그 그룹에서 개별 태그에 접근하기 위해서는
  - 인덱스 번호를 활용

```
1  tags = soup.select('span.name')
2  print(tags)
3  tag_1 = tags[0]    #인덱스 번호로 하나의 태그 지정하기
4  print(tag_1)
```

```
[<span class="name"> 바나나 </span>, <span class="name"> 파인애플 </span>]
<span class="name"> 바나나 </span>
```

tags

  - 반목문

```
tags = soup.select('span.name')
for tag in tags:      # 반복문으로 태그 그룹에서 각각의 태그 선택하여 활용하기
    print(tag) 사용
```

```
<span class="name"> 바나나 </span>
<span class="name"> 파인애플 </span>
```

Beautifulsoup을 이용한 정보 찾기

정보 가져오기(2) - 선택한 태그에서 정보 가져오기

- **인덱스 번호나 반복문을 활용해 원하는 태그 선택 후**
  - 화면에 보이는 글 부분을 가져오거나(.text) => 브라우저에 표시되는 정보를 수집하는 일이 많기에 .text 명령을 자주 활용
  - 태그 내 속성 값을 가져옴(['속성명'])
  - 화면에 보이지 않는 URL 주소를 수집하기 위해 ['href']도 필요
    - 하이퍼링크는 <a href = 'URL주소'> 형식으로 작성됨
  - 태그에서 정보 가져오기
    - content = tag.text # 태그에서 화면에 보이는 텍스트 부분만 가져오기
    - attribute = tag['속성명'] # 태그 내 속성값 가져오기
  - URL 정보 추출 예

```
tags = soup.select('a')
tag = tags[0]
content = tag.text
print(content)
link = tag['href']
print(link)
```

홈페이지
https://map.kakao.com/

변수 html, soup

```
<html>
    <head>
    </head>
    <body>
        <h1> 우리동네시장</h1>
            <div class = 'sale'>
                <p id='fruits1' class='fruits'>
                    <span class = 'name'> 바나나 </span>
                    <span class = 'price'> 3000원 </span>
                    <span class = 'inventory'> 500개 </span>
                    <span class = 'store'> 가나다상회 </span>
                    <a href = 'https://map.kakao.com/' > 홈페이지 </a>
                </p>
            </div>
            <div class = 'prepare'>
                <p id='fruits2' class='fruits'>
                    <span class ='name'> 파인애플 </span>
                </p>
            </div>
    </body>
</html>
```

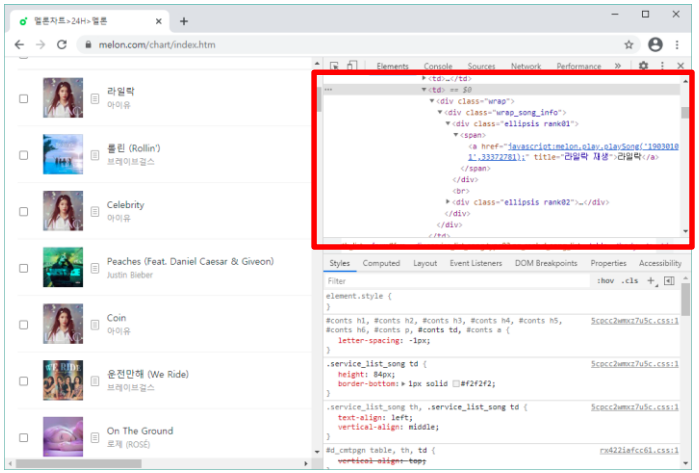
멜론 노래 순위 정보 크롤링(1)

멜론 사이트(www.melon.com) 크롤링

- 인기챠트의 상위 100곡 정보 크롤링
- 크롤링 단계
  - 크롬드라이버 실행 => 크롬 브라우저 열기
  - 멜론 인기차트 웹페이지 접속
  - HTML 다운로드 및 BeautifulSoup 읽기

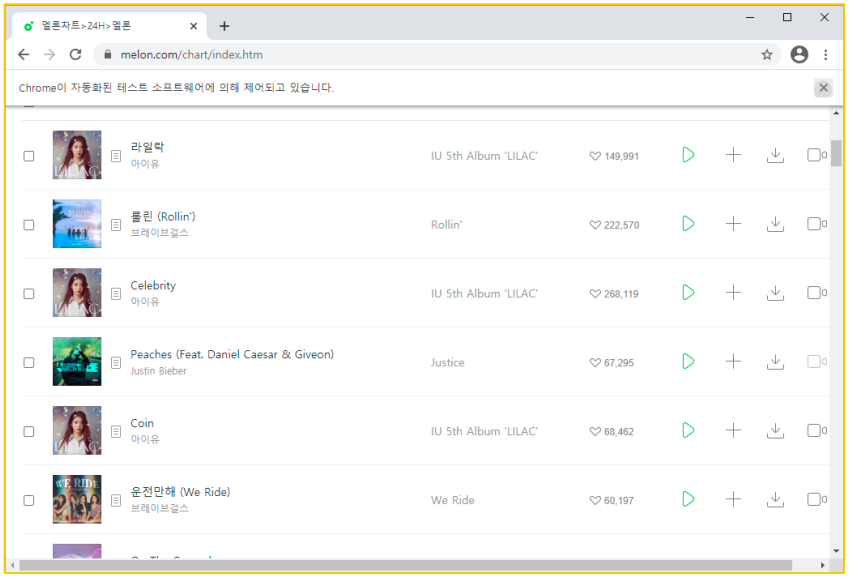
```
from bs4 import BeautifulSoup
html = driver.page_source
soup = BeautifulSoup(html, 'html.parser')
```

  - 변수 soup 노래 정보 포함하는 태그 찾기
    - [F12 키] -[라일락 노래에 마우스포인터 이동]
    - [오른쪽 마우스 버튼]-[검사]



```
from selenium import webdriver
driver = webdriver.Chrome('d:/chromedriver.exe')
```

```
url = 'http://www.melon.com/chart/index.htm'
driver.get(url)
```



멜론 노래 순위 정보 크롤링(2)

- **크롤링 단계(계속)**
  - 노래 한 곡의 정보를 가지는 태그 <table> 아래 <tbody> 아래 <tr>
  - 노래 태그 찾기
    - soup.select('table > tbody > tr') : table 태그 아래 tbody 태그 아래 tr 태그 모두 찾기
    - len(songs) : 해당 원소가 몇 개인지 확인 => 해당 조건을 만족하는 태그는 100개
    - songs[0] : 선택된 태그 중 첫 번째 태그를 화면에 출력

```
1 songs = soup.select('table > tbody > tr')
2 print(len(songs))
3 print(songs[0])
```

```
100
<tr data-song-no="33372781">
<td><div class="wrap t_right"><input class="input_check" name="input_check" title="라일락 곡 선택" type="checkbox" value="33372781"/>
</div></td>
<td><div class="wrap">
<a class="image_typeAll" href="javascript:melon.link.goAlbumDetail('10554246');" title="IU 5th Album 'LILAC'">
<img alt="IU 5th Album 'LILAC' - 페이지 이동" height="60" onerror="WEBPOCIMG.defaultAlbumImg(this);" src="https://cdnimg.melon.co.kr/cm2/album/images/105/54/246/10554246_20210325161233_500.jpg/melon/resize/120/quality/80/optimize" width="60"/>
<span class="bg_album_frame"></span>
</a>
</div></td>
<td><div class="wrap">
<a class="btn button_icons type03 song_info" href="javascript:melon.link.goSongDetail('33372781');" title="라일락 곡정보"><span class="none">곡정보</span></a>
</div></td>
<td><div class="wrap">
<div class="wrap_song_info">
<div class="ellipsis rank01"><span>
<a href="javascript:melon.play.playSong('19030101',33372781);" title="라일락 재생">라일락</a>
</span></div>
<br/>
<div class="ellipsis rank02">
<a href="javascript:melon.link.goArtistDetail('261143');" title="아이유 - 페이지 이동">아이유</a><span class="checkEllipsis" style="display: none;"><a href="javascript:melon.link.goArtistDetail('261143');" title="아이유 - 페이지 이동">아이유</a></span>
</div>
</div>
```

멜론 노래 순위 정보 크롤링(3)

- **크롤링 단계(계속)**
  - 한 곡의 노래 태그에 해당하는 HTML 코드에서 노래 제목과 가수 검색
  - 1위 곡을 먼저 찾은 뒤 반복문으로 100개 전체의 노래와 가수 찾기
    - 한 개의 곡 정보 찾기

```
song = songs[0]
```

```
title = song.select('a')
len(title)
```

```
6
```

```
title = song.select('span > a')
len(title)
```

```
2
```

```
title = song.select('div.ellipsis.rank01 > span > a')
len(title)
```

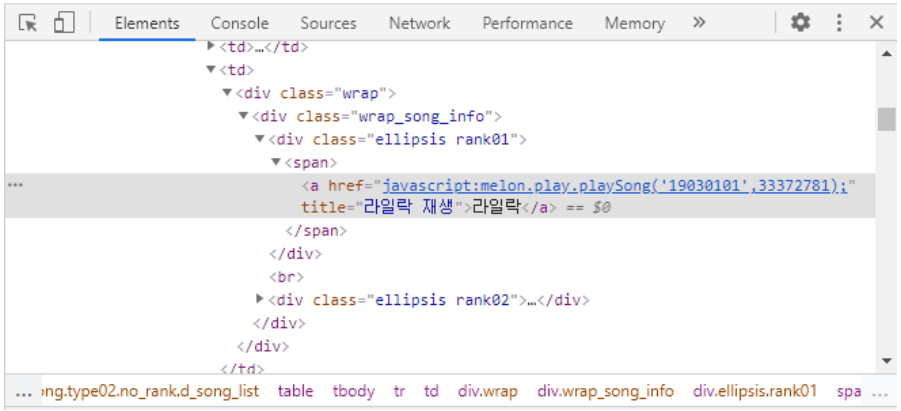
```
1
```

```
title = song.select('div.ellipsis.rank01 > span > a')[0].text
title
```

```
'라일락'
```

```
singer = song.select('div.ellipsis.rank02 > a')
len(singer)
```

```
1
```



- **인덱스 0 : 첫번째 곡**
- **곡 제목은 태그명이 a 이면서 href와 title을 속성으로 가지는 태그에 정보가 있음**
  - a 태그를 찾아보면 6개 : 곡 제목이 포함된 태그만을 가져올 수 없음
  - span > a는 2개 : a태그 상위가 span임을 확인 후 검색 조건을 추가 => 아직 못 가져옴
  - div.ellipsis.rank01 > span > a 는 1개 : span 태그 상위 태그 div 태그 정보 추가
- **동일하게 가수 영역 찾아 출력**
  - div.ellipsis.rank02 > span > a 는 1개

멜론 노래 순위 정보 크롤링(3)

- **크롤링 단계(계속)**
  - 멜론 100위 노래순위 정보 검색

```
for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    print(title, singer, sep = ' | ')
```

  - 멜론 인기차트 상위 100곡 크롤링(정리)

```
#from selenium import webdriver
driver = webdriver.Chrome('d:/chromedriver.exe')

url = 'http://www.melon.com/chart/index.htm'
driver.get(url)

#from bs4 import BeautifulSoup
html = driver.page_source
soup = BeautifulSoup(html, 'html.parser')

songs = soup.select('tbody > tr')

for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    print(title, singer, sep = ' | ')
```

```
라일락 | 아이유
롤린 (Rollin') | 브레이브걸스
Celebrity | 아이유
Peaches (Feat. Daniel Caesar & Giveon) | Justin Bieber
Coin | 아이유
운전만해 (We Ride) | 브레이브걸스
On The Ground | 로제 (ROSÉ)
내 손을 잡아 | 아이유
LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지) | 양요섭
Dynamite | 방탄소년단
밤하늘의 별을(2020) | 경서
밝게 빛나는 별이 되어 비춰줄게 | 송이한
봄 안녕 봄 | 아이유
잠이 오질 않네요 | 장범준
Flu | 아이유
에잇(Prod.&Feat. SUGA of BTS) | 아이유
Blueming | 아이유
취기를 빌려 (취향저격 그녀 X 산들) | 산들
아이와 나의 바다 | 아이유
VVS (Feat. JUSTHIS) (Prod. GroovyRoom) | 미란이 (Mirani)
어푸 (Ah puh) | 아이유
나랑 같이 걸을래 (바른연애 길잡이 X 적재) | 적재
Lovesick Girls | BLACKPINK
흔들리는 꽃들 속에서 네 샴푸향이 느껴진거야 | 장범준
이 밤을 빌려 말해요 (바른연애 길잡이 X 10CM) | 10CM
돌림노래 (Feat. DEAN) | 아이유
그날에 나는 맘이 편했을까 | 이예준
오래된 노래 | 스탠딩 에그
어떻게 이별까지 사랑하겠어, 널 사랑하는 거지 | AKMU (악동뮤지션)
힘든 건 사랑이 아니다 | 임창정
12:45 (Stripped) | Etham
추억은 만남보다 이별에 남아 | 정동하
사실 나는 (Feat.전건호) | 경서예지
```

이하 생략

selenium을 활용한 크롤링

- **앞에서 실행한 크롤링은 +BeautifulSoup으로 원하는 정보 가져오는 방식**
  - 예: soup.select('tbody > tr')
- **원하는 정보를 가져오는 다른 방식**
  - 태그 구조 정보인 CSS Selector 사용하는 방식
  - driver.find_elements_by_css_selector('조건') : 원하는 조건에 해당하는 태그를 모두 찾아옴
  - 조건 : 태그명, class 명, id 값, 부모 태그 등의 구조 정보 지정 => BeautifulSoup에서 작성하는 방식과 동일

```
#from selenium import webdri
driver = webdriver.Chrome('c

url = 'http://www.melon.com/
driver.get(url)

#from bs4 import BeautifulSo
html = driver.page_source
soup = BeautifulSoup(html, '

songs = soup.select('tbody > tr')

for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    print(title, singer, sep = ' | ')
```

```
from selenium import webdriver
```

```
driver = webdriver.Chrome('../chromedriver89.exe')
```

```
url = 'http://www.melon.com/chart/index.htm'
driver.get(url)
songs = driver.find_elements_by_css_selector('tbody > tr')
for song in songs:
    title = song.find_elements_by_css_selector('div.ellipsis.rank01 > span > a')[0].text
    singer = song.find_elements_by_css_selector('div.ellipsis.rank02 > a')[0].text
    print(title, singer, sep = ' | ')
```

selenium만을 활용한 크롤링

- **웹 페이지에 계속 접속한 상태로 정보를 가져옴**
- **html = driver.page_source 로 HTML 정보를 다운로드하는 과정은 필요 없음**
- **BeautifulSoup을 같이 사용**
  - 웹 페이지의 HTML을 다운로드 후 필요한 정보를 찾는 방식 => 속도가 빠름
- **selenium 만을 사용하는 경우에는 웹 페이지에 계속 접속**
  - 필요 내용 찾는데 오랜 시간 소요
  - 특정 위치를 선택한 후 클릭하거나 값을 입력/삭제하는 등의 브라우저 조작 가능
- **권장 방법**
  - selenium을 이용해 원하는 웹 페이지에 접속하고 값을 입력하거나 클릭하는 등의 작업 진행
  - 필요한 정보가 나타났을 때에는 HTML을 다운로드 한 뒤 필요 정보 추출

	selenium + BeautifulSoup	selenium만 이용
웹 페이지 접속	HTML 페이지 다운로드 후 브라우저 영향 없음	웹 페이지 연결 유지 필요
웹 페이지 동작	불가능	클릭, 입력 등 조작 가능
크롤링 속도	빠름	느림

여러 음원 서비스 순위 수집 및 정리

- 멜론, 벅스, 지니에서 제공하는 인기차트 상위 곡 정보 크롤링
- 크롤링한 결과를 엑셀로 저장
- 각 음원 서비스 별로 저장한 엑셀 파일 통합
- 총 250곡 정보 수집 후 엑셀 저장

서비스	순위	타이틀	가수
Melon	1	라일락	아이유
Melon	2	롤린 (Rollin')	브레이브걸스
Melon	99	아무노래	지코 (ZICO)
Melon	100	사이렌 Remix (Feat. UNEDUCATED KID, Paul Blanco)	호미들
Bugs	1	그냥 안아달란 말야	다비치
Bugs	2	Atlantis	SHINee (샤이니)
Bugs	99	슬픔활용법	김범수
Bugs	100	34+35	Ariana Grande(아리아나 그란데)
Genie	1	그냥 안아달란 말야	다비치
Genie	2	Atlantis	SHINee (샤이니)
Genie	49	Savage Love (Laxed - Siren Beat) (BTS Remix)	Jawsh 685 & Jason Derulo & 방탄소년단
Genie	50	2002	Anne-Marie

멜론 크롤링 결과를 리스트에 저장

```
from selenium import webdriver
```

```
from bs4 import BeautifulSoup
```

```
driver = webdriver.Chrome('../chromedriver89.exe')
```

```
url = 'http://www.melon.com/chart/index.htm'
driver.get(url)
html = driver.page_source
soup = BeautifulSoup(html, 'html.parser')
```

- 멜론 인기차트 크롤링 코드
- song_data[] : 엑셀 파일을 만들기 위한 정보 저장
- rank : 곡 순위 정보 저장하는 변수
- 각 곡에서 추출한 정보를 song에 하나씩 저장 후 처리
  - song에 저장된 정보를 [서비스, 순위, 타이틀, 가수] 형식으로 song_data[]에 추가
  - 서비스 : 현재 크롤링하는 서비스 이름인 'Melon' 입력
  - 순위 정보 rank 값 증가시켜 저장
  - 멜론 100개 정보 저장

```
song_data = []
rank = 1
songs = soup.select('tbody > tr')
for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    song_data.append(['Melon', rank, title, singer])
    rank = rank + 1
```

멜론 리스트 데이터를 엑셀 파일로 저장

- song_data 리스트를 데이터프레임 pd_data로 생성

```
1  song_data
```

```
[['Melon', 1, '라일락', '아이유'],
 ['Melon', 2, "롤린 (Rollin')", '브레이브걸스'],
 ['Melon', 3, 'Celebrity', '아이유'],
 ['Melon', 4, 'Peaches (Feat. Daniel Caesar & Giveon)', 'Justin Bieber'],
 ['Melon', 5, 'Coin', '아이유'],
 ['Melon', 6, '운전만해 (We Ride)', '브레이브걸스'],
 ['Melon', 7, 'On The Ground', '로제 (ROSÉ)'],
 ['Melon', 8, '내 손을 잡아', '아이유'],
 ['Melon', 9, 'LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지)', '양요섭'],
 ['Melon', 10, 'Dynamite', '방탄소년단'],
```

```
1  columns = ['서비스', '순위', '타이틀', '가수']
2  pd_data = pd.DataFrame(song_data, columns=columns)
3  pd_data.head()
```

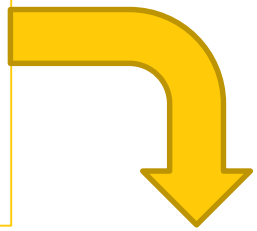
	서비스	순위	타이틀	가수
0	Melon	1	라일락	아이유
1	Melon	2	롤린 (Rollin')	브레이브걸스
2	Melon	3	Celebrity	아이유
3	Melon	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
4	Melon	5	Coin	아이유

멜론 리스트 데이터를 엑셀 파일로 저장

- 데이터프레임 pd_data를 엑셀 파일 melon.xlsx로 생성

```
1  pd_data.to_excel('./music_files/melon.xlsx', index=False)
```

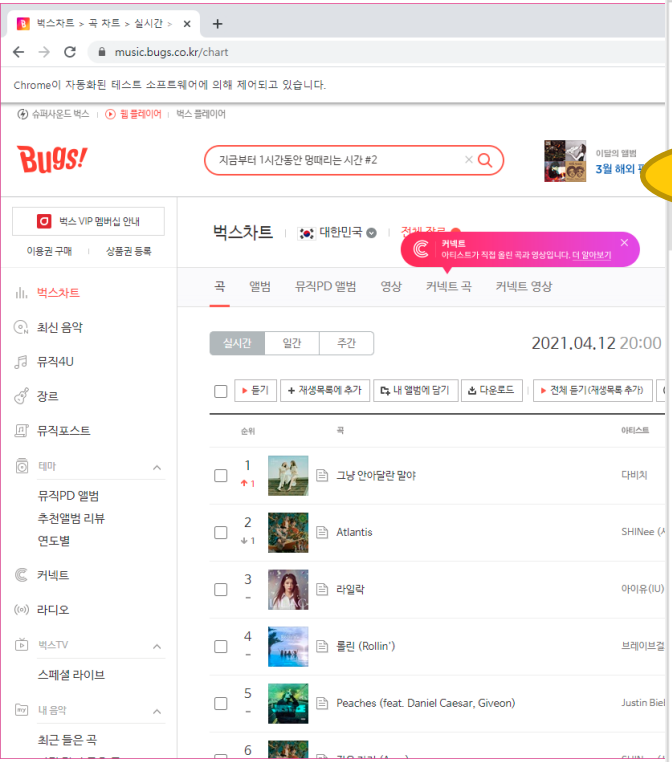
	서비스	순위	타이틀	가수
0	Melon	1	라일락	아이유
1	Melon	2	롤린 (Rollin')	브레이브걸스
2	Melon	3	Celebrity	아이유
3	Melon	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
4	Melon	5	Coin	아이유



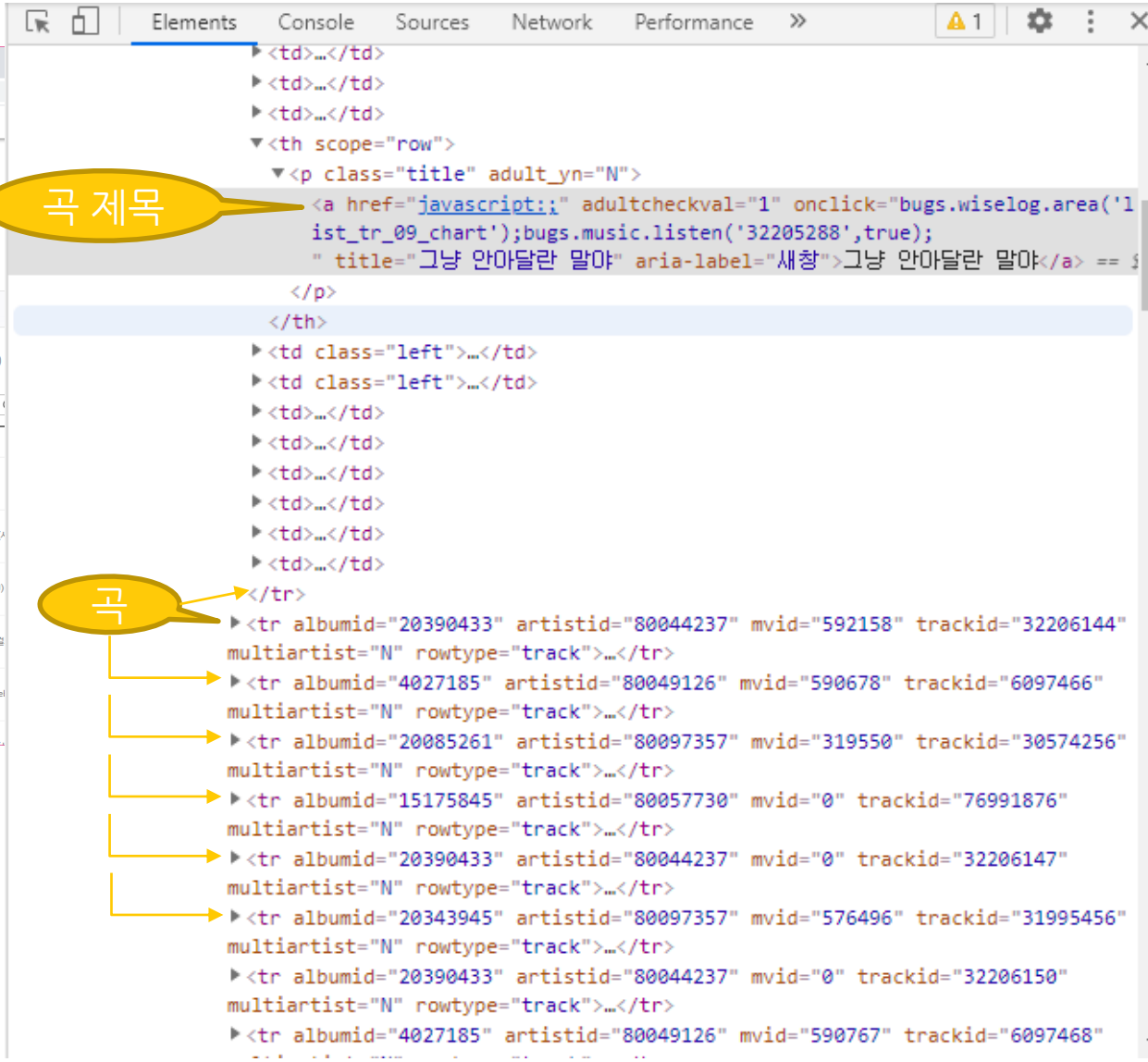
서비스	순위	타이틀	가수
Melon	1	라일락	아이유
Melon	2	롤린 (Rollin')	브레이브걸스
Melon	3	Celebrity	아이유
Melon	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
Melon	5	Coin	아이유
Melon	6	운전만해 (We Ride)	브레이브걸스
Melon	7	On The Ground	로제 (ROSÉ)
Melon	8	내 손을 잡아	아이유
Melon	9	LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지)	양요섭
Melon	10	Dynamite	방탄소년단

.....

벅스 사이트 살펴보기




```
1  url = 'https://music.bugs.co.kr/chart'
2  driver.get(url)
3  html = driver.page_source
4  soup = BeautifulSoup(html, 'html.parser')
```



벅스 크롤링을 위한 태그 찾기

- **멜론 크롤링과 동일**
- **웹 브라우저 접속 주소와 추출할 태그 정보 다름**
- **노래 한 곡의 정보가 담긴 태그 찾기**
  - 개발자 도구의 [Elements] 탭에서 선택된 태그 정보 확인
  - 각 tr 태그를 선택할 때 마다 곡이 선택됨
  - tr 태그 내에 노래 제목, 가수가 포함됨
    - tr 태그 104개 : 다른 태그 찾기
  - 상위 tbody 태그 추가
    - tbody > tr 은 103개
  - 상위 table 태그 추가
    - table > tbody > tr 은 103개
  - table 태그에 있는 byChart 클래스 추가
    - table.byChart>tbody>tr은 100개
    - table.list > tbody > tr도 100개
    - table.trackList>tbody>tr도 100개
  - table.byChart p.title도 100개
    - 노래 제목이 있는 부분만 가져옴

```
songs = soup.select('tr')
print(len(songs))
```

```
104
```

```
songs = soup.select('tbody>tr')
print(len(songs))
```

```
103
```

```
songs = soup.select('table>tbody>tr')
print(len(songs))
```

```
103
```

```
songs = soup.select('table.byChart>tbody>tr')
print(len(songs))
```

```
100
```

벅스 크롤링을 위한 태그 찾기

- **노래 제목과 가수 정보가 담긴 태그 찾기**

```
songs = soup.select('table.byChart>tbody>tr')
print(len(songs))
```

```
100
```

```
print(songs[0])
```

```
<tr albumid="20390261" artistid="80041466" multiartist="N" mvid="591929" rowtype="track" trackid="32205288">
<input name="_isStream" type="hidden" value="32205288"/>
<input name="_isDown" type="hidden" value="32205288"/>
<td class="check"><input buyminquality="T" disc_id="1" name="check" title="그냥 안아달란 말야" type="checkbox" value="32205288"/></td>
<td>
<div class="ranking">
<strong>1</strong>
<p class="change none"><em>0</em><span>변동없음</span></p>
</div>
</td>
<td>
<a class="thumbnail" href="https://music.bugs.co.kr/album/20390261?wl_ref=list_
">
<span class="mask"></span>
```

- songs 중 첫 번째 노래 가져오기
  - 노래 제목 : a 태그 내 문자열
    - » a 추출 11개 => 다른 추출 태그 찾아야 함
    - » 상위 p 태그 추가 후 추출 : p > a는 2개
    - » p태그 클래스 사용 추출 : p.title > a는 1개

```
song = songs[0]
title = song.select('a')
len(title)
```

```
11
```

```
title = song.select('p > a')
len(title)
```

```
2
```

```
title = song.select('p.title > a')
len(title)
```

```
1
```

```
title = song.select('p.title > a')[0].text
title
```

```
'그냥 안아달란 말야'
```

벅스 크롤링을 위한 태그 찾기

- **노래 제목과 가수 정보가 담긴 태그 찾기**
  - songs 중 첫 번째 노래 가져오기
    - 가수 : 노래 제목과 동일하게 p.artist>a에서 추출

```
singer = song.select('p.artist > a')[0].text
singer
```

```
'다비치'
```

- **노래 제목과 가수 정보 추출**

```
songs = soup.select('table.byChart>tbody>tr')
for song in songs:
    title = song.select('p.title > a')[0].text
    singer = song.select ('p.artist > a')[0].text
    print(title, singer, sep = ' | ')
```

```
그냥 안아달란 말야 | 다비치
라일락 | 아이유(IU)
롤린 (Rollin') | 브레이브걸스(Brave Girls)
Peaches (feat. Daniel Caesar, Giveon) | Justin Bieber(저스틴 비버)
Atlantis | SHINee (샤이니)
운전만해 (We Ride) | 브레이브걸스(Brave Girls)
Celebrity | 아이유(IU)
안녕이란 (Two Letters) | 황치열
Coin | 아이유(IU)
어푸 (Ah puh) | 아이유(IU)
On The Ground | 로제(ROSÉ)
ASAP | STAYC(스테이씨)
Leave The Door Open | Bruno Mars(브루노 마스)
Flu | 아이유(IU)
```

벅스 크롤링 결과를 리스트에 저장

- **song_data[] : 엑셀 파일을 만들기 위한 정보 저장 리스트**
- **rank : 곡 순위 정보 저장하는 변수**
- **각 곡에서 추출한 정보를 song에 하나씩 저장 후 처리**
  - song에 저장된 정보를 [서비스, 순위, 타이틀, 가수] 형식으로 song_data[]에 추가
  - 서비스 : 현재 크롤링하는 서비스 이름인 'Bugs' 입력
  - 순위 정보 rank 값 증가시켜 저장
  - 벅스 100개 정보 저장

```
song_data = []
rank = 1
songs = soup.select('tbody > tr')
for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    song_data.append(['Melon', rank, title, singer])
    rank = rank + 1
```

```
song_data =[]
rank = 1
songs = soup.select('table.byChart>tbody>tr')
for song in songs:
    title = song.select('p.title > a')[0].text
    singer = song.select ('p.artist > a')[0].text
    song_data.append(['Bugs', rank, title, singer])
    rank = rank + 1
```

벅스 리스트 데이터를 엑셀 파일로 저장

- song_data 리스트를 데이터프레임 pd_data로 생성

```
1  song_data
```

```
[['Bugs', 1, '그냥 안아달란 말야', '다비치'],
 ['Bugs', 2, '라일락', '아이유(IU)'],
 ['Bugs', 3, "롤린 (Rollin')", '브레이브걸스(Brave Girls)'],
 ['Bugs', 4, 'Peaches (feat. Daniel Caesar, Giveon)', 'Justin Bieber(저스틴 비버)'],
 ['Bugs', 5, 'Atlantis', 'SHINee (샤이니)'],
 ['Bugs', 6, '운전만해 (We Ride)', '브레이브걸스(Brave Girls)'],
 ['Bugs', 7, 'Celebrity', '아이유(IU)'],
 ['Bugs', 8, '안녕이란 (Two Letters)', '황치열'],
 ['Bugs', 9, 'Coin', '아이유(IU)'],
 ['Bugs', 10, '어푸 (Ah puh)', '아이유(IU)'],
```

```
1  pd_data = pd.DataFrame(song_data, columns=columns)
2  pd_data.head()
```

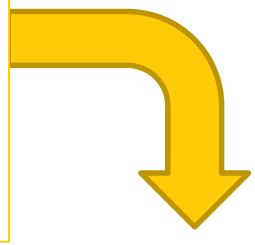
	서비스	순위	타이틀	가수
0	Bugs	1	그냥 안아달란 말야	다비치
1	Bugs	2	라일락	아이유(IU)
2	Bugs	3	롤린 (Rollin')	브레이브걸스(Brave Girls)
3	Bugs	4	Peaches (feat. Daniel Caesar, Giveon)	Justin Bieber(저스틴 비버)
4	Bugs	5	Atlantis	SHINee (샤이니)

벅스 리스트 데이터를 엑셀 파일로 저장

- 데이터프레임 pd_data를 엑셀 파일 bugs.xlsx로 생성

```
1  pd_data.to_excel('./music_files/bugs.xlsx', index=False)
```

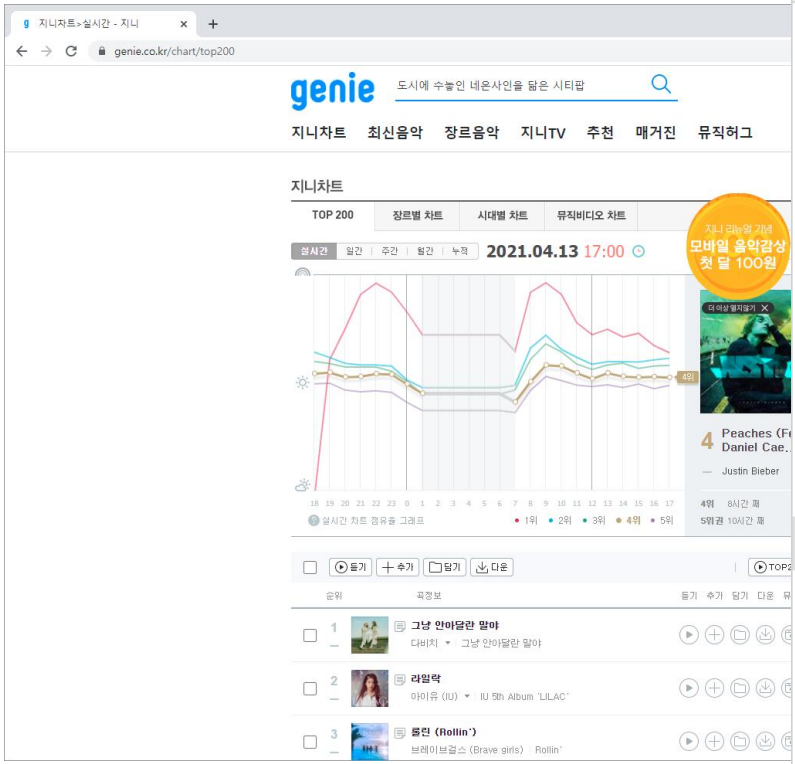
	서비스	순위	타이틀	가수
0	Bugs	1	그냥 안아달란 말야	다비치
1	Bugs	2	라일락	아이유(IU)
2	Bugs	3	롤린 (Rollin')	브레이브걸스(Brave Girls)
3	Bugs	4	Peaches (feat. Daniel Caesar, Giveon)	Justin Bieber(저스틴 비버)
4	Bugs	5	Atlantis	SHINee (샤이니)

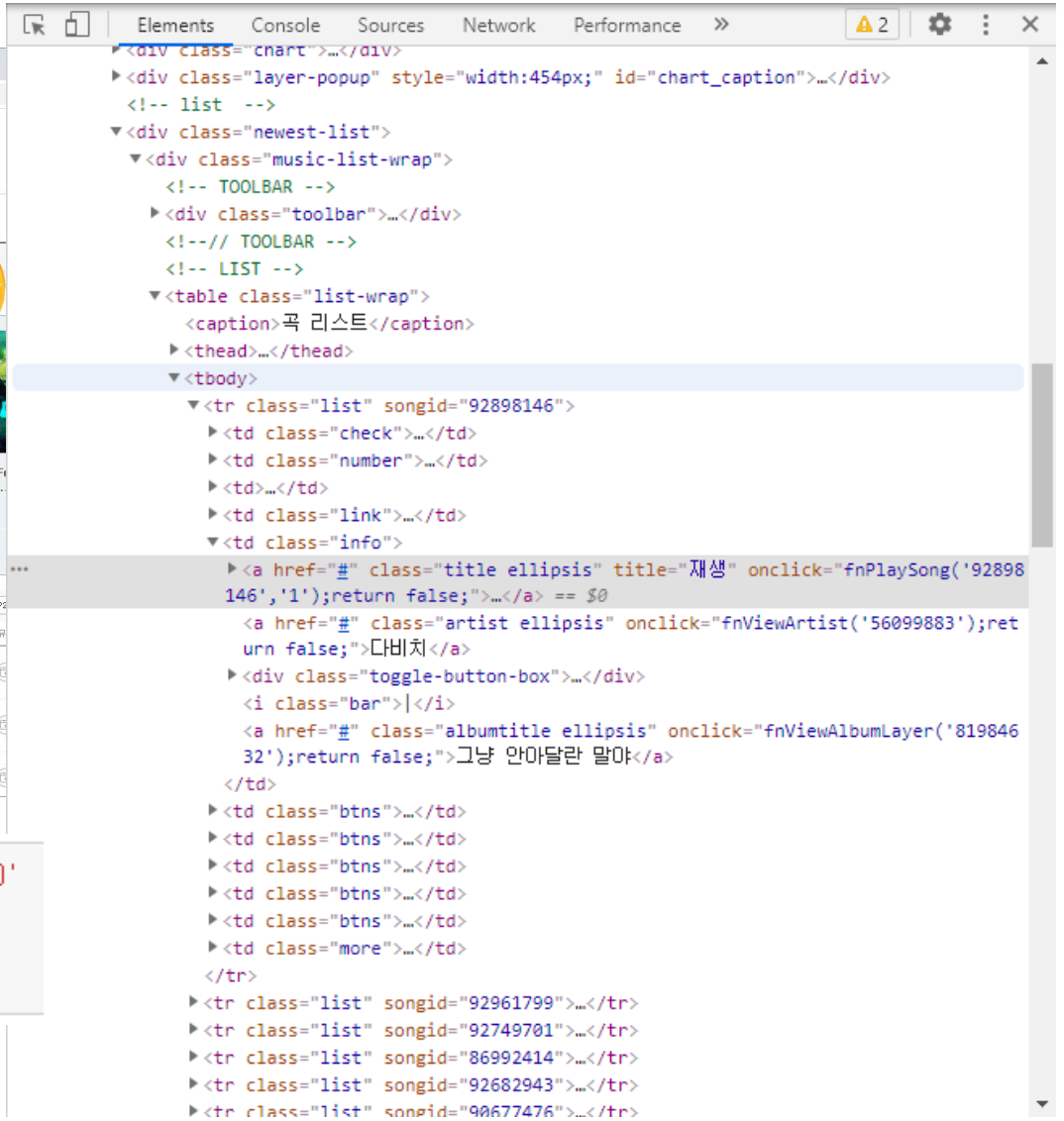


서비스	순위	타이틀	가수
Bugs	1	그냥 안아달란 말야	다비치
Bugs	2	라일락	아이유(IU)
Bugs	3	롤린 (Rollin')	브레이브걸스(Brave Girls)
Bugs	4	Peaches (feat. Daniel Caesar, Giveon)	Justin Bieber(저스틴 비버)
Bugs	5	Atlantis	SHINee (샤이니)
Bugs	6	운전만해 (We Ride)	브레이브걸스(Brave Girls)
Bugs	7	Celebrity	아이유(IU)
Bugs	8	안녕이란 (Two Letters)	황치열
Bugs	9	Coin	아이유(IU)
Bugs	10	어푸 (Ah puh)	아이유(IU)

.....

지니 사이트 살펴보기






```
1  url = 'https://www.genie.co.kr/chart/top200'
2  driver.get(url)
3  html = driver.page_source
4  soup = BeautifulSoup(html, 'html.parser')
```

지니 크롤링을 위한 태그 찾기

- **멜론, 벅스 크롤링과 차이**
  - 1페이지에 50곡
- **노래 한 곡의 정보가 담긴 태그 찾기**
  - 각 tr 태그를 선택할 때 마다 곡이 선택됨
  - tr 태그 내에 노래 제목, 가수가 포함됨
    - tr 태그 51개 : 다른 태그 찾기
  - 상위 tbody 태그 추가 : tbody > tr 은 50개
  - 상위 table 태그 추가 : table > tbody > tr 은 50개

```
1 songs = soup.select('tr')
2 print(len(songs))
```

```
51
```

```
1 songs = soup.select('tbody>tr')
2 print(len(songs))
```

```
50
```

```
1 songs = soup.select('table>tbody>tr')
2 print(len(songs))
```

```
50
```

```
1 print(songs[0])
```

```
<tr class="list" songid="92898146">
<td class="check"><input class="select-check" title="그냥 안아달란 말야" type="checkbox"/></td>
<td class="number">1

                                    <span class="rank">
<span class="rank"><span class="rank-none"><span class="hide">유지</span></span></span>
</span>
</td>
<td><a class="cover" href="#" onclick="fnViewAlbumLayer('81984632');return false;"><span class="mask"></span><img alt="그냥 안아달란
말야" onerror="this.src='//image.genie.co.kr/imageg/web/common/blank_68.gif';" src="//image.genie.co.kr/Y/IMAGE/IMG_ALBUM/081/984/632/
81984632_1618202770849_1_140x140.JPG/dims/resize/Q_80,0"/></a></td>
<td class="link"><a class="btn-basic btn-info" href="#" onclick="fnViewSongInfo('92898146');return false;">곡 제목 정보 페이지</a></td
>
<td class="info">
<a class="title ellipsis" href="#" onclick="fnPlaySong('92898146','1');return false;" title="재생">


                                    그냥 안아달란 말야</a>
<a class="artist ellipsis" href="#" onclick="fnViewArtist('56099883');return false;">다비치</a>
<div class="toggle-button-box">
```

지니 크롤링을 위한 태그 찾기

- **노래 제목과 가수 정보가 담긴 태그 찾기**
  - songs 중 첫 번째 노래 가져오기
    - 노래 제목 추출

```
song = songs[0]
```

```
title = song.select('a.title')
print(len(title))
```

```
1
```

```
title = song.select('a.title')[0].text
title
```

```
'₩n                                    ₩n                                    ₩n                                    ₩n
₩n                                    ₩n                                    ₩n                                    ₩n
그냥 안아달란 말야'
```

```
title = song.select('a.title')[0].text.strip()
title
```

```
'그냥 안아달란 말야'
```

- 가수 추출

```
singer = song.select('a.artist')
len(singer)
```

```
1
```

```
singer = song.select('a.artist')[0].text
singer
```

```
'다비치'
```

지니 크롤링을 위한 태그 찾기

- **노래 제목과 가수 정보 추출**

```
songs = soup.select('tbody>tr')
for song in songs:
    title = song.select('a.title')[0].text.strip()
    singer = song.select('a.artist')[0].text
    print(title, singer, sep=' | ')
```

```
그냥 안아달란 말야 | 다비치
라일락 | 아이유 (IU)
롤린 (Rollin') | 브레이브걸스 (Brave girls)
Peaches (Feat. Daniel Caesar & Giveon) | Justin Bieber
운전만해 (We Ride) | 브레이브걸스 (Brave girls)
Celebrity | 아이유 (IU)
Coin | 아이유 (IU)
LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지) | 양요섭 & 정은지
On The Ground | 로제 (ROSE)
Atlantis | SHINee (샤이니)
내 손을 잡아 | 아이유 (IU)
Dynamite | 방탄소년단
이제 나만 믿어요 | 임영웅
At My Worst | Pink Sweat$
봄 안녕 봄 | 아이유 (IU)
Flu | 아이유 (IU)
어푸 (Ah puh) | 아이유 (IU)
별빛 같은 나의 사랑아 | 임영웅
Blueming | 아이유 (IU)
HERO | 임영웅
돌림노래 (Feat. DEAN) | 아이유 (IU)
아이와 나의 바다 | 아이유 (IU)
밝게 빛나는 별이 되어 비춰줄게 | 송이한
흔들리는 꽃들 속에서 네 샴푸향이 느껴진거야 | 장범준
밤하늘의 별을 (2020) | 경서
에잇 (Prod. & Feat. SUGA of BTS) | 아이유 (IU)
Don't Call Me | SHINee (샤이니)
12 : 45 (Stripped) | etham
```

지니 크롤링 결과를 리스트에 저장

- **song_data[] : 엑셀 파일을 만들기 위한 정보 저장 리스트**
- **rank : 곡 순위 정보 저장하는 변수**
- **각 곡에서 추출한 정보를 song에 하나씩 저장 후 처리**
  - song에 저장된 정보를 [서비스, 순위, 타이틀, 가수] 형식으로 song_data[]에 추가
  - 서비스 : 현재 크롤링하는 서비스 이름인 'Genie' 입력
  - 순위 정보 rank 값 증가시켜 저장
  - 지니 50개 정보 저장

```
song_data = []
rank = 1
songs = soup.select('tbody > tr')
for song in songs:
    title = song.select('div.ellipsis.rank01 > span > a')[0].text
    singer = song.select ('div.ellipsis.rank02 > a')[0].text
    song_data.append(['Melon', rank, title, singer])
    rank = rank + 1
```

```
song_data = []
rank = 1
songs = soup.select('tbody>tr')
for song in songs:
    title = song.select('a.title')[0].text.strip()
    singer = song.select('a.artist')[0].text
    song_data.append(['Genie', rank, title, singer])
    rank = rank + 1
```

```
song_data =[]
rank = 1
songs = soup.select('table.byChart>tbody>tr')
for song in songs:
    title = song.select('p.title > a')[0].text
    singer = song.select ('p.artist > a')[0].text
    song_data.append(['Bugs', rank, title, singer])
    rank = rank + 1
```

지니 리스트 데이터를 엑셀 파일로 저장

- song_data 리스트를 데이터프레임 pd_data로 생성

```
1  song_data
```

```
[['Genie', 1, '그냥 안아달란 말야', '다비치'],
 ['Genie', 2, '라일락', '아이유 (IU)'],
 ['Genie', 3, "롤린 (Rollin')", '브레이브걸스 (Brave girls)'],
 ['Genie', 4, 'Peaches (Feat. Daniel Caesar & Giveon)', 'Justin Bieber'],
 ['Genie', 5, '운전만해 (We Ride)', '브레이브걸스 (Brave girls)'],
 ['Genie', 6, 'Celebrity', '아이유 (IU)'],
 ['Genie', 7, 'Coin', '아이유 (IU)'],
 ['Genie', 8, 'LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지)', '양요섭 & 정은지'],
 ['Genie', 9, 'On The Ground', '로제 (ROSE)'],
 ['Genie', 10, 'Atlantis', 'SHINee (샤이니)'],
```

```
1  pd_data = pd.DataFrame(song_data, columns=columns)
2  pd_data.head()
```

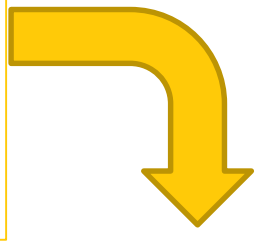
	서비스	순위	타이틀	가수
0	Genie	1	그냥 안아달란 말야	다비치
1	Genie	2	라일락	아이유 (IU)
2	Genie	3	롤린 (Rollin')	브레이브걸스 (Brave girls)
3	Genie	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
4	Genie	5	운전만해 (We Ride)	브레이브걸스 (Brave girls)

지니 리스트 데이터를 엑셀 파일로 저장

- 데이터프레임 pd_data를 엑셀 파일 genie.xlsx로 생성

```
1  pd_data.to_excel('./music_files/genie.xlsx', index=False)
```

	서비스	순위	타이틀	가수
0	Genie	1	그냥 안아달란 말야	다비치
1	Genie	2	라일락	아이유 (IU)
2	Genie	3	롤린 (Rollin')	브레이브걸스 (Brave girls)
3	Genie	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
4	Genie	5	운전만해 (We Ride)	브레이브걸스 (Brave girls)



서비스	순위	타이틀	가수
Genie	1	그냥 안아달란 말야	다비치
Genie	2	라일락	아이유 (IU)
Genie	3	롤린 (Rollin')	브레이브걸스 (Brave girls)
Genie	4	Peaches (Feat. Daniel Caesar & Giveon)	Justin Bieber
Genie	5	운전만해 (We Ride)	브레이브걸스 (Brave girls)
Genie	6	Celebrity	아이유 (IU)
Genie	7	Coin	아이유 (IU)
Genie	8	LOVE DAY (2021) (바른연애 길잡이 X 양요섭, 정은지)	양요섭 & 정은지
Genie	9	On The Ground	로제 (ROSE)
Genie	10	Atlantis	SHINee (샤이니)

.....

엑셀 파일 통합

- 멜론, 벅스, 지니의 인기 차트 크롤링 파일 한 개로 합치기
- pandas 사용
  - appended_data : 통합 데이터 저장할 데이터프레임

```
excel_names = ['./music_files/melon.xlsx', './music_files/bugs.xlsx', './music_files/genie.xlsx']
appended_data = pd.DataFrame()
for name in excel_names:
    pd_data = pd.read_excel(name)
    appended_data = appended_data.append(pd_data)
```

```
appended_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 250 entries, 0 to 49
Data columns (total 4 columns):
서비스     250 non-null object
순위      250 non-null int64
타이틀     250 non-null object
가수      250 non-null object
dtypes: int64(1), object(3)
memory usage: 9.8+ KB
```

```
appended_data.to_excel('./music_files/total.xlsx', index=False)
```