



[스파르타코딩클럽] 웹개발 종합반 - 3주차



매 주차 강의자료 시작에 PDF파일을 올려두었어요!

▼ 단축키 모음

▼ 새로고침

- `F5`

▼ 저장

- Windows: `Ctrl + S`
- macOS: `command + S`

▼ 전체선택

- Windows: `Ctrl + A`
- macOS: `command + A`

▼ 잘라내기

- Windows: `Ctrl + X`
- macOS: `command + X`

▼ 콘솔창 줄바꿈

- `shift + enter`

▼ 코드정렬

- Windows: `Ctrl + Alt + L`
- macOS: `option + command + L`

▼ 들여쓰기

- `Tab`
- 들여쓰기 취소 : `Shift + Tab`

▼ 주석

- Windows: `Ctrl + /`
- macOS: `command + /`

[수업 목표]

1. 파이썬 기초 문법을 안다.
2. 원하는 페이지를 크롤링 할 수 있다.
3. pymongo를 통해 mongoDB를 제어할 수 있다.

[목차]

- 01. 3주차 설치
- 02. 연습 겸 복습 - 스파르타피디아에 OpenAPI 붙여보기
- 03. 파이썬 시작하기
- 04. 파이썬 기초공부
- 05. 파이썬 패키지 설치하기
- 06. 패키지 사용해보기
- 07. 웹스크래핑(크롤링) 기초

08. Quiz 웹스크래핑(크롤링) 연습
09. DB개괄
10. mongoDB 시작하기
11. mongoDB 연결하기
12. pymongo로 DB조작하기
13. 웹스크래핑 결과 저장하기
14. Quiz 웹스크래핑 결과 이용하기
15. 3주차 끝 & 숙제 설명
HW. 3주차 숙제 해설



모든 토글을 열고 닫는 단축키

Windows : **Ctrl** + **alt** + **t**

Mac : **⌘** + **⌥** + **t**

01. 3주차 설치

▼ 1) 이번주 수업을 위해 설치/가입 할 것들!



아래 프로그램/파일들을 가지고 계신 기기에 맞게 차근차근 다운로드 받아주세요!

1. Python

▼ 윈도우

1. 링크를 눌러 다운로드 해주세요. ([다운로드 링크](#))

▼ [코드스니펫] 파이썬(윈도우) 다운로드

<https://www.python.org/ftp/python/3.8.6/python-3.8.6-amd64.exe>



반드시 Python 3.8.6 버전을 설치해주세요.

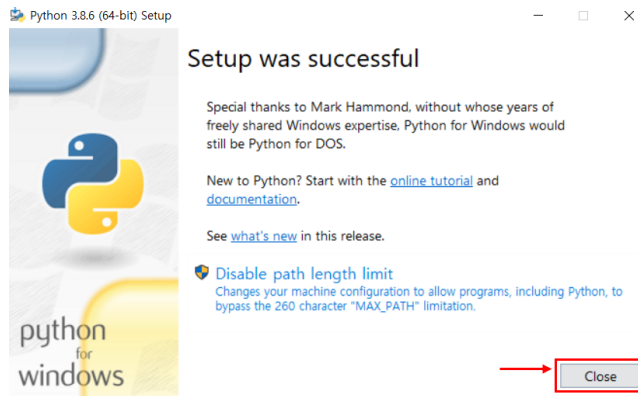
(3.9는 너무 최신 버전이라 오작동 하는 라이브러리들이 있어요)

2. 다운로드된 파이썬 파일을 열어 설치합니다.



주의) Add Python 3.8 to PATH 에 체크해야 합니다!!





- 설치 완료!

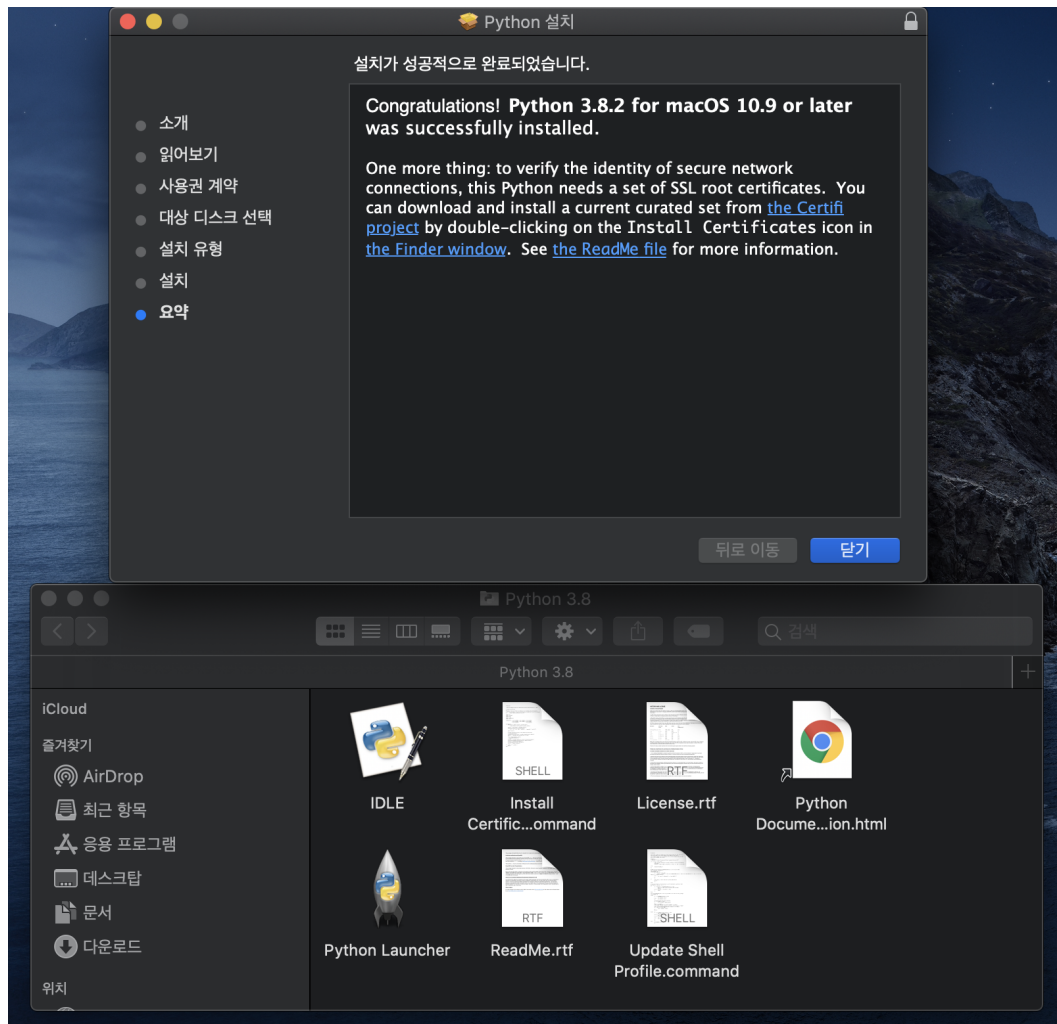
▼ 맥

🔥 Mac에는 기본적으로 파이썬 2.x 버전이 설치되어 있지만, 버전3으로 업데이트 되면서 많은 변화가 있었습니다. 본 강의에서는 3.8.2 버전의 파이썬이 필요하니 아래 가이드에 따라 꼭 설치해주세요!

▼ [코드스니펫] 파이썬(맥) 다운로드

<https://www.python.org/ftp/python/3.8.2/python-3.8.2-macosx10.9.pkg>

1. 버튼을 눌러 다운로드 해주세요. ([다운로드 링크](#))
2. 다운로드된 파이썬 파일을 열어 설치합니다. 아래와 같은 폴더가 나타나면 설치가 완료된 것입니다.



2. (원도우만!) Git bash

▼ 윈도우만! 맥은 다운받지 않아도 돼요!

▼ [코드스니펫] gitbash 다운로드

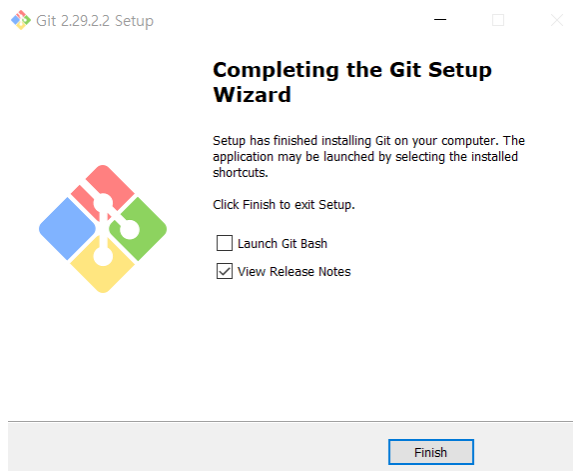
<https://git-scm.com/>

- 다운로드: <https://git-scm.com/>



클릭 후 다운로드 받은 설치 파일을 열어서, 설치를 진행합니다.

👉 모두 next → next → next .. 해서 설치하시면 아래와 같이 완료! 됩니다!

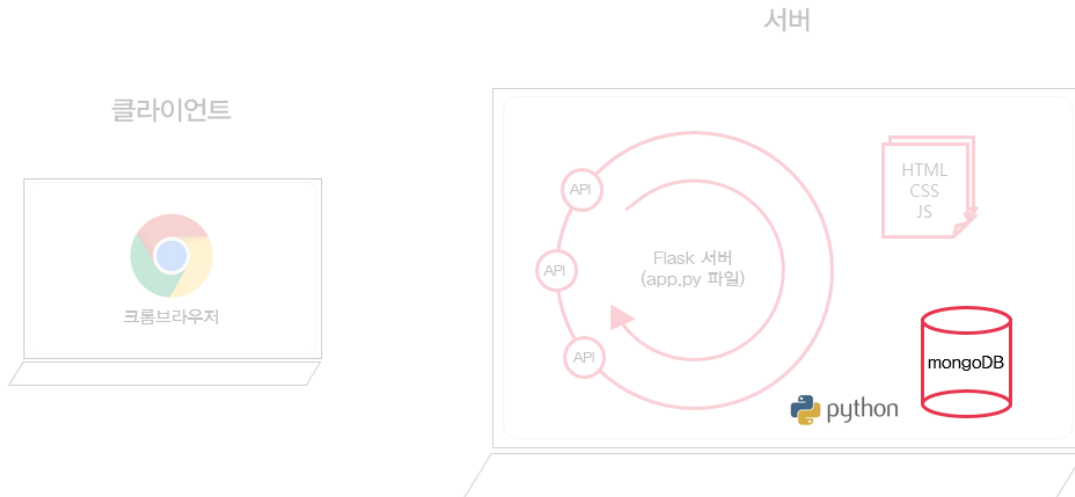


▼ 2) 오늘 배울 것 이야기 - 3주차: Python, 크롤링, mongoDB

👉 오늘은 드디어 '파이썬'을 배울거예요. 먼저 문법을 연습하고, 라이브러리를 활용하여 네이버 영화목록을 짭 가져와보겠습니다. (기대되죠!)

+

그리고, 우리의 인생 첫 데이터베이스. mongoDB를 다뤄볼게요!



02. 연습 겸 복습 - 스파르타피디아에 OpenAPI 붙여보기

🔥 2주차 리마인드를 위해 특별히 복습을 준비했어요. 한번 더 뽐내고 갑시다!

▼ 1) 스파르타가 만들어둔 OpenAPI 파악하기

▼ [코드스니펫] 스파르타피디아 API(GET)

```
http://spartacodingclub.shop/web/api/movie
```

▼ [코드스니펫] 스파르타피디아 구경하기

```
http://spartacodingclub.shop/web/movie
```

👉 네 그렇습니다!

나홀로메모장에 들어가는 **아티클들의 정보를 불러오는 OpenAPI**입니다.
이 API를 써서 저장된 포스팅 불러오기를 만들어볼게요!
(나중에 이 API를 실제로 만들어 볼 예정!)

▼ 2) 포스팅가져오기API 붙이기

1. 2주차에 완성했던 '스파르타 피디아'를 켜봅시다!

▼ [코드스니펫] 코드가 없다면!

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztCQTWfspd3yD65VohhpUuCOMLASjC" crossorigin="anonymous">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-Mrcw6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
        crossorigin="anonymous"></script>
```

```

<title>스파르타 피디아</title>

<link href="https://fonts.googleapis.com/css2?family=Gowun+Dodum&display=swap" rel="stylesheet">

<style>
* {
    font-family: 'Gowun Dodum', sans-serif;
}

.mytitle {
    width: 100%;
    height: 250px;

    background-image: linear-gradient(0deg, rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)), url('https://movie-phinf.pst

    background-position: center;
    background-size: cover;

    color: white;

    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
}

.mytitle > button {
    width: 200px;
    height: 50px;

    background-color: transparent;
    color: white;

    border-radius: 50px;
    border: 1px solid white;

    margin-top: 10px;
}

.mytitle > button:hover {
    border: 2px solid white;
}

.mycomment {
    color: gray;
}

.mycards {
    margin: 20px auto 0px auto;
    width: 95%;
    max-width: 1200px;
}

.mypost {
    width: 95%;
    max-width: 500px;
    margin: 20px auto 0px auto;
    padding: 20px;
    box-shadow: 0px 0px 3px 0px gray;

    display: none;
}

.mybtns {
    display: flex;
    flex-direction: row;
    align-items: center;
    justify-content: center;

    margin-top: 20px;
}

.mybtns > button {
    margin-right: 10px;
}
</style>
<script>
    function open_box(){
        $('#post-box').show()
    }
    function close_box(){
        $('#post-box').hide()
    }
</script>
</head>

<body>
<div class="mytitle">
    <h1>내 생애 최고의 영화들</h1>

```

```

        <button onclick="open_box()">영화 기록하기</button>
    </div>
    <div class="mypost" id="post-box">
        <div class="form-floating mb-3">
            <input id="url" type="email" class="form-control" placeholder="name@example.com">
            <label>영화URL</label>
        </div>
        <div class="input-group mb-3">
            <label class="input-group-text" for="inputGroupSelect01">별점</label>
            <select class="form-select" id="inputGroupSelect01">
                <option selected>-- 선택하기 --</option>
                <option value="1">★</option>
                <option value="2">★★</option>
                <option value="3">★★★</option>
                <option value="4">★★★★</option>
                <option value="5">★★★★★</option>
            </select>
        </div>
        <div class="form-floating">
            <textarea id="comment" class="form-control" placeholder="Leave a comment here" id="floatingTextarea2"
                style="height: 100px"></textarea>
            <label for="floatingTextarea2">코멘트</label>
        </div>
        <div class="mybtns">
            <button type="button" class="btn btn-dark">기록하기</button>
            <button onclick="close_box()" type="button" class="btn btn-outline-dark">닫기</button>
        </div>
    </div>
    <div class="mycards">
        <div class="row row-cols-1 row-cols-md-4 g-4" id="cards-box">
            <div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">영화 제목이 들어갑니다</h5>
                        <p class="card-text">여기에 영화에 대한 설명이 들어갑니다.</p>
                        <p>★★★</p>
                        <p class="mycomment">나의 한줄 평을 씁니다</p>
                    </div>
                </div>
            </div>
            <div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">영화 제목이 들어갑니다</h5>
                        <p class="card-text">여기에 영화에 대한 설명이 들어갑니다.</p>
                        <p>★★★</p>
                        <p class="mycomment">나의 한줄 평을 씁니다</p>
                    </div>
                </div>
            </div>
            <div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">영화 제목이 들어갑니다</h5>
                        <p class="card-text">여기에 영화에 대한 설명이 들어갑니다.</p>
                        <p>★★★</p>
                        <p class="mycomment">나의 한줄 평을 씁니다</p>
                    </div>
                </div>
            </div>
            <div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">영화 제목이 들어갑니다</h5>
                        <p class="card-text">여기에 영화에 대한 설명이 들어갑니다.</p>
                        <p>★★★</p>
                        <p class="mycomment">나의 한줄 평을 씁니다</p>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
</body>

</html>

```

2. 우선, 로드가 다 되면 실행되는 함수를 하나 만듭니다

 구글링해보기: javascript 로딩 후 실행

▼ [코드스니펫] 로딩 후 바로실행

```
$(document).ready(function(){
    listing();
});

function listing() {
    console.log('화면 로딩 후 잘 실행되었습니다');
}
```

3. API 결과값을 다시한번 확인하기

```
$(document).ready(function(){
    listing();
});

function listing() {
    $.ajax({
        type: "GET",
        url: "http://spartacodingclub.shop/web/api/movie",
        data: {},
        success: function(response){
            console.log(response)
        }
    })
}
```

4. 영화 데이터를 console에 찍어봅시다!

```
$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/web/api/movie",
    data: {},
    success: function(response){
        console.log(response['movies'])
    }
})
```

5. movies를 돌면서, 하나씩 출력해봅니다.

```
$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/web/api/movie",
    data: {},
    success: function(response){
        let movies = response['movies']
        for (let i = 0 ; i < movies.length; i++) {
            let movie = movies[i]
            console.log(movie)
        }
    }
})
```

6. movie 내용을 (image, comment, title, desc, star) 가지고 HTML을 만들어 붙입니다.

```
$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/web/api/movie",
    data: {},
    success: function(response){
        let movies = response['movies']
        for (let i = 0 ; i < movies.length; i++) {
            let movie = movies[i]
            let title = movie['title']
            let desc = movie['desc']
            let image = movie['image']
            let comment = movie['comment']
            let star = movie['star']
```

```

        let temp_html = `<div class="col">
            <div class="card h-100">
                
                <div class="card-body">
                    <h5 class="card-title">${title}</h5>
                    <p class="card-text">${desc}</p>
                    <p>${star}</p>
                    <p class="mycomment">${comment}</p>
                </div>
            </div>
        </div>`
        $('#cards-box').append(temp_html)
    }
}
})

```

7. 별을 그려서 붙여줍니다.

 `let star_image = '*'.repeat(star)` 을 사용하기!

```

$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/web/api/movie",
    data: {},
    success: function(response){
        let movies = response['movies']
        for (let i = 0 ; i < movies.length; i++) {
            let movie = movies[i]
            let title = movie['title']
            let desc = movie['desc']
            let image = movie['image']
            let comment = movie['comment']
            let star = movie['star']

            let star_image = '*'.repeat(star)

            let temp_html = `<div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">${title}</h5>
                        <p class="card-text">${desc}</p>
                        <p>${star_image}</p>
                        <p class="mycomment">${comment}</p>
                    </div>
                </div>
            </div>`
            $('#cards-box').append(temp_html)
        }
    }
})

```

8. 먼저 있던 카드들을 지워줍니다.

```

$('#cards-box').empty('');

```

9. 완성된 코드 전체

▼ [코드스니펫] 스파르타피디아 완성 코드

```

<!doctype html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztQCwTfwFsp3yD65VohhpuaComLASjC" crossorigin="anonymous">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-Mrcw6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
        crossorigin="anonymous"></script>

```

```

<title>스파르타 피디아</title>

<link href="https://fonts.googleapis.com/css2?family=Gowun+Dodum&display=swap" rel="stylesheet">

<style>
  * {
    font-family: 'Gowun Dodum', sans-serif;
  }

  .mytitle {
    width: 100%;
    height: 250px;

    background-image: linear-gradient(0deg, rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)), url('https://movie-phinf.pst

    background-position: center;
    background-size: cover;

    color: white;

    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
  }

  .mytitle > button {
    width: 200px;
    height: 50px;

    background-color: transparent;
    color: white;

    border-radius: 50px;
    border: 1px solid white;

    margin-top: 10px;
  }

  .mytitle > button:hover {
    border: 2px solid white;
  }

  .mycomment {
    color: gray;
  }

  .mycards {
    margin: 20px auto 0px auto;
    width: 95%;
    max-width: 1200px;
  }

  .mypost {
    width: 95%;
    max-width: 500px;
    margin: 20px auto 0px auto;
    padding: 20px;
    box-shadow: 0px 0px 3px 0px gray;

    display: none;
  }

  .mybtns {
    display: flex;
    flex-direction: row;
    align-items: center;
    justify-content: center;

    margin-top: 20px;
  }

  .mybtns > button {
    margin-right: 10px;
  }
</style>
<script>
  $(document).ready(function(){
    listing();
  });

  function listing() {
    $('#cards-box').empty()
    $.ajax({
      type: "GET",
      url: "http://spartacodingclub.shop/web/api/movie",
      data: {},
      success: function(response){

```

```

        let movies = response['movies']
        for (let i = 0 ; i < movies.length; i++) {
            let movie = movies[i]
            let title = movie['title']
            let desc = movie['desc']
            let image = movie['image']
            let comment = movie['comment']
            let star = movie['star']

            let star_image = '*'.repeat(star)

            let temp_html = `<div class="col">
                <div class="card h-100">
                    
                    <div class="card-body">
                        <h5 class="card-title">${title}</h5>
                        <p class="card-text">${desc}</p>
                        <p>${star_image}</p>
                        <p class="mycomment">${comment}</p>
                    </div>
                </div>
            </div>`
            $('#cards-box').append(temp_html)
        }
    })
}

function open_box(){
    $('#post-box').show()
}
function close_box(){
    $('#post-box').hide()
}
}
</script>
</head>

<body>
<div class="mytitle">
    <h1>내 생애 최고의 영화들</h1>
    <button onclick="open_box()">영화 기록하기</button>
</div>
<div class="mypost" id="post-box">
    <div class="form-floating mb-3">
        <input id="url" type="email" class="form-control" placeholder="name@example.com">
        <label>영화URL</label>
    </div>
    <div class="input-group mb-3">
        <label class="input-group-text" for="inputGroupSelect01">별점</label>
        <select class="form-select" id="inputGroupSelect01">
            <option selected>-- 선택하기 --</option>
            <option value="1">*</option>
            <option value="2">*</option>
            <option value="3">*</option>
            <option value="4">*</option>
            <option value="5">*</option>
        </select>
    </div>
    <div class="form-floating">
        <textarea id="comment" class="form-control" placeholder="Leave a comment here" id="floatingTextarea2"
            style="height: 100px"></textarea>
        <label for="floatingTextarea2">코멘트</label>
    </div>
    <div class="mybtms">
        <button type="button" class="btn btn-dark">기록하기</button>
        <button onclick="close_box()" type="button" class="btn btn-outline-dark">닫기</button>
    </div>
</div>
<div class="mycards">
    <div class="row row-cols-1 row-cols-md-4 g-4" id="cards-box">
    </div>
</div>
</body>
</html>

```

03. 파이썬 시작하기



바탕화면에 **sparta** 폴더 → **pythonprac** 폴더를 만들고 시작할게요!

▼ 1) 파이썬을 설치한다는 것의 의미



파이썬을 설치한다?

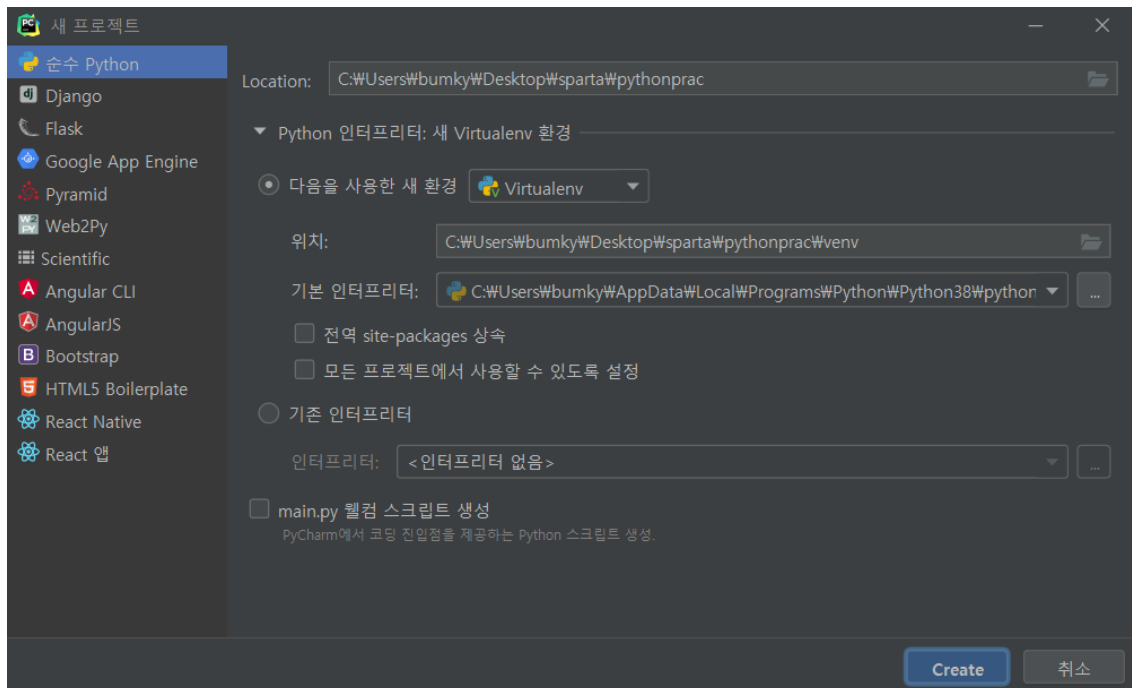
→ 일종의 번역팩을 설치한다고 생각하면 됩니다. 컴퓨터는 101010001 과 같은 언어만 알아듣는다고 했지요? 파이썬 문법으로 된 것을 101010001로 변환해줄 수 있도록, 번역 패키지를 설치하는 것입니다.

▼ 2) 프로젝트 만들기 & 파이썬 파일 실행

- 파일 → new project 를 클릭합니다. (맥은 pycharm → new project)



Location는 pythonprac으로 잡을게요!



- **hello.py** 파일 안에 다음 내용을 붙여넣고, 마우스 오른쪽 키 → '실행' 합니다!

```
print('Hello, sparta')
```

▼ 3) 파이썬 문법을 시작하기에 앞서..

- 파이썬은 매우 직관적인 언어이고, 할 수 있는 것도 많습니다. 그런데, 개발자들도 모든 문법을 기억하기란 쉽지 않습니다. 오늘 배우는 것 외에 필요한 것들은 구글링해서 찾아보면 됩니다!

04. 파이썬 기초공부

▼ 1) 파이썬 기초 문법

▼ 변수 & 기본연산

```
a = 3      # 3을 a에 넣는다
b = a      # a를 b에 넣는다
```

```

a = a + 1 # a+1을 다시 a에 넣는다

num1 = a*b # a*b의 값을 num1이라는 변수에 넣는다
num2 = 99 # 99의 값을 num2이라는 변수에 넣는다

# 변수의 이름은 마음대로 지을 수 있음!
# 진짜 "마음대로" 짓는 게 좋을까? var1, var2 이렇게?

```

▼ 자료형

- 숫자, 문자형

```

name = 'bob' # 변수에는 문자열이 들어갈 수도 있고,
num = 12 # 숫자가 들어갈 수도 있고,

is_number = True # True 또는 False -> "Boolean"형이 들어갈 수도 있습니다.

#####
# 그리고 List, Dictionary 도 들어갈 수도 있죠. 그게 원지는 아래에서!

```

- 리스트 형 (Javascript의 배열형과 동일)

```

a_list = []
a_list.append(1) # 리스트에 값을 넣는다
a_list.append([2,3]) # 리스트에 [2,3]이라는 리스트를 다시 넣는다

# a_list의 값은? [1, [2,3]]
# a_list[0]의 값은? 1
# a_list[1]의 값은? [2,3]
# a_list[1][0]의 값은? 2

```

- Dictionary 형 (Javascript의 dictionary형과 동일)

```

a_dict = {}
a_dict = {'name':'bob', 'age':21}
a_dict['height'] = 178

# a_dict의 값은? {'name':'bob', 'age':21, 'height':178}
# a_dict['name']의 값은? 'bob'
# a_dict['age']의 값은? 21
# a_dict['height']의 값은? 178

```

- Dictionary 형과 List형의 조합

```

people = [{'name':'bob', 'age':20}, {'name':'carry', 'age':38}]

# people[0]['name']의 값은? 'bob'
# people[1]['name']의 값은? 'carry'

person = {'name':'john', 'age':7}
people.append(person)

# people의 값은? [{'name':'bob', 'age':20}, {'name':'carry', 'age':38}, {'name':'john', 'age':7}]
# people[2]['name']의 값은? 'john'

```

▼ 함수

- 함수의 정의 - 이름은 마음대로 정할 수 있음!

```

# 수학문제에서
f(x) = 2*x+3
y = f(2)
y의 값은? 7

# 참고: 자바스크립트에서는
function f(x) {
  return 2*x+3
}

# 파이썬에서
def f(x):
  return 2*x+3

```

```
y = f(2)
y의 값은? 7
```

- 함수의 응용

```
def sum_all(a,b,c):
    return a+b+c

def mul(a,b):
    return a*b

result = sum_all(1,2,3) + mul(10,10)

# result라는 변수의 값은?
```

▼ 조건문

- if / else 로 구성!


```
def oddeven(num): # oddeven이라는 이름의 함수를 정의한다. num을 변수로 받는다.
    if num % 2 == 0: # num을 2로 나눈 나머지가 0이면
        return True # True (참)을 반환한다.
    else: # 아니면,
        return False # False (거짓)을 반환한다.

result = oddeven(20)
# result의 값은 무엇일까요?
```

```
def is_adult(age):
    if age > 20:
        print('성인입니다') # 조건이 참이면 성인입니다를 출력
    else:
        print('청소년이에요') # 조건이 거짓이면 청소년이에요를 출력

is_adult(30)
# 무엇이 출력될까요?
```

▼ 반복문

 파이썬에서의 반복문은, 리스트의 요소들을 하나씩 꺼내쓰는 형태입니다.

- 4줄, 무조건 리스트와 함께 씁니다!

```
fruits = ['사과', '배', '감', '귤']

for fruit in fruits:
    print(fruit)

# 사과, 배, 감, 귤 하나씩 꺼내어 찍힙니다.
```

- 살짝 응용해볼까요? - 과일 갯수 세기 함수

▼ [코드스니펫] 리스트 예제

```
fruits = ['사과', '배', '배', '감', '수박', '귤', '딸기', '사과', '배', '수박']
```

```
fruits = ['사과', '배', '배', '감', '수박', '귤', '딸기', '사과', '배', '수박']

count = 0
for fruit in fruits:
    if fruit == '사과':
        count += 1

print(count)

# 사과의 갯수를 세어 보여줍니다.
```

```
def count_fruits(target):
    count = 0
    for fruit in fruits:
        if fruit == target:
            count += 1
    return count

subak_count = count_fruits('수박')
print(subak_count) #수박의 갯수

gam_count = count_fruits('감')
print(gam_count) #감의 갯수
```

- 다른 예제를 살펴봅시다.

▼ [코드스니펫] 딕셔너리 예제

```
people = [{'name': 'bob', 'age': 20},
          {'name': 'carry', 'age': 38},
          {'name': 'john', 'age': 7},
          {'name': 'smith', 'age': 17},
          {'name': 'ben', 'age': 27}]
```

```
people = [{'name': 'bob', 'age': 20},
          {'name': 'carry', 'age': 38},
          {'name': 'john', 'age': 7},
          {'name': 'smith', 'age': 17},
          {'name': 'ben', 'age': 27}]

# 모든 사람의 이름과 나이를 출력해봅시다.
for person in people:
    print(person['name'], person['age'])

# 이번엔, 반복문과 조건문을 응용한 함수를 만들어봅시다.
# 이름을 받으면, age를 리턴해주는 함수
def get_age(myname):
    for person in people:
        if person['name'] == myname:
            return person['age']
    return '해당하는 이름이 없습니다'

print(get_age('bob'))
print(get_age('kay'))
```

05. 파이썬 패키지 설치하기

▼ 1) 파이썬 패키지(package) 설치하기



패키지? 라이브러리? →

Python 에서 패키지는 모듈(일종의 기능들 묶음)을 모아 놓은 단위입니다. 이런 패키지 의 묶음을 라이브러리 라고 볼 수 있습니다. 지금 여기서는 외부 라이브러리를 사용하기 위해서 패키지를 설치합니다.

즉, 여기서는 **패키지 설치 = 외부 라이브러리 설치!**

▼ 1. 가상 환경(virtual environment) 이란? - 프로젝트별로 패키지들을 담을 공간

**문제상황:**

회사에서는 패키지 A, B, C를 설치해서 쓰고,
개인 프로젝트에서는 패키지 B, C, D, E를 설치해서 쓰고 있었어요.

그런데 회사팀장님이 B를 이전 버전인 B'로 쓰자고 하시네요.
그렇게 되면, 같은 컴퓨터에 깔려 있는 개인 프로젝트에서는 B'로 쓰면 코드를 다 바꿔야 해요 🤖

어떻게 하면 좋을까요?

**해결책:**

다 담아둘 필요 없이 공구함을 2개 만들어서,

공구함1에 A, B', C를 담아두고,
공구함2에 B, C, D, E를 담아두고 쓰면 관리하기 편하겠죠?

그래서, 가상환경이라는 개념이 등장했습니다.
즉, **프로젝트별 공구함** 이예요.



정리하자면,

가상환경(virtual environment)은

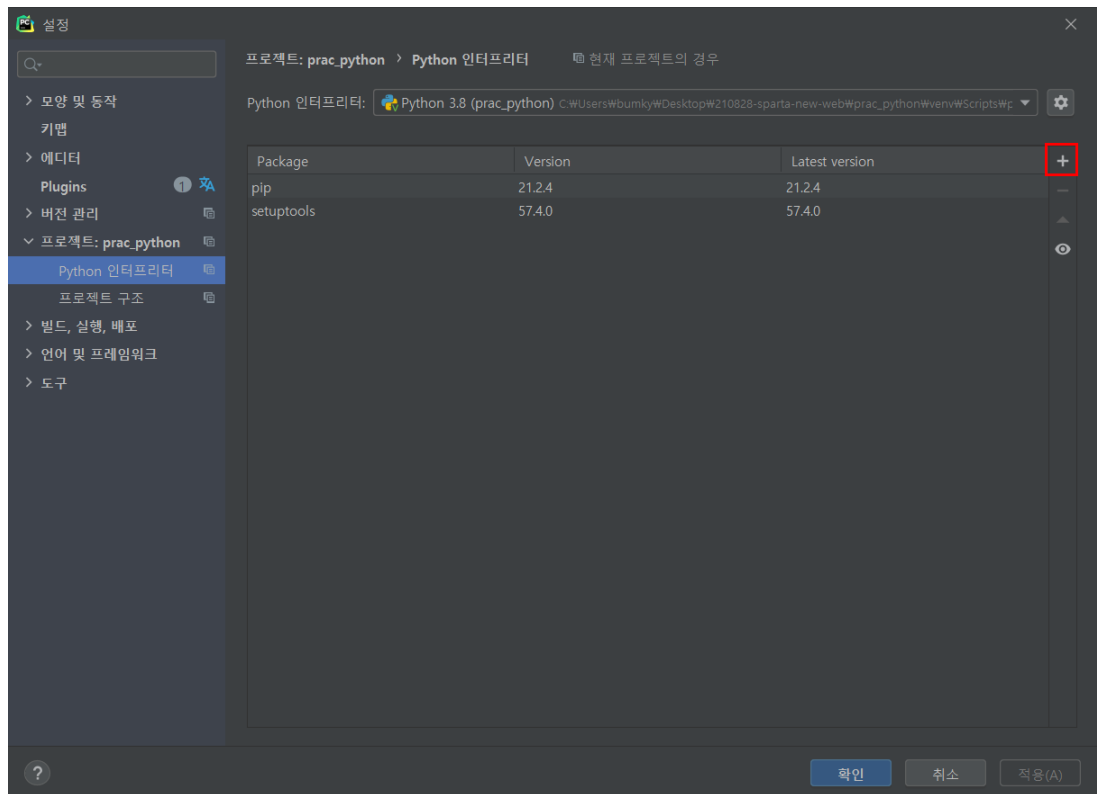
같은 시스템에서 실행되는 다른 파이썬 응용 프로그램들의 동작에 영향을 주지 않기 위해, 파이썬 배포 패키지들을 설치하거나 업그레이드하는 것을 가능하게 하는 **격리된 실행 환경** 입니다.

출처 : [파이썬 공식 용어집- 가상환경](#)

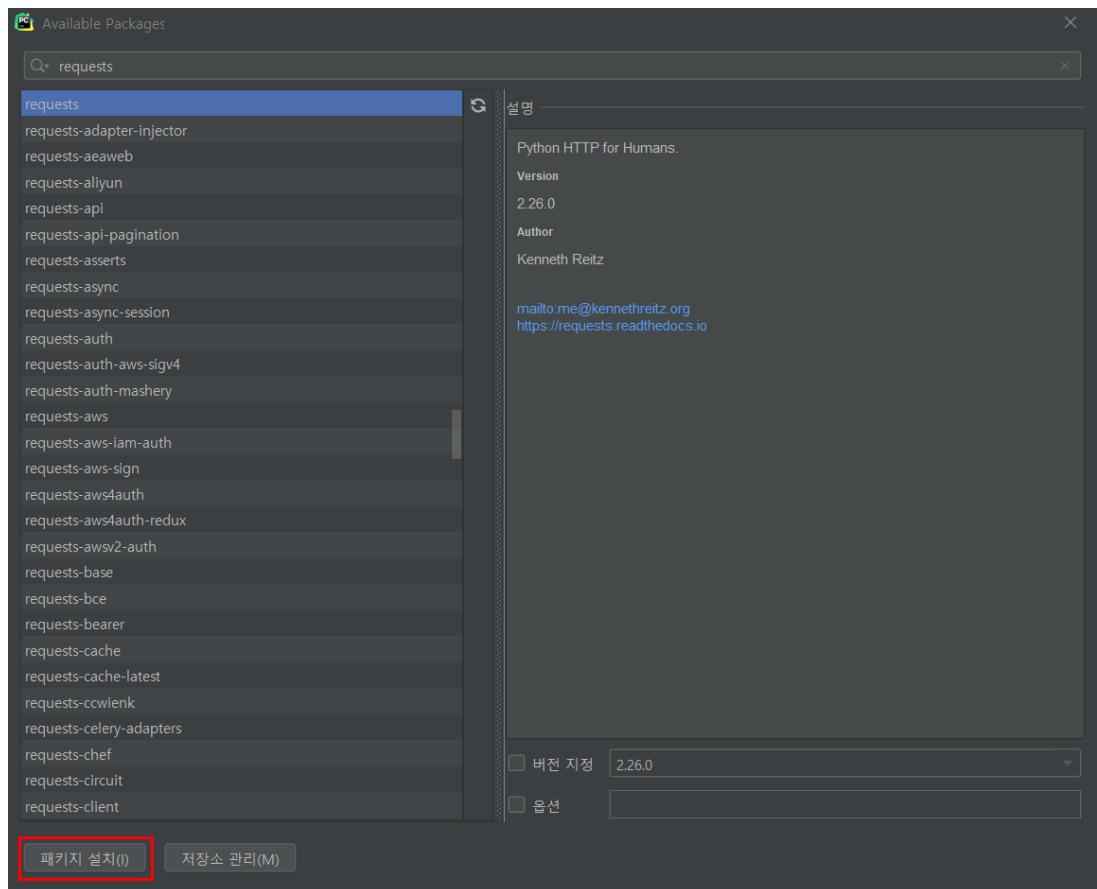
▼ 2. pip(python install package) 사용 - requests 패키지 설치해보기

앱을 설치할 때 앱스토어/플레이스토어를 가듯이, 새로운 프로젝트의 라이브러리를 가상환경(공구함)에 설치하려면 pip 를 이용하게 됩니다.

- project interpreter 화면에서 + 버튼을 누르면 아래 창이 뜹니다!



- requests를 검색하기!



06. 패키지 사용해보기

▼ 1) Requests 라이브러리 사용해보기 + List/Dictionary/함수/If/For문 연습

- 아래 방법으로 서울시 대기 OpenAPI에서, 중구의 미세먼지 값을 가져올 수 있습니다.

▼ [코드스니펫] requests 써보기

```
import requests # requests 라이브러리 설치 필요

r = requests.get('http://spartacodingclub.shop/sparta_api/seoulair')
rjson = r.json()
```

- 모든 구의 IDEX_MVL 값을 찍어주자!


```
import requests # requests 라이브러리 설치 필요

r = requests.get('http://spartacodingclub.shop/sparta_api/seoulair')
rjson = r.json()

gus = rjson['RealtimeCityAir']['row']

for gu in gus:
    print(gu['MSRSTE_NM'], gu['IDEX_MVL'])
```

- IDEX_MVL 값이 60 미만인 구만 찍어주자!

 들여쓰기가 얼마나 중요한지 다시한번 확인해보세요!

```
import requests # requests 라이브러리 설치 필요

r = requests.get('http://spartacodingclub.shop/sparta_api/seoulair')
rjson = r.json()

gus = rjson['RealtimeCityAir']['row']

for gu in gus:
    if gu['IDEX_MVL'] < 60:
        print (gu['MSRSTE_NM'], gu['IDEX_MVL'])
```

07. 웹스크래핑(크롤링) 기초

▼ 1) 웹스크래핑 해보기 (영화 제목)

- 어떤 걸 스크래핑 할 계획인가요?

▼ [코드스니펫] 네이버영화 페이지

```
https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829
```

- 패키지 추가 설치하기(beautifulsoup4)

```
bs4
```

- 크롤링 기본 세팅

▼ [코드스니펫] 크롤링 기본 세팅

```
import requests
from bs4 import BeautifulSoup

headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.36
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)
```

```
soup = BeautifulSoup(data.text, 'html.parser')

# 코딩 시작
```


```
import requests
from bs4 import BeautifulSoup


# 타겟 URL을 읽어서 HTML을 받아오고,
headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81 Safari/537.36'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만들
# soup이라는 변수에 "파싱 용이해진 html"이 담긴 상태가 됨
# 이제 코딩을 통해 필요한 부분을 추출하면 된다.
soup = BeautifulSoup(data.text, 'html.parser')

#####
# (입맛에 맞게 코딩)
#####
```

- select / select_one의 사용법을 익혀봅니다.

 영화 제목을 가져와보기!

 태그 안의 텍스트를 찍고 싶을 땐 → 태그.text
태그 안의 속성을 찍고 싶을 땐 → 태그['속성']

```
import requests
from bs4 import BeautifulSoup

# URL을 읽어서 HTML을 받아오고,
headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81 Safari/537.36'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만들
soup = BeautifulSoup(data.text, 'html.parser')

# select를 이용해서, tr들을 불러오기
movies = soup.select('#old_content > table > tbody > tr')

# movies (tr들) 의 반복문을 돌리기
for movie in movies:
    # movie 안에 a 가 있으면,
    a_tag = movie.select_one('td.title > div > a')
    if a_tag is not None:
        # a의 text를 찍어본다.
        print(a_tag.text)
```

- BeautifulSoup 내 select에 미리 정의된 다른 방법을 알아봅니다

```
# 선택자를 사용하는 방법 (copy selector)
soup.select('태그명')
soup.select('.클래스명')
soup.select('#아이디명')

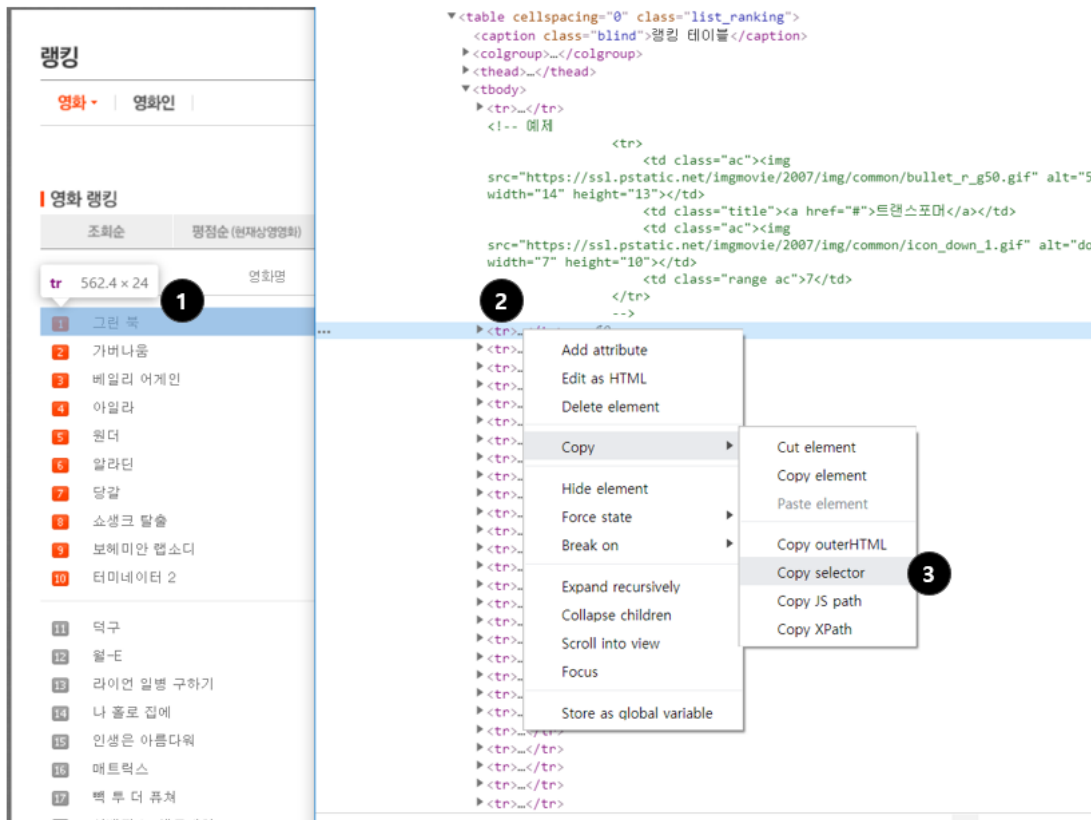
soup.select('상위태그명 > 하위태그명 > 하위태그명')
soup.select('상위태그명.클래스명 > 하위태그명.클래스명')

# 태그와 속성값으로 찾는 방법
soup.select('태그명[속성="값"]')

# 한 개만 가져오고 싶은 경우
soup.select_one('위와 동일')
```

- 항상 정확하지는 않으나, 크롬 개발자도구를 참고할 수도 있습니다.
1. 원하는 부분에서 마우스 오른쪽 클릭 → 검사

- 원하는 태그에서 마우스 오른쪽 클릭
- Copy → Copy selector로 선택자를 복사할 수 있음



08. Quiz_웹스크래핑(크롤링) 연습

▼ 1) 🐞 웹스크래핑 더 해보기 (순위, 제목, 별점)

▼ Q. 아래와 같이 보이면 완성!



▼ A. 완성 코드

```
import requests
from bs4 import BeautifulSoup

headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

soup = BeautifulSoup(data.text, 'html.parser')

#old_content > table > tbody > tr:nth-child(3) > td.title > div > a
#old_content > table > tbody > tr:nth-child(4) > td.title > div > a

movies = soup.select('#old_content > table > tbody > tr')

for movie in movies:
    a = movie.select_one('td.title > div > a')
    if a is not None:
        title = a.text
        rank = movie.select_one('td:nth-child(1) > img')['alt']
        star = movie.select_one('td.point').text
        print(rank, title, star)
```

09. DB개괄

▼ 1) 들어가기 전에 - DB는 왜 쓰는 것일까?

👉 (뜬금) 우리가 방 정리를 하는 이유는 무엇일까요?

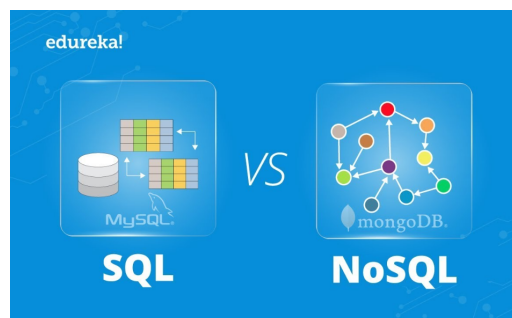
- 1번: 잘 넣어두기 위해서 / 2번: 나중에 잘 찾기 위해서

👉 (뜬금2) 한가지 더! 교보문고에 가서 책을 찾는다고 하면?

- 꽃혀진 방법대로 찾아야 쉽게 찾을 수 있겠죠! 😊 (섹션 → 출판사 → 책 제목)
- 우리 눈에 보이지 않지만, 사실 DB에는 **Index** 라는 순서로 데이터들이 정렬되어 있습니다!

▼ 2) 들어가기 전에 - DB의 두 가지 종류

👉 Database에는, 크게 두 가지 종류가 있습니다.



👉 RDBMS(SQL)

행/열의 생김새가 정해진 엑셀에 데이터를 저장하는 것과 유사합니다. 데이터 50만 개가 적재된 상태에서, 갑자기 중간에 열을 하나 더하기는 어려울 것입니다. 그러나, 정형화되어 있는 만큼, 데이터의 일관성이나 / 분석에 용이할 수 있습니다.

ex) MS-SQL, My-SQL 등

☞ No-SQL

딕셔너리 형태로 데이터를 저장해두는 DB입니다. 고로 데이터 하나 하나 마다 같은 값들을 가질 필요가 없게 됩니다. 자유로운 형태의 데이터 적재에 유리한 대신, 일관성이 부족할 수 있습니다.

ex) MongoDB

▼ 3) 들어가기 전에 - DB의 실체에 관하여

☞ 자, 그럼 DB의 실체는 무엇일까요? 특별한 컴퓨터일까요?

- 아닙니다! 아주 간단하게, 우리가 쓰는 프로그램과 같은 것이랍니다.
- 즉, 내 컴퓨터에 게임도 설치하고, PPT도 설치하고, DB도 설치할 수 있는 것이죠.

☞ 그 런 데! 이 마저도 요새는 Cloud 형태로 제공해주는 곳들이 많답니다.

- 유저가 몰리거나 / DB를 백업해야 하거나 / 모니터링 하기가 아주 용이하기 때문이죠!
(꿀팁 - 요새 트렌드는 **클.라.우.드** !)
- 그래서, 우리도 최신 클라우드 서비스인 **mongoDB Atlas** 를 사용해 볼 것입니다!



10. mongoDB 시작하기

▼ 1) mongoDB - Atlas 가입하기

☞ 아래 순서에 따라 차근 차근 따라해주세요

1. 가입하기

- 구글로 로그인하고 → Accept Privacy ... Service 에 체크 → Submit 합니다.

▼ [코드스니펫] 회원가입하기

```
https://account.mongodb.com/account/register
```

2. 다음 화면 체크하고 넘어가기

MONGODB ATLAS

Let's get your account set up

Name your organization and project

Organization
Your organization can be a business, team, or an individual

Sparta Coding Club's Org - 2021-08-29

Project Name
Use projects to isolate different environments (development/testing/production)

Project 0

What is your preferred language?

We'll use this to customize code samples and content we share with you. You can always change this later.

JavaScript

C++

C# / .NET

Go

Java

C

Perl

PHP

Python

Ruby

Scala

Other

Skip **Continue**

3. Shared를 클릭하고 넘어가기

MONGODB ATLAS

Deploy a cloud database

Experience the best of MongoDB on AWS, Azure, and Google Cloud. Choose a deployment option to get started.

PREVIEW

Serverless

For serverless applications that aren't critical with variable traffic. Minimal configuration required.

- ✓ Pay only for the operations you run
- ✓ Resources scale seamlessly to meet your workload
- ✓ Always-on security and backups

Create

Starting at
\$0.30/1M reads

Dedicated

For production applications with sophisticated workload requirements. Advanced configuration controls.

- ✓ Network isolation and fine-grained access controls
- ✓ On-demand performance advice
- ✓ Multi-region and multi-cloud options available

Create

Starting at
\$0.08/hr*
*estimated cost \$6.94/month

FREE

Shared

For learning and exploring MongoDB in a cloud environment. Basic configuration options.

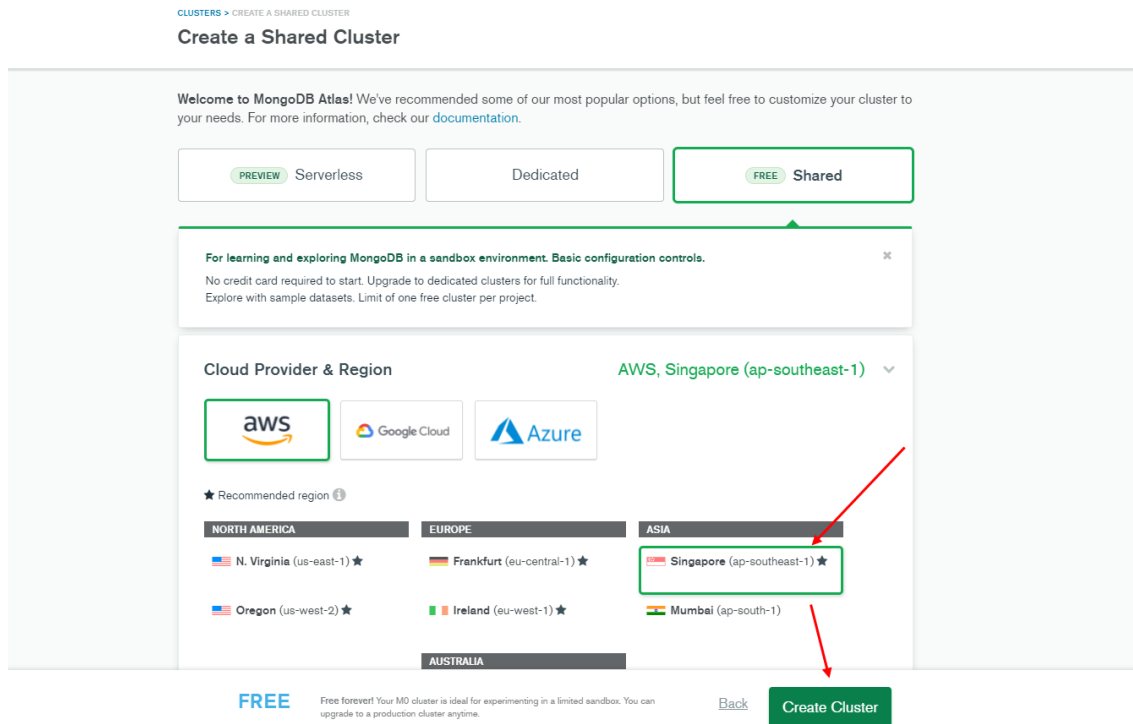
- ✓ No credit card required to start
- ✓ Explore with sample datasets
- ✓ Upgrade to dedicated clusters for full functionality

Create

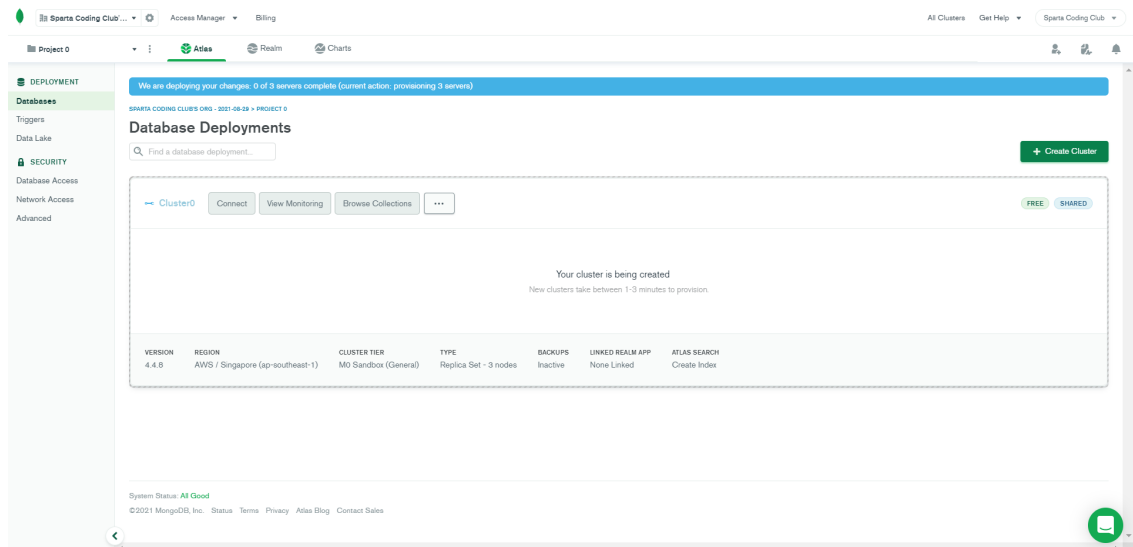
Starting at
FREE

[Dismiss](#) [Advanced Configuration Options](#)

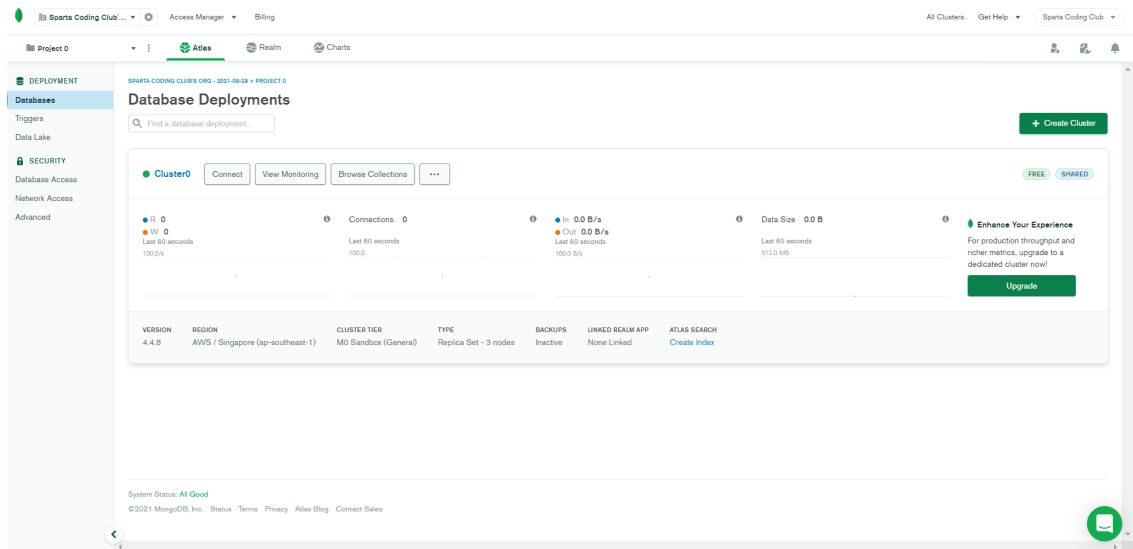
4. **싱가포르** 로 체크하고, **Create Cluster** 클릭하기



5. 아래와 같은 화면이 잠시 동안 나온 뒤에

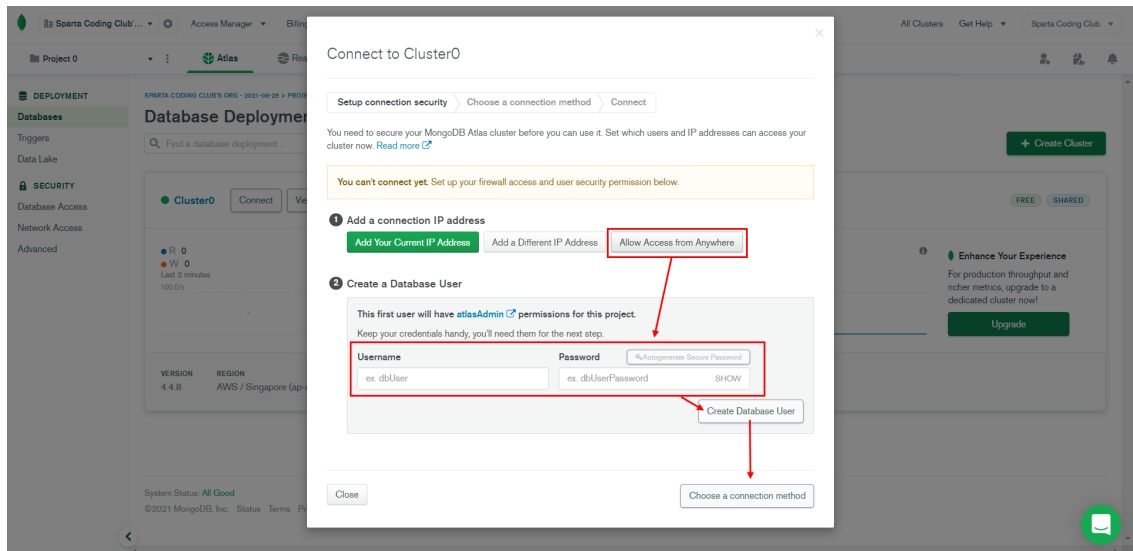
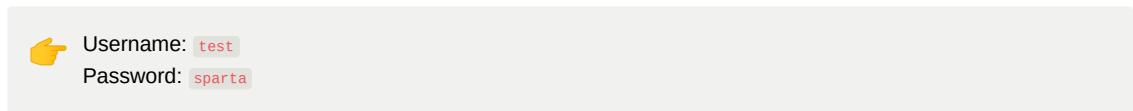


6. 새로고침 후 최종 아래와 같은 화면을 만나면 끝!



7. 연결 준비하기

- 1) Allow Access from Anywhere 클릭 → Add IP address 클릭
- 2) Username, Password를 아래와 같이 입력 → Create Database User 클릭
- 3) Choose a connection method 클릭



11. mongoDB 연결하기

- ▼ 1) mongoDB - Atlas 연결하기

👉 pymongo 라이브러리의 역할

예를 들어, MS Excel를 파이썬으로 조작하려면,
특별한 라이브러리가 필요하지 않겠어요?

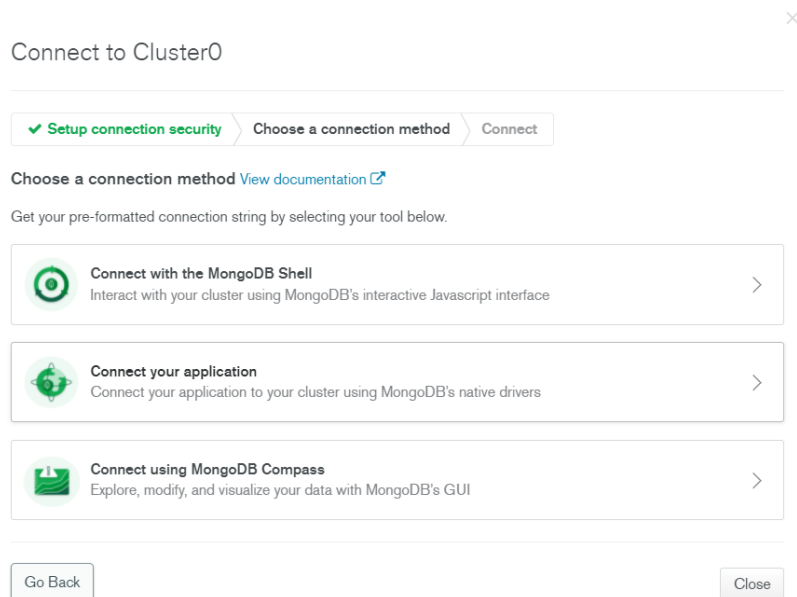
마찬가지로, mongoDB 라는 프로그램을 조작하려면,
특별한 라이브러리, pymongo가 필요합니다!

1. 패키지 설치하기

```
pymongo, dnspython
```

2. 다시, mongoDB Atlas 화면에서 Connect your application 클릭

▼ 화면 보기



Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

1 Select your driver and version

DRIVER

Python

VERSION

3.6 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://test:<password>@cluster0.55vah.mongodb.net/myFirstDatabase?
retryWrites=true&w=majority
```

Replace **<password>** with the password for the **test** user. Replace **myFirstDatabase** with the name of the database that connections will use by default. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

3. pymongo로 조작하기

▼ [코드스니펫] pymongo 기본 코드

```
from pymongo import MongoClient
client = MongoClient('여기에 URL 입력')
db = client.dbsparta
```

4. 잘 연결됐는지 테스트해보기

- 아래와 같이 입력! (데이터 넣기. 곧 배워요!)

```
doc = {
    'name': 'bob',
    'age': 27
}

db.users.insert_one(doc)
```

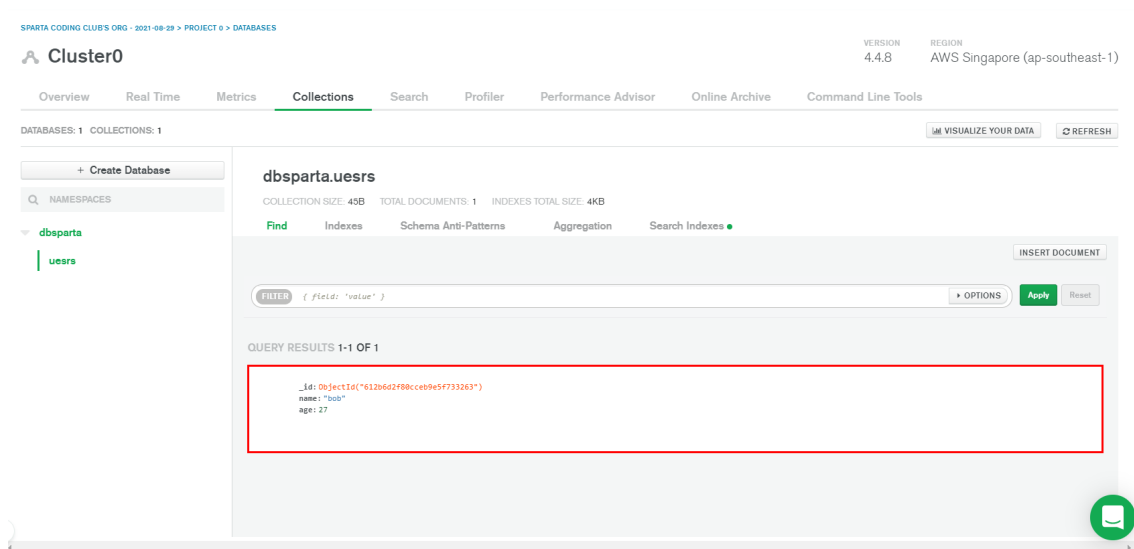
5. 다시 Cluster0의 Collections를 확인하면! 데이터가 잘 들어온 것을 확인 할 수 있습니다!



이제 한번 연결을 했으니, 복잡한 과정 없이 세 줄만 복사해서 쓰면 되겠네요!



Collections는 '즐거찾기' 해두면 더 편하겠죠?



12. pymongo로 DB조작하기

▼ 1) pymongo로 mongoDB 조작하기

- DB연결하기 & 데이터 넣기

```
# 'users'라는 collection에 {'name':'bobby', 'age':21}를 넣습니다.
db.users.insert_one({'name':'bobby', 'age':21})
db.users.insert_one({'name':'kay', 'age':27})
db.users.insert_one({'name':'john', 'age':30})
```

- 모든 결과 값을 보기

▼ [코드스니펫] pymongo(find)

```
same_ages = list(db.users.find({}, {'_id':False}))
```

```
# 모든 데이터 뽑아보기
all_users = list(db.users.find({}, {'_id':False}))

print(all_users[0])          # 0번째 결과값을 보기
print(all_users[0]['name'])  # 0번째 결과값의 'name'을 보기

for user in all_users:      # 반복문을 돌며 모든 결과값을 보기
    print(user)
```

- 특정 결과 값을 뽑아 보기

▼ [코드스니펫] pymongo(find_one)

```
user = db.users.find_one({'name':'bobby'})
```

```
user = db.users.find_one({'name':'bobby'})
print(user)
```

- 수정하기

▼ [코드스니펫] pymongo(update_one)

```
db.users.update_one({'name':'bobby'}, {'$set':{'age':19}})
```

```
# 오타가 많으니 이 줄을 복사해서 씁시다!
db.users.update_one({'name':'bobby'}, {'$set':{'age':19}})

user = db.users.find_one({'name':'bobby'})
print(user)
```

- 삭제하기 (거의 안 씀)

▼ [코드스니펫] pymongo(delete_one)

```
db.users.delete_one({'name':'bobby'})

db.users.delete_one({'name':'bobby'})

user = db.users.find_one({'name':'bobby'})
print(user)
```

▼ 2) pymongo 사용법. 코드요약

▼ [코드스니펫] pymongo 코드 요약

```
# 저장 - 예시
doc = {'name':'bobby', 'age':21}
db.users.insert_one(doc)

# 한 개 찾기 - 예시
user = db.users.find_one({'name':'bobby'})

# 여러개 찾기 - 예시 ( _id 값은 제외하고 출력)
all_users = list(db.users.find({}, {'_id':False}))

# 바꾸기 - 예시
db.users.update_one({'name':'bobby'}, {'$set':{'age':19}})

# 지우기 - 예시
db.users.delete_one({'name':'bobby'})
```



우리는 딱 네 가지 기능만 알면 됩니다. 저장하고, 찾고, 바꾸고, 지우고!
이 기능들을 어떻게 사용하는지 요약하면 다음과 같습니다.

dbtest 파일에 코드스니펫의 내용을 복사해들게요!

```
# 저장 - 예시
doc = {'name':'bobby', 'age':21}
db.users.insert_one(doc)

# 한 개 찾기 - 예시
user = db.users.find_one({'name':'bobby'})

# 여러개 찾기 - 예시 ( _id 값은 제외하고 출력)
all_users = list(db.users.find({}, {'_id':False}))

# 바꾸기 - 예시
db.users.update_one({'name':'bobby'}, {'$set':{'age':19}})

# 지우기 - 예시
db.users.delete_one({'name':'bobby'})
```

13. 웹스크래핑 결과 저장하기

▼ 1) insert 연습하기 - 웹스크래핑 결과를 DB에 저장하기

- 이 코드에서 시작해봅시다!

▼ [코드스니펫] 크롤링 완성코드

```

import requests
from bs4 import BeautifulSoup

# URL을 읽어서 HTML을 받아오고,
headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만들
soup = BeautifulSoup(data.text, 'html.parser')

# select를 이용해서, tr들을 불러오기
movies = soup.select('#old_content > table > tbody > tr')

# movies (tr들) 의 반복문을 돌리기
for movie in movies:
    # movie 안에 a 가 있으면,
    a_tag = movie.select_one('td.title > div > a')
    if a_tag is not None:
        rank = movie.select_one('td:nth-child(1) > img')['alt'] # img 태그의 alt 속성값을 가져오기
        title = a_tag.text # a 태그 사이의 텍스트를 가져오기
        star = movie.select_one('td.point').text # td 태그 사이의 텍스트를 가져오기
        print(rank, title, star)

```

- pymongo 기본 세팅

```

import requests
from bs4 import BeautifulSoup

from pymongo import MongoClient
client = MongoClient('mongodb+srv://test:sparta@cluster0.55vah.mongodb.net/Cluster0?retryWrites=true&w=majority')
db = client.dbsparta

# URL을 읽어서 HTML을 받아오고,
headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만들
soup = BeautifulSoup(data.text, 'html.parser')

# select를 이용해서, tr들을 불러오기
movies = soup.select('#old_content > table > tbody > tr')

# movies (tr들) 의 반복문을 돌리기
for movie in movies:
    # movie 안에 a 가 있으면,
    a_tag = movie.select_one('td.title > div > a')
    if a_tag is not None:
        rank = movie.select_one('td:nth-child(1) > img')['alt'] # img 태그의 alt 속성값을 가져오기
        title = a_tag.text # a 태그 사이의 텍스트를 가져오기
        star = movie.select_one('td.point').text # td 태그 사이의 텍스트를 가져오기
        print(rank, title, star)

```

- 도큐먼트 만들어 하나씩 insert 하기

```

import requests
from bs4 import BeautifulSoup

from pymongo import MongoClient
client = MongoClient('mongodb+srv://test:sparta@cluster0.55vah.mongodb.net/Cluster0?retryWrites=true&w=majority')
db = client.dbsparta

# URL을 읽어서 HTML을 받아오고,
headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.81'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.naver?sel=pnt&date=20210829', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만들
soup = BeautifulSoup(data.text, 'html.parser')

# select를 이용해서, tr들을 불러오기
movies = soup.select('#old_content > table > tbody > tr')

# movies (tr들) 의 반복문을 돌리기
for movie in movies:
    # movie 안에 a 가 있으면,
    a_tag = movie.select_one('td.title > div > a')
    if a_tag is not None:
        rank = movie.select_one('td:nth-child(1) > img')['alt'] # img 태그의 alt 속성값을 가져오기
        title = a_tag.text # a 태그 사이의 텍스트를 가져오기
        star = movie.select_one('td.point').text # td 태그 사이의 텍스트를 가져오기
        doc = {

```

```

        'rank': rank,
        'title': title,
        'star': star
    }
    db.movies.insert_one(doc)

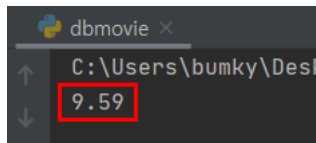
```

14. Quiz_웹스크래핑 결과 이용하기

▼ 1) 🚩 find, update 연습하기 (delete는 연습 안할게요!)

- 새 파이썬 파일 `dbmovie.py` 을 하나 만들어 연습해봅시다.
- (1) 영화제목 '가버나움'의 평점을 가져오기

▼ Q. 이렇게 되면 완성



▼ A. 완성 코드

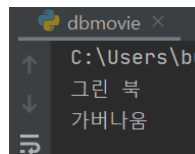
```

target_movie = db.movies.find_one({'title': '가버나움'})
print(target_movie['star'])

```

- (2) '가버나움'의 평점과 같은 평점의 영화 제목들을 가져오기

▼ Q. 이렇게 되면 완성



▼ A. 완성 코드

```

target_movie = db.movies.find_one({'title': '가버나움'})
target_star = target_movie['star']

movies = list(db.movies.find({'star': target_star}))

for movie in movies:
    print(movie['title'])

```

- (3) '가버나움' 영화의 평점을 0으로 만들기

▼ Q. 이렇게 되면 완성


```
_id: ObjectId("612b70902fd273aa0e3c5970")
rank: "02"
title: "그린 북"
star: "9.59"
```

```
_id: ObjectId("612b70912fd273aa0e3c5971")
rank: "03"
title: "가버나움"
star: "0"
```

```
_id: ObjectId("612b70912fd273aa0e3c5972")
rank: "04"
title: "디지털 아트벤처 라스트 에볼루션 : 인연"
star: "9.53"
```

▼ A. 완성 코드

```
db.movies.update_one({'title': '가버나움'}, {'$set': {'star': '0'}})
```

15. 3주차 끝 & 숙제 설명



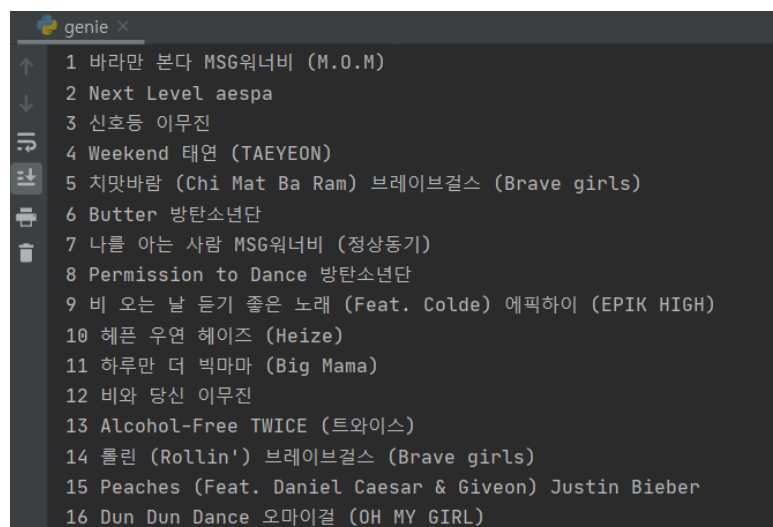
지니뮤직의 1~50위 곡을 스크래핑 해보세요.

▼ [코드스니펫] 지니뮤직 사이트

```
https://www.genie.co.kr/chart/top200?ditc=M&rtm=N&ymd=20210701
```

- 순위 / 곡 제목 / 가수를 스크래핑 하면 됩니다.

▼ Q. 이렇게 되면 완성





힌트:

0) 출력 할 때는 `print(rank, title, artist)` 하면 됩니다!

1) 앞에서 두 글자만 끊기! `text[0:2]` 를 써보세요!

2) 순위와 곡제목이 깔끔하게 나오지 않을 거예요. 옆에 여백이 있다면가, 다른 글씨도 나온다면가.. 파이썬 내장 함수인 `strip()` 을 잘 연구해보세요!

HW. 3주차 숙제 해설

▼ [코드스니펫] 3주차 숙제 답안 코드

```
import requests
from bs4 import BeautifulSoup

headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Saf
data = requests.get('https://www.genie.co.kr/chart/top200?ditc=M&rtm=N&ynd=20210701',headers=headers)

soup = BeautifulSoup(data.text, 'html.parser')

trs = soup.select('#body-content > div.newest-list > div > table > tbody > tr')

for tr in trs:
    title = tr.select_one('td.info > a.title.ellipsis').text.strip()
    rank = tr.select_one('td.number').text[0:2].strip()
    artist = tr.select_one('td.info > a.artist.ellipsis').text
    print(rank, title, artist)
```

◀ 이전 주차

다음 주차 ▶

Copyright © TeamSparta All rights reserved.