



스파르타코딩클럽 3주차



매 주차 강의자료 시작에 PDF파일과 영상 링크를 올려두었어요!

▼ PDF 강의자료 다운받기

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/6b47f3c0-25d4-4a42-8c37-a2a6dc72a3ab/week03.pdf>

▼ 영상강의 참고하기

- [3주차 복습용 영상강의 링크](#)



모든 토글을 열고 닫는 단축키

Windows : `Ctrl + alt + t`

Mac : `⌘ + option + t`

목차



수업 목표



체크인



배우고 적용하기 - 전반부



5분 꿀팁 - 프로그래밍은 문제 해결#2_질문 잘하기

[1시간] 웹 기초 동작 원리 - 3주차: Python, 웹 스크래핑(크롤링), mongoDB

[1시간] 실전 복습 "나홀로메모장"에 OpenAPI 붙여보기

[1시간] : 파이썬(Python) 기초 문법 익히기

[1시간]: 파이썬 갖고 놀기 - "웹 스크래핑(크롤링)"



문제뱅크



배우고 적용하기 - 후반부



5분 꿀팁 - CRUD

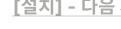
[1시간] DB 사용하기 - pymongo로 제어하고 Robo 3T로 확인하기

[0.5시간] 스크래핑 한 결과를 mongoDB에 저장하기

[0.5시간] 버전 관리 Git 사용하기



문제뱅크



소화 타임

숙제 설명

복습 도우미

[1시간] 스스로 소화 타임 시작



체크아웃

[설치] - 다음 시간을 위해 미리 설치해와야 할 것들



수업 목표

1. 파이썬(Python) 기본 문법 익히기

1) 앞으로 만들 API에서 사용하는 프로그래밍 언어인 파이썬의 기본 문법을 배우기

2. 파이썬 활용 - 웹 페이지를 스크래핑(크롤링) 하기

1) 정보 수집에 유용한 방법인 웹 스크래핑이 무엇인지 이해하기

2) 파이썬(Python)을 사용해 웹 스크래핑 하기

3. 데이터베이스 사용하기

1) pymongo를 통해 mongoDB를 제어하기

전반 3시간

✓ 체크인



튜터님은 체크인과 함께 출석 체크(링크)를 진행해주세요!

스파르타코딩클럽 출석체크

<http://spartacodingclub.shop/attendance>

▼ "15초 체크인"

- 튜터님은 타이머를 띄워주세요! ([링크](#))
- 본인의 감정상태와 오늘 있었던 소소한 일을 공유하는 시간입니다.
 - 예. 저번 숙제 재밌었어요! 내 쇼핑몰을 직접 만들어보고 발전시켜보니까 신기해요.



저번주 안내한 설치 프로그램(Python, mongoDB, Robo3T, [윈도우만]Git Bash)
설치되어 있어야 오늘 실습을 제대로 따라할 수 있습니다! 꼭 확인해주세요!

💡 배우고 적용하기 - 전반부

💡 5분 꿀팁 - 프로그래밍은 문제 해결#2_질문 잘하기

▼ 질문 잘하기

- 세상에 바보같은 질문은 없습니다! 하지만 좀 더 잘 질문할 수는 있죠!
- '프로그래밍은 문제를 해결해나가는 과정'이라고 한거 기억나시나요? 모르거나 막히는 부분이 있을 때 여러 가지 방법을 써서 문제를 해결해나가면 됩니다. 개발자들이 문제 해결을 위해 많이 쓰는 방법인 [구글링\(Googling\)](#), [질문 잘하기](#), [디버깅\(에러 해결해나가기\)](#) 중 질문 잘하기에 대한 팁을 드리겠습니다.
- ❗ 질문 그냥 하면 되는 거 아닌가요? 제가 아무렇게나 이야기해도 잘 알아들어야죠!
 - 질문을 기술적인 개념을 설명하고 커뮤니케이션하는 연습 기회로 삼아보세요.
 - 개발은 협력해야하는 것들이 많습니다. 혼자 처음부터 끝까지 만드는 경우는 매우 드물어요. (우리도 부트스트랩 같은 외부 자원을 가져다 사용하죠)
 - 어떻게 질문하느냐에 따라 심지어 답변을 못 받기도 해요. 사람들이 이해할 수 없는 질문에 답변하긴 어렵겠죠.
- 질문 잘하기! 네 가지만 기억하세요!

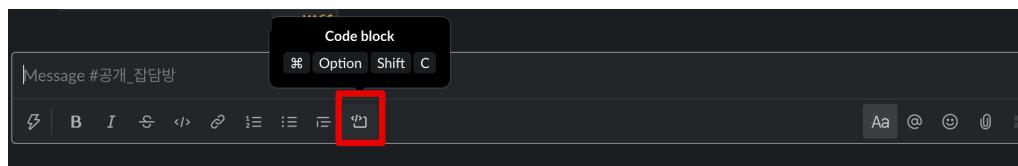
▼ 1. 내가 해결하고 싶은 문제를 정확히 알려주기

- 여러분들이 고민하고 있는 코드를 올려주세요. 그리고 에러가 나는 라인수(예. 180번째줄)을 정확히 알려주세요.
- 정확히 이 부분이라고 아는게 아니면, 되도록 설정과 관련된 부분과 앞 뒤 관련된 부분을 모두 올려주셔야합니다! 문제의 종류는 다양합니다. 앞 부분 때문에 발생하는 에러도 있고 import 에러도 있고요. 부분만 봐서는 파악하기 어려운 경우가 많아요.
 - 수업 관련 질문은 파일 통째로 올려주기 😊
- 에러 메시지 원본 그대로
 - 에러 메시지는 스크린캡쳐보다 텍스트 메시지로 올리는게 좋아요. (화면 모양과 같이 이미지로 봐야하는 건 제외)
 - [5분 꿀팁 - 구글링](#) 편에서 에러메시지로 구글링 하는 거 배웠었죠? 이미지 파일이면 복사-붙여넣기로 검색하기가 어렵겠죠?

▼ 참고. 슬랙 메시지에서 코드 붙여넣는 법

1. 짧은 코드일 경우

- '(백틱)으로 감싸기 예. '짧은 코드'
- 2. 긴 코드일 경우
 - ```(백틱 세 개)로 감싸거나
 - 메시지창 버튼 사용하기



▼ 2. 지금 내가 무엇을 알고 있는지 알려주기

- 지금까지 무엇을 시도했고, 시도하면서 알게 된 것은 무엇인지 함께 알려주는 것이 좋습니다. 이것도 문제를 해결하기 위한 실마리가 될 수 있어요.
- 현재 수준 알려주기
 - 개발 커뮤니티처럼 사람들이 여러분의 정보를 모를 때에는 '제가 python 함수까지 배웠습니다.'라고 정보를 알려주면 여러분의 현재 수준을 고려해서 답변을 하기 쉬워요.
- 구글링해서 찾고 이해한 링크
 - 앗, 설마 구글링으로 찾아보지도 않고 그냥 바로 질문하는 건 아니겠죠? **개발할 때는 스스로 문제를 해결하기 위해 고민하면서 경험치가 많이 쌓입니다. (a.k.a 삽질)**

▼ 3. 어디에 어떤 시간에 질문할지

- 질문은 되도록 공개적으로 하는 걸 추천드립니다. 1:1 질문에 대한 지식은 두 명에게만 남지만, 공개적으로 하는 질문에 대한 지식은 다수에게 남겠죠.
- 사람들이 많이 질문하는 곳을 몇 군데 알려드릴게요.
 - 스파르타코딩클럽 슬랙채널 🔥
 - stackoverflow([링크](#)) : 개발 QnA 사이트
 - 페이스북 /슬랙 개발자 커뮤니티
 - Javascript 커뮤니티에서 Python 질문하면 사람들이 어랏? 하겠죠?
 - 기술별(Javascript, Python,...)로, 직군별(프론트엔드, 백엔드, 인프라,...)로 커뮤니티가 세분화되어 있기도 합니다.
 - 기술과 관련된 Github repository 의 issue ([링크](#))
 - 나만 겪는 게 아닌 거 같은 버그일 때 여기에 버그알림(bug report)를 해주시면 좋겠죠?
 - repository 마다 분위기가 조금씩 다르니 기존에 있었던 issue를 보면서 파악해보세요

▼ 4. 질문 해결방법에 대해 피드백하기

- 시간을 들여 답변을 해준 사람에게 [감사합니다!](#) 인사 한마디 남기는 거 너무 당연하겠죠?
- 답변받은 방법대로 고쳐졌는지, 그렇지 않은지도 정보가 될 수 있어요. 여러분과 같은 고민을 하고 있는 사람이 이 방법대로 하면 해결되는지, 안되는지 알 수 있겠죠?

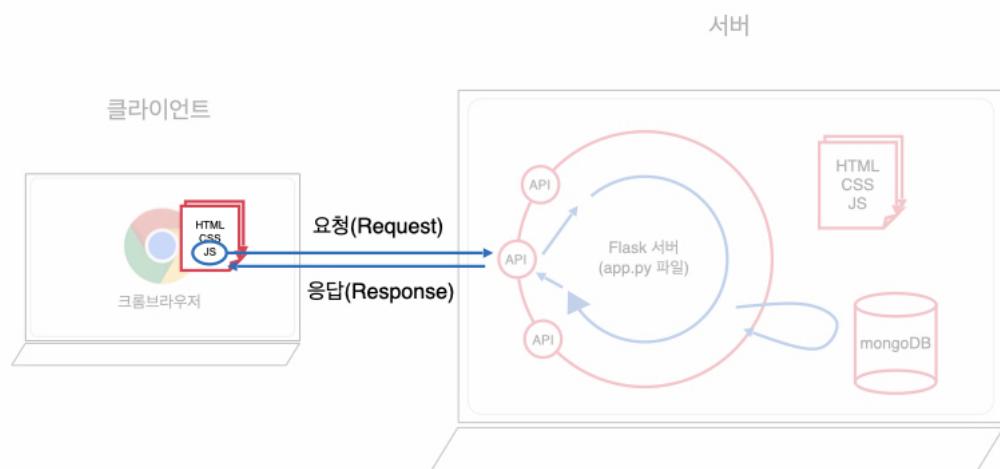
한 걸음 더! 🚶

- 스파르타코딩클럽 슬랙에 여러분들이 알고 있는 내용 질문이 올라온다면, 댓글로 답변해보세요. Learning By Teaching 가르쳐주면서 배우기!
- [질문하는 법](#), [how to ask](#) 키워드로 구글링해보기

WEB 기초 동작 원리 - 3주차: Python, 웹 스크래핑(크롤링), mongoDB

▼ 지난주 복습 - 웹 기초 동작원리 & 2주차 개념

- HTTP: 웹은 HTTP라는 규약(규칙)을 따릅니다. url에서 `http://` 가 바로 HTTP 라는 규약을 따른다는 표시예요.
- 클라이언트 : HTTP에서 요청을 하는 쪽
- 서버 : HTTP에서 요청을 받아 응답하는 쪽
- Ajax를 이용해 Javascript로 서버에 데이터를 요청하고, 서버가 응답한 데이터를 받아 조작하는 방법을 배웠습니다. 이 때 사용한 것이 API였습니다.
- API를 사용할 땐, 미리 정해둔 약속을 따라야 작동합니다. 약속들은 API 페이지(문서)에 적혀있습니다.
 - 지난 수업시간에 사용한 API 페이지 ([서울시 권역별 실시간 대기환경 현황](#) / [서울시 공공자전거 실시간 대여정보](#) / [랜덤 고양이 사진 API Doc](#))
- JSON : 데이터 표현방식. API로 요청하면 JSON형태로 데이터를 전달해줍니다. Key:Value로 이루어져 있습니다. 자료형 Dictionary와 아주- 유사하죠.



우리가 앞으로 만들 API에서 사용하는 프로그래밍 언어인 파이썬(Python)을 배울 겁니다.
먼저 문법을 연습하고, 라이브러리를 활용하여 네이버 영화목록을 가져와보겠습니다. (기대되죠!)

그리고, 우리의 인생 첫 데이터베이스. mongoDB를 다뤄볼게요!
(API를 만드는 건 다음주에!)



[1시간] 실전 복습 "나홀로메모장"에 OpenAPI 붙여보기



2주차 복습! API 사용하는거 정말 중요하니까, 한 번 더 뽀개고 갑시다!

▼ 1) API 분석 - 스파르타가 만들어 둔 OpenAPI 파악하기



API주소(GET 요청) → <http://spartacodingclub.shop/post>

어떤 기능을 수행하는 API일까요?

완성본을 다시 보고, 생각해봅시다! ([링크](#)).



바로

나홀로 메모장에 들어가는 아티클들의 정보를 불러오는 OpenAPI입니다.

서버에서 JSON 형식으로 아티클 정보(articles)를 응답(response) 데이터로 보내주죠!

이 API를 써서 '저장된 포스팅 불러오기' 기능을 만들어볼게요!

(나중에 이 API를 우리가 직접 만들어 볼겁니다)

▼ [리마인드] 응답(response) 결과 데이터 JSON 분석하는 방법

- chrome 브라우저에서 JSONView 설치하셨죠? (아직이라면 [링크](#) 들어가 설치!)

```
{  
  articles: [  
    {  
      comment: "안녕하세요. 케이론입니다! 오늘 리뷰할 영화는 그린 북입니다. 지금히 개인적인 몇 줄 평<그린 북>은 2018 토론토국제영화제 관객상 수상과 다가오는  
      desc: "1962년 미국, 입담과 주의만 믿고 살아가던 토니 발레룡가(비고 모滕스)는 교양과 우아함 그 자체인 천재...",  
      image: "https://movie-phinf.pstatic.net/20190115_228/1547528180168jgEP7_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "그린 북",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=171539"  
    },  
    {  
      comment: "문득 이런 생각이 들 때가 있습니다. 내가 이 나라가 아닌 다른 나라에서 다른 인종으로 태어났다면 어떤 삶을 살고 있을까. 이 나라에서 태어나  
      desc: "나를 세상에 태어나게 한 '부모님을 고소하고 싶어요...' 출생기록조차 없이 살아온 어쩌면 1...",  
      image: "https://movie-phinf.pstatic.net/20190109_149/1546998123676c6Ljj_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "기바나움",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=174830"  
    },  
    {  
      comment: "저는 개인적으로 매우 감동깊게 보았습니다. 루미에 혼획되지 마시고 보시길 추천드립니다.! 이미지와 장면에서..... 내용을 말씀드리고 싶지만..  
      desc: "귀여운 소년 '이든'의 단짝 반려견 '베일리'는 행복한 생을 미감한다. 하지만 눈을 떠보니 다시 시작...",  
      image: "https://movie-phinf.pstatic.net/20181031_264/1540949415228uy0j3_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "베일리 어개인",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=144906"  
    },  
    {  
      comment: "[인터뷰] 자민당과 미국 속내까지 고고한 다큐멘터리 영화 <주전장>, 미키 데자카 감독 19.07.03 11:461글: 강연주(play224) 김종성(qqqkim  
      desc: "일본의 인종차별 문제를 다룬 영상을 올린 후 우리들의 공격 대상이 된 일본계 미국인 유튜버, 미키 데...",  
      image: "https://movie-phinf.pstatic.net/20190712_279/1562912698606Qy0GH_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "주전장",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=179518"  
    },  
    {  
      comment: "포드 v 페라리 감독 제임스 맨골드 출연 크리스찬 베일, 맷 데이먼 개봉 2019.12.04. 미국 평점 리뷰보기 ...",  
      desc: "1960년대, 매출 감소에 빠진 '포드'는 판매 활로를 찾기 위해 스포츠카 레이스를 장악한 절대적 1위 '...',  
      image: "https://movie-phinf.pstatic.net/20191204_91/15754257752042khvB_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "포드 v 페라리",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=181710"  
    },  
    {  
      comment: "안녕하세요. 케이론입니다! 오늘 리뷰할 영화는 아일라입니다. 지금히 개인적인 몇 줄 평 6.25 전쟁 당시 터키 참전 병사 '슬레이만' 과 전쟁 고아가  
      desc: "1950년 한국전쟁에 파병된 '슬레이만'은 칠흑 같은 아동 속 홀로 남겨진 5살 소녀를 발견한다. 전쟁과 ...",  
      image: "https://movie-phinf.pstatic.net/20180524_108/1527129319423XRTci_JPEG/movie_image.jpg?type=m665_443_2",  
      title: "아일라",  
      url: "https://movie.naver.com/movie/bi/mi/basic.nhn?code=169240"  
    }  
  ]  
}
```

- articles '리스트' 안에 '딕셔너리'가 있는 형태로 되어있네요! 이 각 딕셔너리는 형태가 다 똑같죠?

- 찬찬히 살펴보니 아래처럼 되어있군요!
- comment(카드내용 코멘트), desc(카드내용), image(카드 이미지), title(카드 제목), url(카드를 클릭했을 때 연결되는 url주소)

- JSON View 팁!

- 화면에서 원하는 부분에 커서를 두면 선택된 부분이 파란색으로 바뀌고, 하단에 어떤 부분이 선택되어 있는지 보입니다.(그림에서는 articles[0]) JSONView로 분석하면, JSON 형태를 좀 더 쉽게 분석하실 수 있을 거예요!

▼ 2) [👉 투터와 함께] API를 화면 코드에 사용하기 - 포스팅 가져오기 API

1. 준비하기

- 2주차에 완성했던 '나홀로 메모장'을 사용합니다. 아래 코드를 복사 붙여넣기해서 week03에 `memo.html` 파일로 만들어 주세요. 단계별로 함수를 만들어가며 복습해볼거예요.

▼ [💻 코드 나홀로 메모장 01]

```
<!Doctype html>
<html lang="ko">

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

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
        integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGFAW/dAiS6JXm" crossorigin="anonymous">

    <!-- JS -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/umd/popper.min.js"
        integrity="sha384-ApNbgh8Y+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFpSkvXusvfa0b4Q"
        crossorigin="anonymous"></script>

    <!-- 구글폰트 -->
    <link href="https://fonts.googleapis.com/css?family=Stylish&display=swap" rel="stylesheet">

<title>스파르타코딩클럽 | 나홀로 메모장</title>

<!-- style -->
<style type="text/css">
    * {
        font-family: 'Stylish', sans-serif;
    }

    .wrap {
        width: 900px;
        margin: auto;
    }

    .comment {
        color: blue;
        font-weight: bold;
    }

    #post-box {
        width: 500px;
        margin: 20px auto;
        padding: 50px;
        border: black solid;
        border-radius: 5px;
    }
</style>
<script>
    function openClose() {
        // id 값 post-box의 display 값이 block 이면(= 눈에 보이면)
        if ($("#post-box").css("display") == "block") {
            // post-box를 가리고
            $("#post-box").hide();
            // 다시 버튼을 클릭하면, 박스 열기를 할 수 있게 텍스트 바꿔두기
            $("#btn-post-box").text('포스팅 박스 열기');
        } else {
            // 아니면(눈에 보이지 않으면) post-box를 펴라
            $("#post-box").show();
            // 다시 버튼을 클릭하면, 박스 닫기를 할 수 있게 텍스트 바꿔두기
            $("#btn-post-box").text("포스팅 박스 닫기");
        }
    }
</script>
```

```

</head>

<body>
<div class="wrap">
    <div class="jumbotron">
        <h1 class="display-4">나홀로 링크 메모장!</h1>
        <p class="lead">중요한 링크를 저장해두고, 나중에 볼 수 있는 공간입니다</p>
        <hr class="my-4">
        <p class="lead">
            <button onclick="openClose()" id="btn-post-box" type="button" class="btn btn-primary">포스팅 박스 열기</button>
        </p>
    </div>
    <div id="post-box" class="form-post" style="display:none">
        <div>
            <div class="form-group">
                <label for="post-url">아티클 URL</label>
                <input id="post-url" class="form-control" placeholder="">
            </div>
            <div class="form-group">
                <label for="post-comment">간단 코멘트</label>
                <textarea class="form-control" rows="2"></textarea>
            </div>
            <button type="button" class="btn btn-primary">기사저장</button>
        </div>
    </div>
    <div id="cards-box" class="card-columns">
        <div class="card">
            
            <div class="card-body">
                <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
                <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세 무궁화 삼</p>
                <p class="card-text comment">여기애 코멘트가 들어갑니다.</p>
            </div>
        </div>
        <div class="card">
            
            <div class="card-body">
                <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
                <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세 무궁화 삼</p>
                <p class="card-text comment">여기애 코멘트가 들어갑니다.</p>
            </div>
        </div>
        <div class="card">
            
            <div class="card-body">
                <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
                <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세 무궁화 삼</p>
                <p class="card-text comment">여기애 코멘트가 들어갑니다.</p>
            </div>
        </div>
        <div class="card">
            
            <div class="card-body">
                <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
                <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세 무궁화 삼</p>
                <p class="card-text comment">여기애 코멘트가 들어갑니다.</p>
            </div>
        </div>
        <div class="card">
            
            <div class="card-body">
                <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
                <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세 무궁화 삼</p>
                <p class="card-text comment">여기애 코멘트가 들어갑니다.</p>
            </div>
        </div>
    </div>
</div>

```

```
</div>
</div>
</body>

</html>
```

2. 우선, 웹 페이지(document) 로드가 다 되면(ready) 실행되는 함수를 하나 만듭니다



구글링 해보기: javascript 로딩 후 실행

【💻 코드 나홀로 메모장 02】화면 로딩 후 함수 실행

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

function showPost() {
    console.log("화면 로딩 후 잘 실행되었습니다");
}
```

3. API 응답(response) 데이터를 확인하기



잠깐!

웹 브라우저(크롬) 주소창에서 url 주소를 입력하고 enter! 치는 것은 GET 방식 요청과 같아요.
크롬에서 url입력해서 enter 치는 것 = GET 요청 = ajax로 GET 요청

- 앞 1) API 분석에서 크롬에서 응답 데이터인 JSON 어떻게 생겼나 분석했었죠? 아까 크롬에서 본 JSON과 코드로 ajax로 GET 요청한 응답 데이터 JSON은 같은 내용이겠네요!
- 우리는 지금 어떤 데이터가 있는지 단계별로 차근차근 확인해서 접근하는 걸 배우고 있어요. 어떤 응답 결과인지 데이터를 잘 분석해야 내가 원하는 데이터를 화면에 보여줄 수 있겠죠?
- ajax로 요청을 하고, 응답 결과가 잘 왔는지 console.log로 확인해봅시다!

【💻 코드 나홀로 메모장 03】

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

function showPost() {
    $.ajax({
        type: "GET",
        url: "http://spartacodingclub.shop/post",
        data: {},
        success: function(response){
            console.log(response);
        }
    });
}
```



console.log 함수를 사용했으니까 출력값을 크롬 개발자도구 console 창에서 확인할 수 있겠죠?

response['result']에 데이터 정상 여부('success')가 있고, response['articles']에 리스트 형태로 아티클들이 들어가 있습니다.

4. 기사데이터를 출력해봅시다.

【💻 코드 나홀로 메모장 04】showPost 함수 수정 1

```
$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/post",
    data: {},
```

```

        success: function(response){
            console.log(response['articles'])
        }
    });

```

5. articles를 돌면서, 하나씩 출력해봅니다.

코드 나홀로 메모장 05] showPost 함수 수정 2

```

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

```

6. article 내용(comment, desc, image, title, url)으로 카드를 만들어주는 함수를 만들어봅시다!

코드 나홀로 메모장 06] makeCard 함수 만들기

```

function makeCard(image, url, title, desc, comment) {
    let tempHtml = `<div class="card">
        
        <div class="card-body">
            <a href="${url}" target="_blank" class="card-title">${title}</a>
            <p class="card-text">${desc}</p>
            <p class="card-text comment">${comment}</p>
        </div>
    </div>`;
    $("#cards-box").append(tempHtml);
}

```

7. makeCard 함수를 ajax에 연결합니다.

코드 나홀로 메모장 07] showPost 함수 수정03

```

$.ajax({
    type: "GET",
    url: "http://spartacodingclub.shop/post",
    data: {},
    success: function(response){
        let articles = response["articles"];
        for (let i = 0; i < articles.length; i++) {
            let article = articles[i];
            makeCard(article["image"], article["url"], article["title"], article["desc"], article["comment"])
        }
    }
});

```

8. 먼저 있던 카드들을 지워줍니다.

코드 나홀로 메모장 08]

```

$(document).ready(function () {
    $("#cards-box").empty();
    showPost();
});

```

9. 1~8단계 통해 완성된 나홀로 메모장 코드

▼ script 부분 완성 코드

```

<script>
$(document).ready(function () {
    $("#cards-box").empty();

```

```

        showPost();
    });

    function showPost() {
        $.ajax({
            type: "GET",
            url: "http://spartacodingclub.shop/post",
            data: {},
            success: function (response) {
                let articles = response["articles"];
                for (let i = 0; i < articles.length; i++) {
                    let article = articles[i];
                    makeCard(article["image"], article["url"], article["title"], article["desc"], article["comment"]);
                }
            });
    }

    function makeCard(image, url, title, desc, comment) {
        let tempHtml = `<div class="card">
            
            <div class="card-body">
                <a href="${url}" target="_blank" class="card-title">${title}</a>
                <p class="card-text">${desc}</p>
                <p class="card-text comment">${comment}</p>
            </div>
        </div>`;
        $("#cards-box").append(tempHtml);
    }

    function openClose() {
        // id 값 post-box의 display 값이 block 이면(= 눈에 보이면)
        if ($("#post-box").css("display") == "block") {
            // post-box를 가리고
            $("#post-box").hide();
            // 다시 버튼을 클릭하면, 박스 열기를 할 수 있게 텍스트 바꿔두기
            $("#btn-post-box").text("포스팅 박스 열기");
        } else {
            // 아니면(눈에 보이지 않으면) post-box를 펴라
            $("#post-box").show();
            // 다시 버튼을 클릭하면, 박스 닫기를 할 수 있게 텍스트 바꿔두기
            $("#btn-post-box").text("포스팅 박스 닫기");
        }
    }

```

▼ [ 코드! 나홀로 메모장 전체 완성 코드]

```

<!Doctype html>
<html lang="ko">

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

        <!-- Bootstrap CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
              integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
              crossorigin="anonymous">

        <!-- JS -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
        <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"
               integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
               crossorigin="anonymous"></script>

        <!-- 구글폰트 -->
        <link href="https://fonts.googleapis.com/css?family=Stylish&display=swap" rel="stylesheet">

    <title>스파르타코딩클럽 | 나홀로 메모장</title>

    <!-- style -->
    <style type="text/css">
        * {
            font-family: "Stylish", sans-serif;
        }

        .wrap {
            width: 900px;
            margin: auto;
        }
    </style>

```

```

.comment {
    color: blue;
    font-weight: bold;
}

#post-box {
    width: 500px;
    margin: 20px auto;
    padding: 50px;
    border: black solid;
    border-radius: 5px;
}
</style>
<script>
$(document).ready(function () {
    $("#cards-box").empty();
    showPost();
});

function showPost() {
    $.ajax({
        type: "GET",
        url: "http://spartacodingclub.shop/post",
        data: {},
        success: function (response) {
            let articles = response["articles"];
            for (let i = 0; i < articles.length; i++) {
                let article = articles[i];
                makeCard(article["image"], article["url"], article["title"], article["desc"], article["comment"]);
            }
        }
    });
}

function makeCard(image, url, title, desc, comment) {
    let tempHtml = `<div class="card">
        
        <div class="card-body">
            <a href="${url}" target="_blank" class="card-title">${title}</a>
            <p class="card-text">${desc}</p>
            <p class="card-text comment">${comment}</p>
        </div>
    </div>`;
    $("#cards-box").append(tempHtml);
}

function openClose() {
    // id 값 post-box의 display 값이 block 이면(= 눈에 보이면)
    if ($("#post-box").css("display") == "block") {
        // post-box를 가리고
        $("#post-box").hide();
        // 다시 버튼을 클릭하면, 박스 열기를 할 수 있게 텍스트 바꿔두기
        $("#btn-post-box").text("포스팅 박스 열기");
    } else {
        // 아니면(눈에 보이지 않으면) post-box를 펴라
        $("#post-box").show();
        // 다시 버튼을 클릭하면, 박스 닫기를 할 수 있게 텍스트 바꿔두기
        $("#btn-post-box").text("포스팅 박스 닫기");
    }
}
</script>

</head>

<body>
    <div class="wrap">
        <div class="jumbotron">
            <h1 class="display-4">나홀로 링크 메모장!</h1>
            <p class="lead">중요한 링크를 저장해두고, 나중에 볼 수 있는 공간입니다</p>
            <hr class="my-4">
            <p class="lead">
                <button onclick="openClose()" id="btn-post-box" type="button" class="btn btn-primary">포스팅 박스
            </button>
        </p>
    </div>
    <div id="post-box" class="form-post" style="display:none">
        <div>
            <div class="form-group">
                <label for="post-url">아래를 URL</label>
                <input id="post-url" class="form-control" placeholder="">
            </div>
            <div class="form-group">
                <label for="post-comment">간단 코멘트</label>
                <textarea class="form-control" rows="2"></textarea>
            </div>
            <button type="button" class="btn btn-primary">기사저장</button>
        </div>
    </div>
</body>

```

```

</div>
<div id="cards-box" class="card-columns">
    <div class="card">
        
        <div class="card-body">
            <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
            <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세</p>
            <p class="card-text comment">여기에 코멘트가 들어갑니다.</p>
        </div>
    </div>
    <div class="card">
        
        <div class="card-body">
            <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
            <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세</p>
            <p class="card-text comment">여기에 코멘트가 들어갑니다.</p>
        </div>
    </div>
    <div class="card">
        
        <div class="card-body">
            <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
            <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세</p>
            <p class="card-text comment">여기에 코멘트가 들어갑니다.</p>
        </div>
    </div>
    <div class="card">
        
        <div class="card-body">
            <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
            <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세</p>
            <p class="card-text comment">여기에 코멘트가 들어갑니다.</p>
        </div>
    </div>
    <div class="card">
        
        <div class="card-body">
            <a href="#" class="card-title">여기 기사 제목이 들어가죠</a>
            <p class="card-text">기사의 요약 내용이 들어갑니다. 동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라만세</p>
            <p class="card-text comment">여기에 코멘트가 들어갑니다.</p>
        </div>
    </div>
</div>
</body>

</html>

```



하나의 함수를 만들 때에도

값을 출력해서 확인하고(console.log), 차근차근 단계를 밟아가며 진행했죠?

프로그래밍할 때는 단계별로 차근차근 접근해야 한다는 걸 기억하세요!

[1시간] : 파이썬(Python) 기초 문법 익히기

▼ 3) 파이썬을 설치한다는 것의 의미



Python을 설치한다?

→ 일종의 번역팩을 설치한다고 생각하면 됩니다.

컴퓨터는 101010001 과 같은 언어만 알아듣는다고 했지요? 파이썬 문법으로 된 것을 101010001로 변환해 줄 수 있도록, 번역기를 설치하고 프로그래밍을 쉽게 할 수 있는 기본 코드(예를 들면, python 기본 함수)를 설치하는 것입니다.

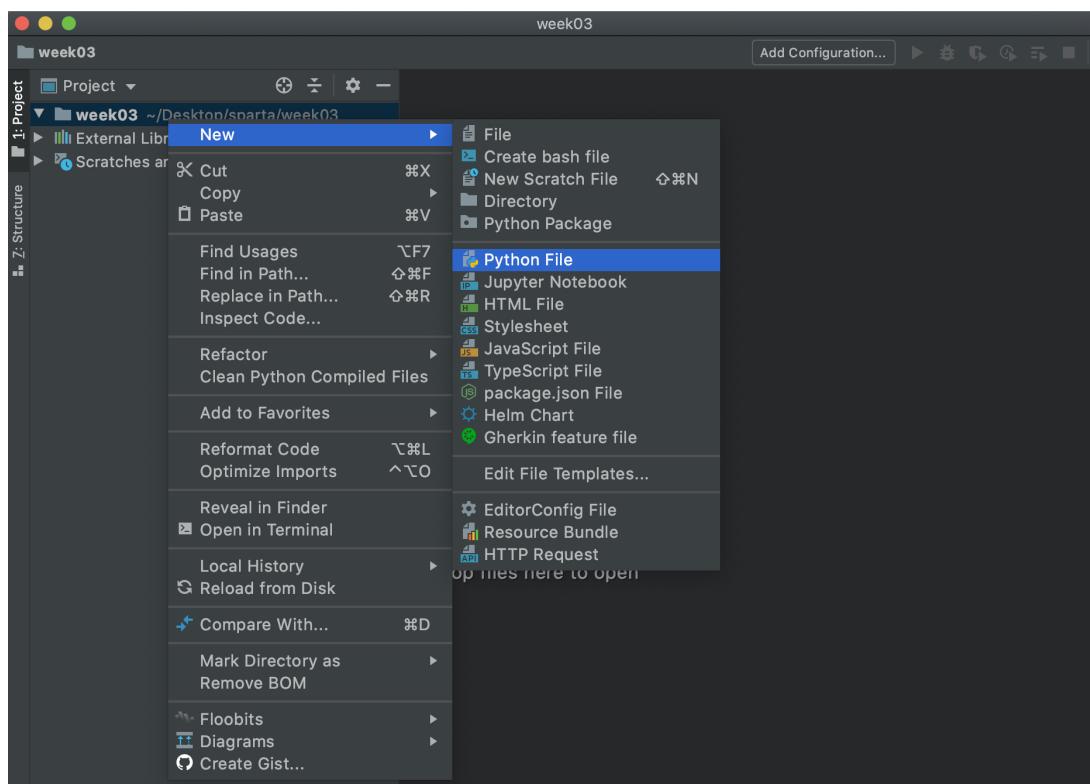
▼ 4) [튜터와 함께] 첫 파이썬 파일 실행

1. Pycharm 으로 바탕화면의 sparta 안 week03 폴더를 엽니다.

2. hello.py 파일을 만듭니다.

- 파일 생성 단축키 : 폴더 선택하고 ctrl(cmd) + n

▼ 화면에서 클릭으로 만들기



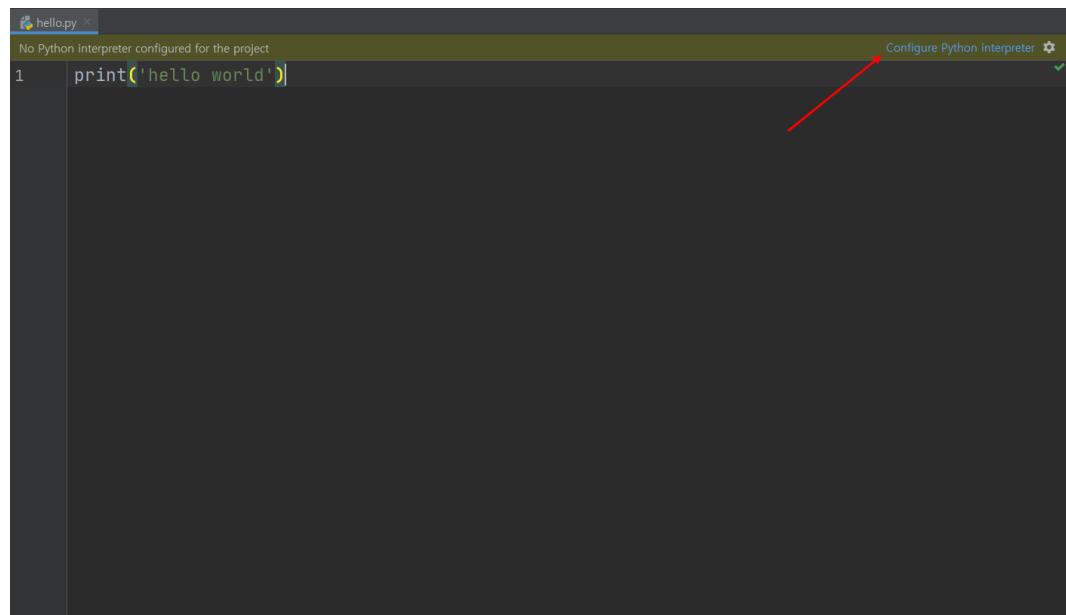
3. hello.py 파일 안에 다음 내용을 붙여넣습니다.

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

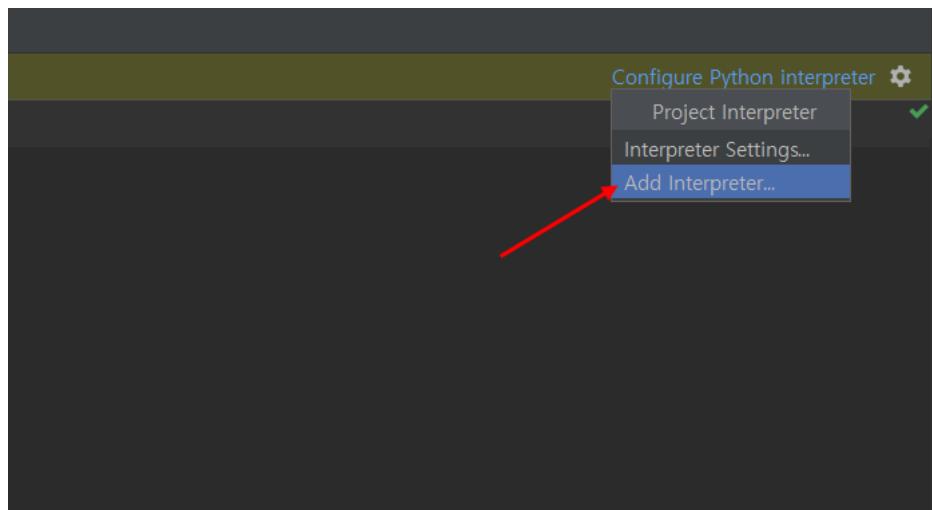
4. 그럼 아래와 같은 문구를 만날 거예요! 파이썬 세팅해주기!

▼ 튜터님과 그림을 보며 함께 설정해보아요!

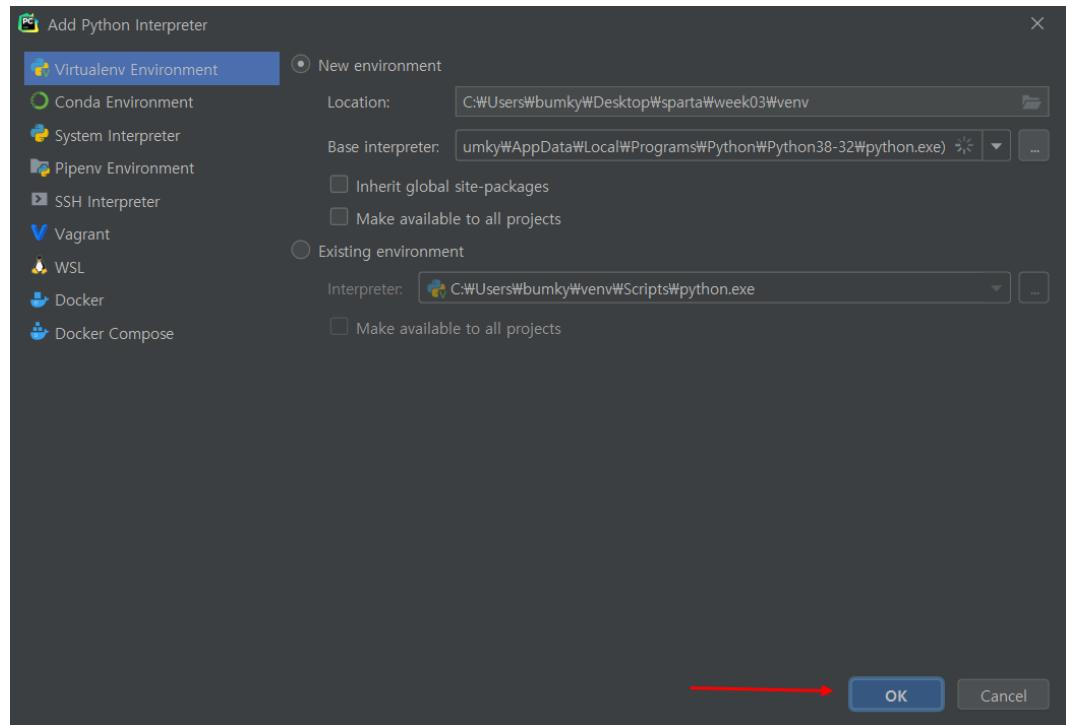
1. Configure Python Interpreter 클릭



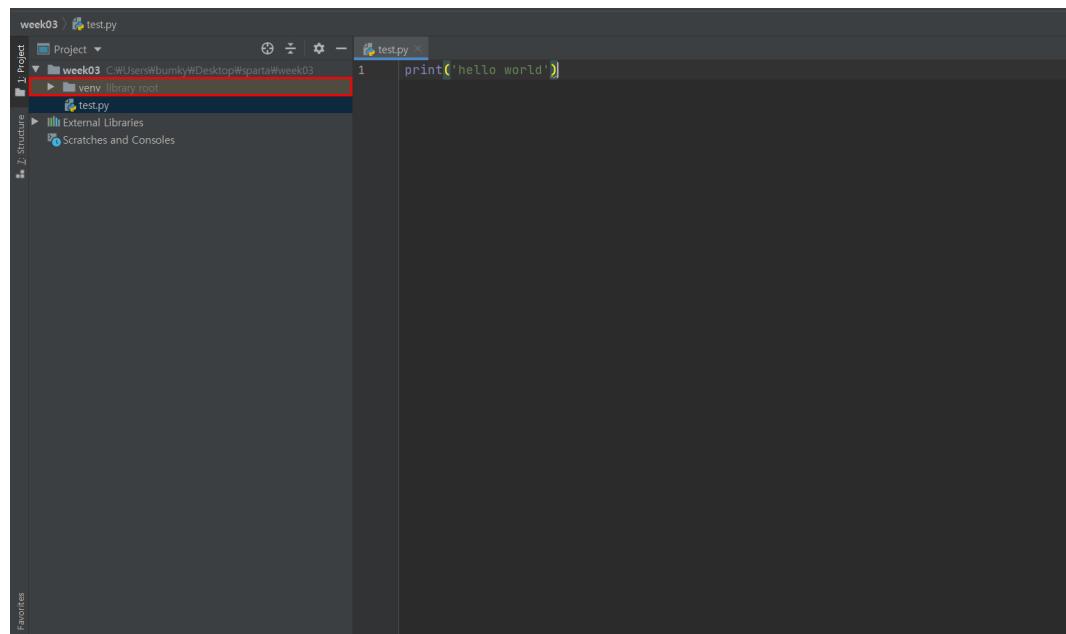
2. Add interpreter를 클릭!



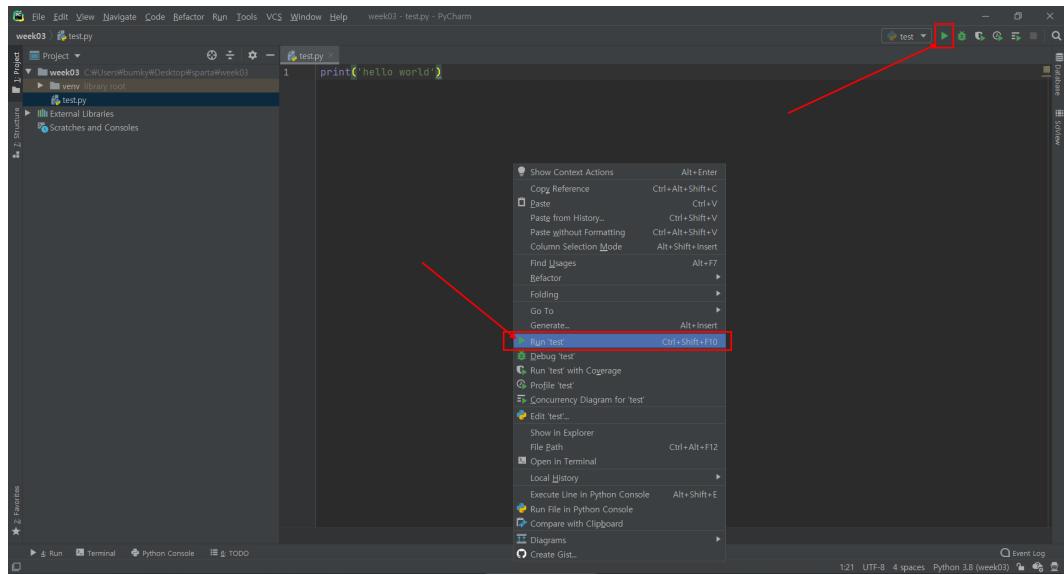
3. 그대로 OK를 클릭하면



4. 시간이 조금 흐르고(30~60초 정도) venv라는 폴더를 볼 수 있게 됩니다!

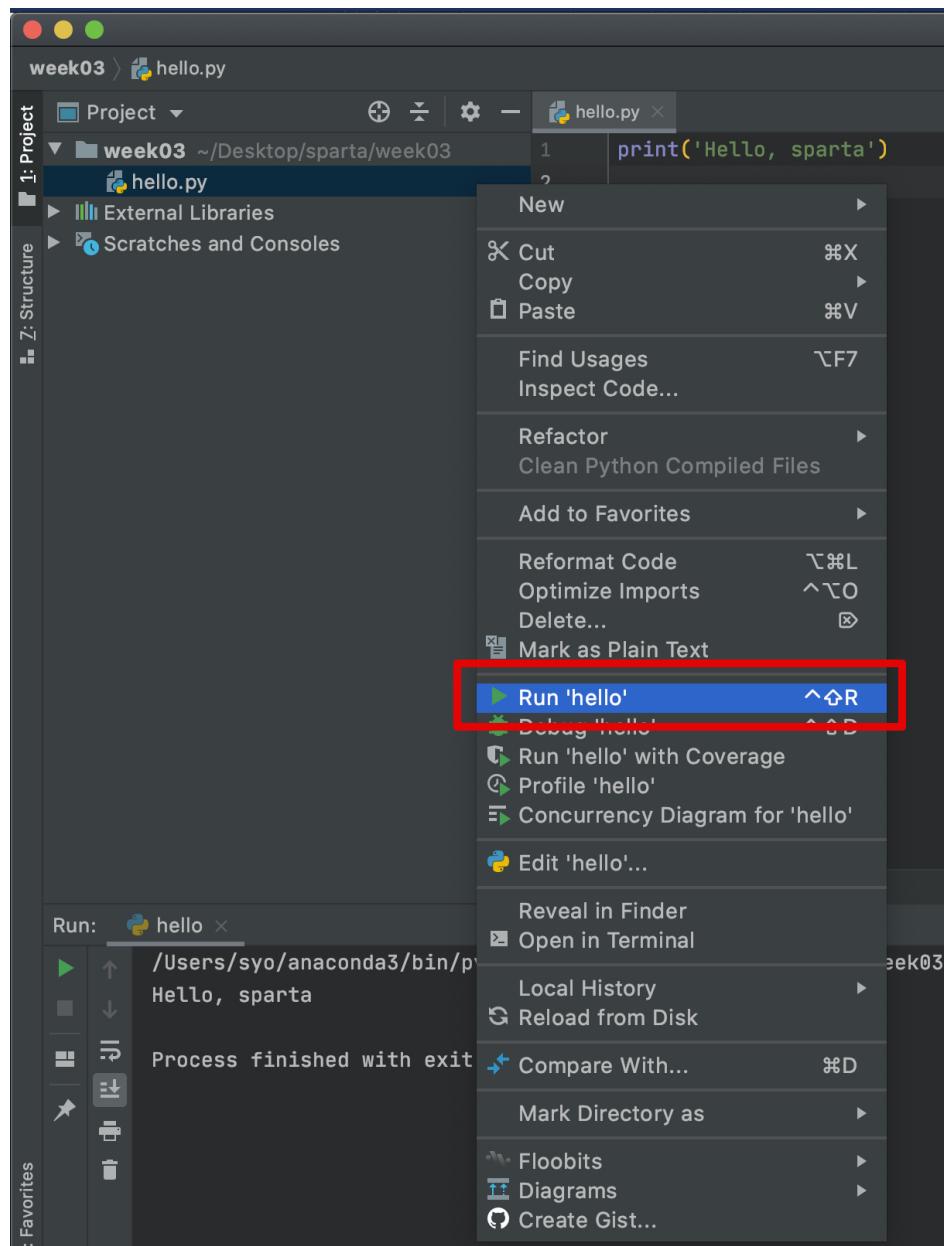


5. 마우스 오른쪽 버튼을 누르고 run을 클릭하면 실행! (또는 오른쪽 초록 삼각형 클릭!)



5. Pycharm에서 파일 실행하기

- 단축키 Windows - **ctrl + shift + F10** / Mac - **ctrl+ shift + r**
- ▼ 화면에서 클릭으로 실행



6. 아래와 같이 뜨면 성공! 🎉

A screenshot of the PyCharm 'Run' tool window. It shows the command `/Users/[REDACTED]/Desktop/sparta/week03/hello.py` and the output: `Hello, sparta` followed by `Process finished with exit code 0`.

- 만약 `configure python interpreter...` 메시지가 계속 보이면서 실행이 잘 안되면 Python Interpreter 문제일 수도 있습니다. 이 경우 week03 폴더를 삭제하고, 퓨터님의 도움을 받아 file→new project로 week03 폴더를 다시 만들어 진행해주세요.

▼ 잠깐! 파이썬 2 가 아니라 파이썬 3 로 설정되어있는지 꼭 확인해주세요.

- 우리는 파이썬 3를 사용합니다! 2와 3는 문법이 다르고 쓸 수 있는 라이브러리도 다르답니다. 파이썬 2는 더 이상 보완 및 관리가 되지 않아요([링크](#))
- Mac에서는 운영체제 실행을 위해 파이썬 2가 기본적으로 깔려 있기 때문에 파이썬 2로 실행하도록 설정이 자동으로 잡힐 수 있어요. 꼭 확인해주세요~

▼ 5) 파이썬 문법을 시작하기에 앞서..

▼ 6) 파이썬 기초 문법

- Javascript에서 보통의 프로그래밍 언어는 변수, 자료형, 함수, 조건문, 반복문 요소가 있다는 말 기억나시죠? 파이썬에서도 이 순서대로 배워볼 거예요.



리마인드!

1. 출력하기

값을 확인하거나, 에러를 찾을 때 자주 쓰입니다. 출력해서 버그찾기(Debugging By Printing) 스킬! Python에서는 출력할 때, `print('출력할 값')` 사용합니다.

2. 이름 짓기 (naming)

담고 있는 데이터, 기능을 잘 나타낼 수 있는 이름으로 지어주세요. 보통 변수는 명사형으로, 함수는 동사형으로 많이 짓습니다.

파이썬에서는 이름짓기 규칙(naming convention)은 snake style 을 사용합니다. (_로 단어 연결하기. 예.

`first_name`)

👉파이썬 공식 스타일 가이드([PEP 8 링크](#) / [파이썬 자습서-코딩 스타일](#))

- 퓨터님은 Pycharm에서 Python Console(REPL) 또는, terminal에서 `python` 을 입력해 창을 띄워서 진행해주세요.

▼ 변수 & 기본연산

```
a = 3 # 3을 a에 넣는다
print(a)

b = a # a를 b에 넣는다
print(b)

a = a + 1 # a+1을 다시 a에 넣는다
print(a)

num1 = a * b # a*b의 값을 num1이라는 변수에 넣는다
print(num1)

num2 = 99 # 99의 값을 num2이라는 변수에 넣는다
print(num2)
```

▼ 자료형

[💻 코드 자료형 01] 숫자, 문자형

```
name = 'Harry' # 변수에는 문자열이 들어갈 수도 있고,
print(name)

num = 12 # 숫자가 들어갈 수도 있고,
print(num)

number_status = True # True 또는 False -> "Boolean"형이 들어갈 수도 있습니다.
print(number_status)
```

[💻 코드 자료형 02] 리스트 형 (Javascript의 배열형과 동일)

```
a_list = [] # 비어있는 리스트 만들기
a_list.append(1) # 리스트에 값을 넣는다
print(a_list)
```

```
a_list.append([2, 3]) # 리스트에 [2,3]이라는 리스트를 다시 넣는다
print(a_list)

# print로 값 확인해보기
# 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
```

코드 자료형 03] Dictionary 형 (Javascript의 dictionary형과 동일)

```
a_dict = {} # 비어있는 딕셔너리 만들기

wizard = {'name': 'Harry Potter', 'age': 40}
print(wizard)

wizard['height'] = 173
print(wizard)

# print로 값 확인해보기
# wizard의 값은? {'name':'Harry Potter', 'age':40, 'height':173}
# wizard['name']의 값은? 'Harry Potter'
# wizard['age']의 값은? 40
# wizard['height']의 값은? 173
```

코드 자료형 04] Dictionary 형과 List형의 조합

```
wizards = [{"name": "Harry Potter", "age": 40}, {"name": "Ron Weasley", "age": 40}]
print(wizards)

# print로 값 확인해보기
# wizards[0]['name']의 값은? 'Harry Potter'
# wizards[1]['name']의 값은? 'Ron Weasley'

new_wizard = {'name': 'Albus Potter', 'age': 14}
wizards.append(new_wizard)
print(wizards)

# print로 값 확인해보기
# wizards의 값은? [{"name":'Harry Potter','age':40}, {"name":'Ron Weasley','age':40}, {"name":'Albus Potter','age':14}
# wizards[2]['name']의 값은? 'Albus Potter'
```



파이썬에서는 들여쓰기(indent)로 블록 단위를 나눕니다(JS에서는  를 썼었죠?).

들여쓰기를 잘못하면 들여쓰기 에러(indentation error)가 발생하죠! 들여쓰기가 매우 중요합니다.

▼ 함수



리마인드!

왠지 이건 있을 것 같은데?(예 - 특정 문자를 바꾸고 싶다 등) 싶으면 직접 만들지 말고 구글에서 먼저 검색해보세요!

- 함수 기본 생김새

```
# 만들기
def 함수이름(필요한 변수들) :
    내릴 명령어들을 순차적으로 작성

# 사용하기
함수이름(필요한 변수들)
```

- 예시

```
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)

# print로 값 확인해보기
# result라는 변수의 값은?

```

▼ 조건문 - ! 패턴 !에 집중하세요

- 역시 형식만 조금 다를 뿐 논리는 같습니다! 만약 조건을 만족한다면 (`if` 판별식)과 그렇지 않다면 (`else`)로 구성

 **코드 조건문 01]**

```

def is_even(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 = is_even(20)

# print로 값 확인해보기
# result의 값은 무엇일까요?

```

 **코드 조건문 02]**

```

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

# is_adult(30) 하면 무엇이 출력될까요?

```

 **코드 조건문 03] if / elif / else**

```

# 조건을 여러 개 사용하고 싶을 때
def check_generation(age):
    if age > 120:
        print('와 19세기에 태어나셨군요!')
    elif age >= 80:
        print('80세 이상! 인생은 여든부터!')
    else:
        print('젊으시군요! 장래희망이 뭔가요?')

my_age = 55
check_generation(my_age)

```

▼ 반복문 - ! 패턴 !에 집중하세요



Python에서의 반복문은, 리스트나 문자열의 요소들을 하나씩 꺼내쓰는 형태입니다. 즉, 임의의 열(sequence, 리스트나 문자열처럼)의 항목들을 그 순서대로 꺼내어 반복합니다.

우리 수업에서는 리스트와 함께 쓰입니다! 아래 ! 패턴 !을 잘 기억해주세요!

 **코드 반복문 01] 기본**

```

fruits = ['사과', '배', '감', '귤']

for fruit in fruits: # fruit은 우리가 임의로 지어준 이름입니다.
    print(fruit) # 사과, 배, 감, 귤 하나씩 끌어내어 출력합니다.

```

 **코드 반복문 02-1] 살짝 응용해볼까요? - 과일 갯수 세기 함수**

```

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

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

# 사과의 갯수를 출력합니다.
print(count)

```

코드 반복문 02-2] 함수로 만들어봅니다

```

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

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

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

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

```

코드 반복문 03-1] 예제 한 번 더!

```

wizards = [
    {'name': '해리', 'age': 40},
    {'name': '히마이오니', 'age': 40},
    {'name': '론', 'age': 41},
]

# 모든 마법사의 이름과 나이를 출력
for wizard in wizards:
    print(wizard['name'], wizard['age'])

```

코드 반복문 03-2] 함수를 만들어 봅시다!

```

professor_wizards = [
    {'name': '덤블도어', 'age': 116},
    {'name': '맥고나걸', 'age': 85},
    {'name': '스네이프', 'age': 60},
]

# 이번엔, 반복문과 조건문을 활용한 함수를 만들어봅시다.
# 마법사의 이름을 받으면, age를 리턴해주는 함수
def get_age(name, wizards):
    # wizards! 뒷 줄 함수 선언에서 사용한 변수죠? 함수 사용하는 쪽에서 쓰는 변수명 아닙니다!
    for wizard in wizards:
        if wizard['name'] == name:
            return wizard['age']
    return '해당하는 이름이 없습니다'

print(get_age('덤블도어', professor_wizards))
print(get_age('맥고나걸', professor_wizards))

```

▼ 7) 파이썬 패키지(package) 설치하기



패키지? 라이브러리? →

Python에서 패키지는 모듈(일종의 기능들 묶음)을 모아 놓은 단위입니다. 이런 패키지의 묶음을 라이브러리라고 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 공구함



문제상황:

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

그런데 회사팀장님 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** 이에요.



정리하자면,

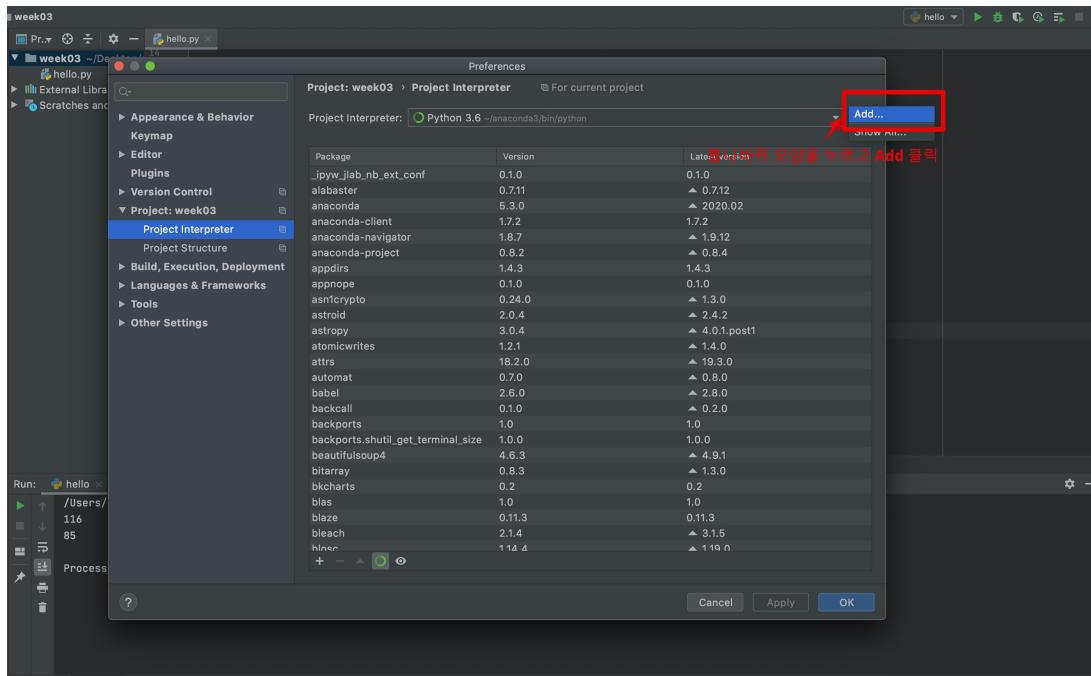
가상환경(virtual environment)은

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

출처 : [파이썬 공식 용어집- 가상환경](#)

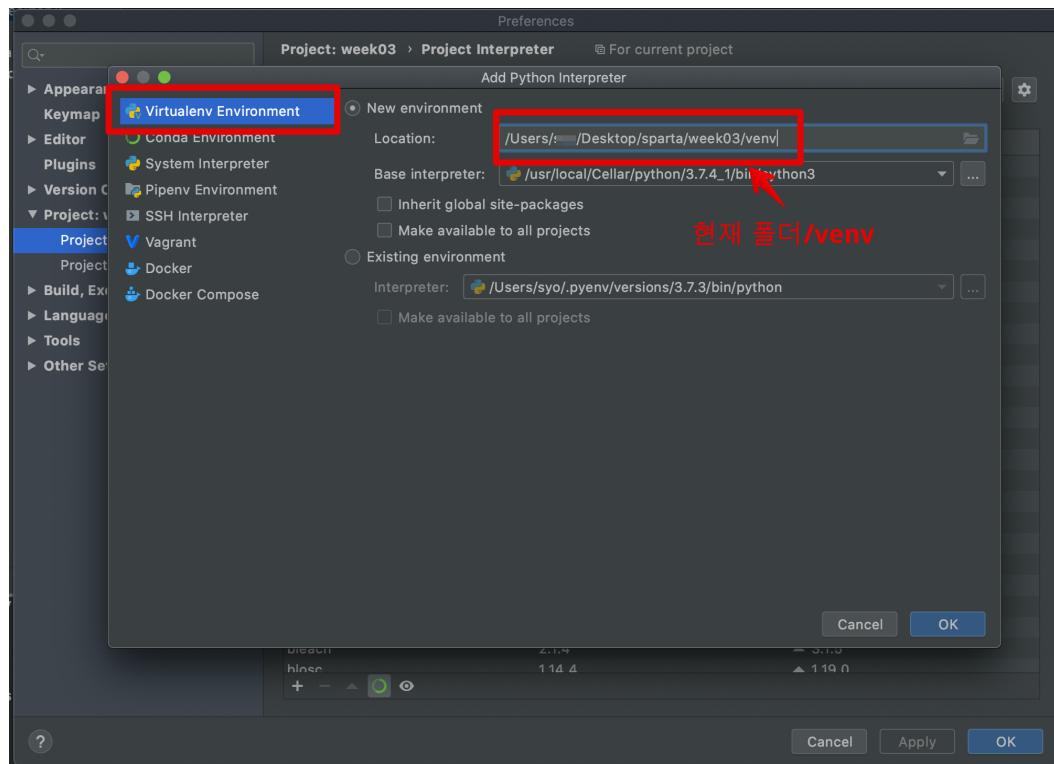
▼ 2. 가상 환경 설치 및 확인

- Windows : File → setting → project interpreter
Mac : Preference → Project Interpreter로 접근
 - 앞에서 configure interpreter 세팅하는 과정을 거쳤다면 아래 그림처럼 뜰 거에요.
- ▼ 만약 아래 그림처럼 안뜬다면 여기를 따라 가상환경을 세팅하면 됩니다!
- 톱니바퀴 add 버튼 클릭



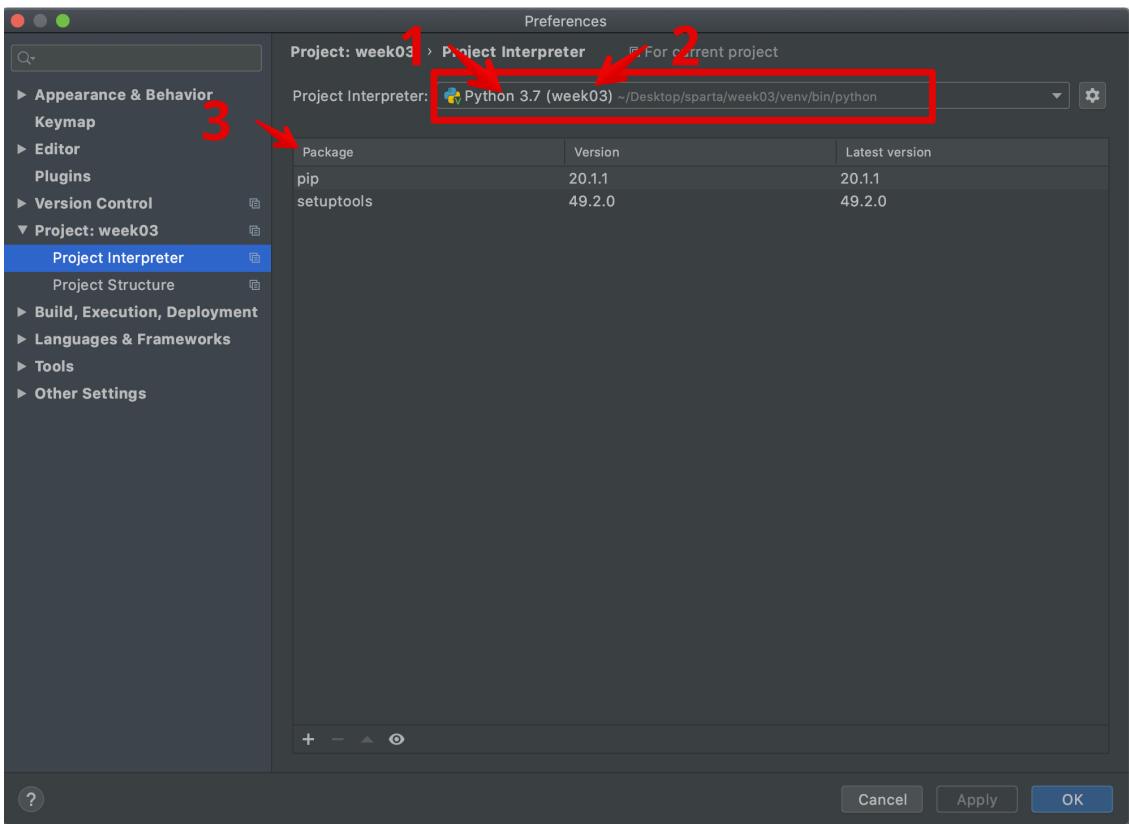
- virtual environment - New environment 선택

- Location을 현재 폴더/venv로 설정해주세요. 현재 프로젝트 폴더(week03) 아래에 가상환경 폴더(venv)가 생성이 됩니다.



- 설정 부분을 해석하면

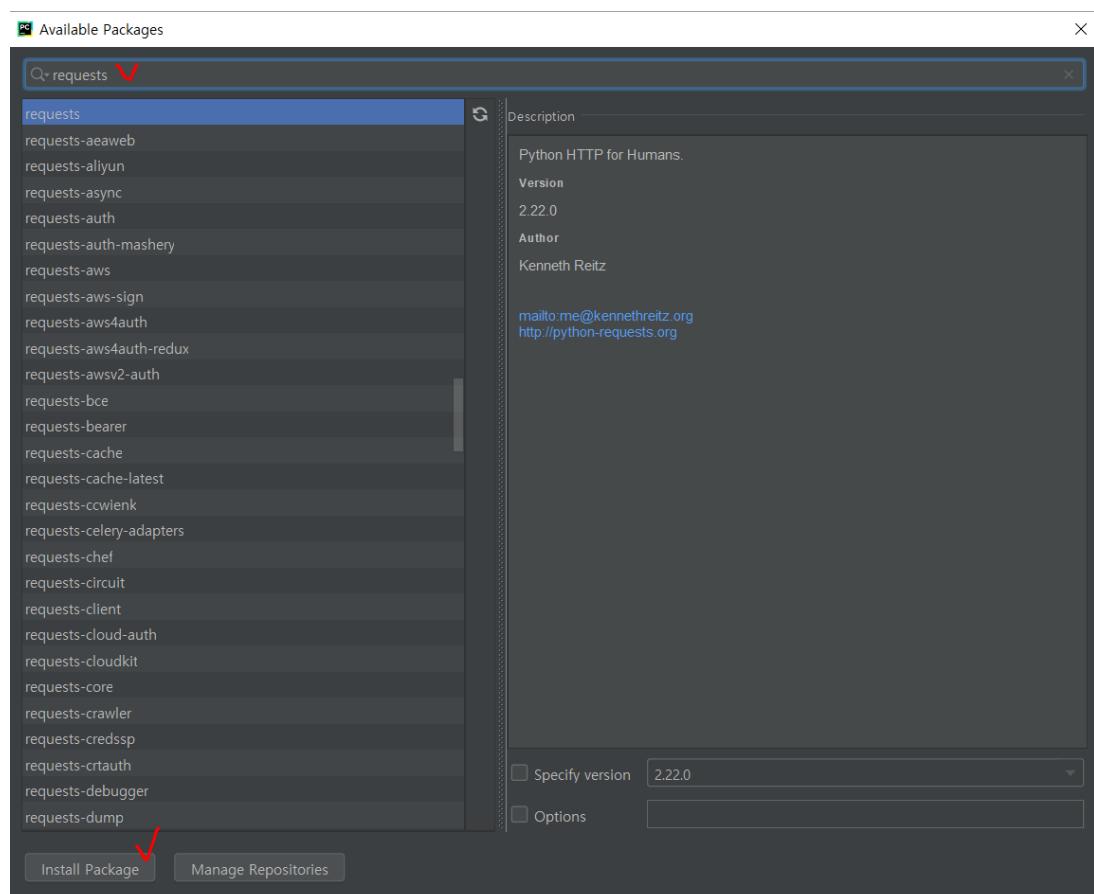
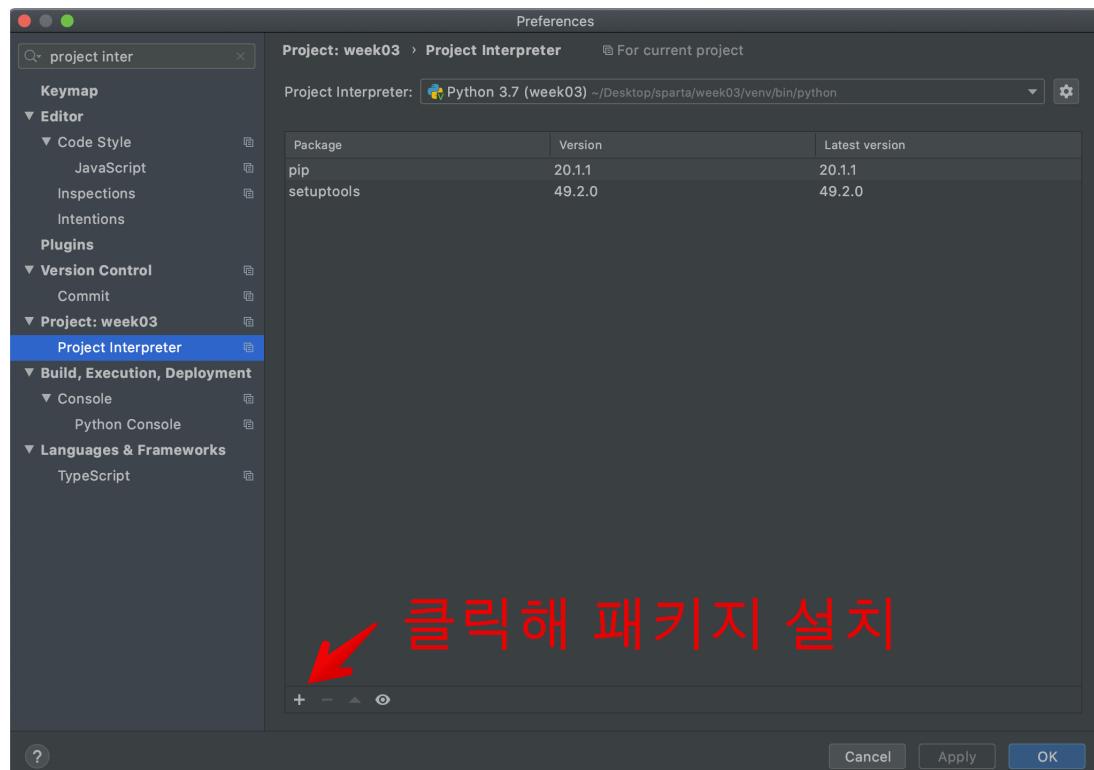
- 파이썬은 3.7 버전을 사용하고 있고
- week03이라는 가상환경 명으로 현재 폴더/venv에 가상환경이 설정
- week03 가상환경에 설치된 패키지는 두 가지가 있네요.



▼ 3. 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!



▼ 8) 설치한 Requests 라이브러리 사용 & 파이썬 기초 문법 연습

- 서울시 OpenAPI 를 사용해 List / Dictionary / 함수 / if / for문 연습을 해보겠습니다.



서울시 권역별 실시간 대기환경 현황

API 문서 : ([링크](#))

API 요청(request) URL :

<http://openapi.seoul.go.kr:8088/6d4d776b466c656533356a4b4b5872/json/RealtimeCityAir/1/99>

- `cityair.py` 라는 이름으로 파일 하나를 만들어 보겠습니다. 주석(단축키 : `ctrl(또는 cmd) + /`) 을 적절하게 사용해보세요.

[코드 API 연습 01] 중구의 NO2 값 가져오기

```
import requests # requests 라이브러리 설치 필요

# requests 를 사용해 요청(Request)하기
response_data = requests.get('http://openapi.seoul.go.kr:8088/6d4d776b466c656533356a4b4b5872/json/RealtimeCityAir/1/99')

# 응답(response) 데이터인 json을 쉽게 접근할 수 있게 만들어 city_air 에 담고
city_air = response_data.json()

# 값을 출력
print(city_air['RealtimeCityAir']['row'][0]['NO2'])
```

[코드 API 연습 02] 모든 구의 PM10 값을 찍어주자! (위 01번 코드는 주석처리)

```
import requests # requests 라이브러리 설치 필요

# requests 를 사용해 요청(Request)하기
response_data = requests.get('http://openapi.seoul.go.kr:8088/6d4d776b466c656533356a4b4b5872/json/RealtimeCityAir/1/99')
# 응답(response) 데이터인 json을 쉽게 접근할 수 있게 만들어 city_air 에 담고
city_air = response_data.json()

gu_infos = city_air['RealtimeCityAir']['row']

for gu_info in gu_infos:
    print(gu_info['MSRSTE_NM'], gu_info['PM10'])
```

[코드 API 연습 03] PM10 값이 20 미만인 구만 찍어주자! (위 02번 코드는 주석처리)

```
import requests # requests 라이브러리 설치 필요

# requests 를 사용해 요청(Request)하기
response_data = requests.get('http://openapi.seoul.go.kr:8088/6d4d776b466c656533356a4b4b5872/json/RealtimeCityAir/1/99')
# 응답(response) 데이터인 json을 쉽게 접근할 수 있게 만들어 city_air 에 담고
city_air = response_data.json()

gu_infos = city_air['RealtimeCityAir']['row']

for gu_info in gu_infos:
    if gu_info['PM10'] < 20:
        print(gu_info['MSRSTE_NM'], gu_info['PM10'])
```



리마인드!

파이썬에서는 들여쓰기(indent)로 블록 단위를 나눕니다(JS에서는 `{ }` 를 썼었죠?).

들여쓰기를 잘못하면 들여쓰기 에러(indentation error) 가 발생하죠! 들여쓰기가 매우 중요합니다.

[1시간]: 파이썬 갖고 놀기 - "웹 스크래핑(크롤링)"

▼ 9) 웹 스크래핑이란?

- 신문 스크래핑 해보셨나요? 신문에서 원하는 기사만 오려서 정보를 추리는 것을 스크래핑이라고 하죠? 웹 스크래핑도 똑같습니다.



웹 스크래핑(web scraping)은 웹 페이지에서 우리가 원하는 부분의 데이터를 수집해오는 것입니다. (한국에서는 같은 작업을 크롤링 crawling 이라는 용어로 혼용해서 씁니다.)

- 구글 검색할 때는 **web scraping** 으로 검색해야 우리가 배우는 페이지 추출에 대한 결과가 나오니 참고하세요.

- 참고

[Web Scarping\(wikipedia\)](#) / [Web Crawler\(wikipedia\)](#)

[Web Scraping vs Web Crawling: What's the Difference?](#)

▼ 10) 웹 스크래핑 해보기 (영화 제목)

▼ 11) 🎨 웹 스크래핑 더 해보기 (순위, 제목, 별점)

▼ Q. 아래와 같이 보이면 완성!

```

01 그린 북 9.59
02 가버나움 9.59
03 베일리 어게인 9.53
04 원더 9.49
05 포드 V 페라리 9.49
06 아일라 9.49
07 주전장 9.48
08 당갈 9.47
09 쇼생크 탈출 9.44
010 터미네이터 2:오리지널 9.44
11 보헤미안 랩소디 9.42
12 덕구 9.42
13 월-E 9.41
14 나 홀로 집에 9.41
15 아이즈 온 미 : 더 무비 9.41
16 라이언 일병 구하기 9.40
17 사운드 오브 뮤직 9.40
18 살인의 추억 9.40
19 매트릭스 9.40
20 인생은 아름다워 9.40
21 헬프 9.40
22 빠 투 더 퓨쳐 9.40
23 포레스트 검프 9.40
24 위대한 쇼맨 9.40
25 클래식 9.39
26 글래디에이터 9.39
27 센과 치히로의 행방불명 9.39
28 토이 스토리 3 9.38
29 타이타닉 9.38
30 알라딘 9.38
31 어벤져스: 엔드게임 9.38
32 안녕 베일리 9.37
33 죽은 시인의 사회 9.37
34 레옹 9.37

```

▼ A. 완성 코드

```

import requests
from bs4 import BeautifulSoup

# URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200716', 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 a가 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)

```

▼ 12) 웹 스크래핑 팁

- 웹 스크래핑은 웹 페이지 구조에 따라 조금씩 방법이 다르기 때문에 처음에는 좀 까다롭게 느껴질 수도 있어요. 하지만 여러 번 스크래핑 하다보면 여러분만의 팁이 생길 거예요!
- 필요할 때마다 bs4 함수를 검색해 활용해보세요.



스크래핑 팁 세 가지!

1. 문제 분석(화면 분석)

원하는 데이터가 화면 어디에 있는지 확인 selector 기억나시나요? 스크래핑도 원하는 부분을 가리켜서 정보를 가져오죠? 페이지 구조를 파악하는게 필요합니다.

2. 부모 요소부터 접근

내가 원하는 데이터의 부모(조상) 요소 부터 가져와보세요. 부모 요소부터 페이지 구조를 살펴보다보면 규칙을 찾을 수 있을 거예요.

3. 값 출력하기

값을 제대로 가져오고 있는지 중간중간 출력(print)



와, 지금까지 파이썬도 배우고 정보를 가져오는 스크래핑도 배워봤어요.
영화정보를 가져오는 프로그램까지 완성!

정말 재밌죠? 이제 후반부에서는 DB를 배워볼게요~

😎 문제뱅크

▼ (1) 네이버 영화 정보 가져오기 (난이도 : ★★★)

- 아래 네이버 영화 URL([10](#) 번에서 써봤죠?)을 수정해 2018년 3월 27일 기준 네이버 영화 순위의 영화의 순위, 제목, 평점 정보를 스크래핑해 출력하세요.
- 2020년 7월 16일의 네이버 영화 순위 URL
<https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200716>

▼ Q. 이렇게 출력되면 완성

01	쇼생크 탈출	9.41
02	원더	9.40
03	터미네이터 2	9.39
04	인생은 아름다워	9.39
05	매트릭스	9.38
06	레옹	9.38
07	위대한 쇼맨	9.38
08	포레스트 검프	9.38
09	죽은 시인의 사회	9.38
010	월-E	9.38
11	나 홀로 집에	9.38
12	빽 투 더 퓨처	9.38
13	토이 스토리 3	9.37
14	라이언 일병 구하기	9.37
15	사운드 오브 뮤직	9.37
16	살인의 추억	9.37
17	센과 치히로의 행방불명	9.36
18	반지의 제왕: 왕의 귀환	9.36
19	헬프	9.36
20	글래디에이터	9.36
21	클래식	9.35
22	패왕별희	9.35
23	동주	9.35
24	미세스 다웃파이어	9.35
25	아이 캔 스피크	9.35
26	굿바이 마이 프랜드	9.35
27	여인의 향기	9.35
28	주토피아	9.35
29	반지의 제왕: 두 개의 탑	9.35
30	캐스트 어웨이	9.34
31	에이리언 2	9.34

힌트: URL 을 잘 분석해보세요. date!

▼ A. 완성코드

```

import requests
from bs4 import BeautifulSoup

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20180327', 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')

# 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.
        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)

```

▼ (2) 네이버 한국 야구 순위 가져오기 (난이도 : ★★★)

- 네이버 한국 야구 페이지에서 팀 순위 정보를 스크래핑해 출력하세요.
(사용할 URL: <https://sports.news.naver.com/kbaseball/record/index.nhn?category=kbo>)

The screenshot shows the NAVER SPORTS homepage with a blue header bar. Below it, a navigation bar includes links for 스포츠, 야구, 해외야구, 축구, 해외축구, 농구, 배구, 골프, 일반, e스포츠&게임, 오늘의 경기, 라디오, 연재, 랭킹. The main content area is titled "KBO리그 기록 및 순위". A table titled "팀순위" displays the top 10 teams of the 2020 KBO League. The table has columns for 순위 (Rank), 팀 (Team), 경기수 (Games Played), 승 (Wins), 패 (Losses), 무 (Ties), 승률 (Winning Percentage), 게임차 (Game Differential), 연속 (Consecutive Wins/Losses), 출루율 (On-base Percentage), 장타율 (Home Run Percentage), and 최근 10경기 (Recent 10 Games). The top team is NC with a winning percentage of 0.672.

순위	팀	경기수	승	패	무	승률	게임차	연속	출루율	장타율	최근 10경기
1	NC	59	39	19	1	0.672	0.0	2패	0.357	0.475	5승-4패-1무
2	키움	62	37	25	0	0.597	4.0	2승	0.358	0.437	5승-5패-0무
3	두산	60	35	25	0	0.583	5.0	1승	0.370	0.449	6승-4패-0무
4	KIA	58	32	26	0	0.552	7.0	1승	0.349	0.423	5승-5패-0무
5	LG	60	31	28	1	0.525	8.5	1승	0.342	0.416	3승-6패-1무
6	삼성	61	31	30	0	0.508	9.5	1패	0.338	0.406	4승-6패-0무
7	KT	60	30	30	0	0.500	10.0	1패	0.363	0.457	7승-3패-0무
8	롯데	58	28	30	0	0.483	11.0	1패	0.343	0.399	5승-5패-0무
9	SK	61	19	42	0	0.311	21.5	1패	0.315	0.359	4승-6패-0무
10	한화	61	17	44	0	0.279	23.5	1승	0.315	0.340	5승-5패-0무

▼ Q. 이렇게 출력하기

- 물론 순위는 매번 바뀌니 꼭 아래와 똑같은 순서로 보이진 않겠죠? 형식만 참고하세요

```

1 NC
2 키움
3 두산
4 KIA
5 LG
6 삼성
7 KT
8 롯데
9 SK
10 한화

```

▼ A. 완성코드

```
import requests
from bs4 import BeautifulSoup

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86
data = requests.get('https://sports.news.naver.com/kbaseball/record/index.nhn?category=kbo', 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')

# select를 이용해서, 팀순위 표를 가져오기
ranking_table = soup.select('#regularTeamRecordList_table > tr')
# print(ranking_table)

# ranking_table (tr들) 의 반복문을 돌리기
for ranking_info in ranking_table:
    # print('#####', ranking_info)
    rank = ranking_info.select_one('th > strong').text
    name = ranking_info.select_one('td.tm > div > span').text

    print(rank, name)
```

▼ (3) 네이버 한국 야구 순위 가져오기 - 응용 (난이도 : ★★★★☆)

- 네이버 한국 야구 페이지에서 팀 순위 정보를 스크래핑해서, 승률이 0.5 이상인 팀만 출력하세요.
(사용할 URL: <https://sports.news.naver.com/kbaseball/record/index.nhn?category=kbo>)

The screenshot shows the NAVER SPORTS homepage with a blue header bar. Below it, there's a navigation bar with links like '스포츠홈', '야구', '해외야구', '축구', '해외축구', '농구', '배구', '골프', '일반', 'e스포츠&게임', '오늘의 경기', '라디오', '연재', and '랭킹'. A search bar is also present. The main content area is titled 'KBO리그 기록 및 순위' and features a large table for the '팀순위'. The table has columns for '순위' (Rank), '팀' (Team), '경기수' (Games Played), '승' (Wins), '패' (Losses), '무' (Ties), '승률' (Winning Percentage), '게임차' (Game Differential), '연속' (Consecutive Wins/Losses), '출루율' (On-base Percentage), '장타율' (Home Run Percentage), and '최근 10경기' (Recent 10 Games). The table is sorted by winning percentage. A red box highlights the first 10 rows of the table.

순위	팀	경기수	승	패	무	승률	게임차	연속	출루율	장타율	최근 10경기
1	NC	59	39	19	1	0.672	0.0	2패	0.357	0.475	5승-4패-1무
2	키움	62	37	25	0	0.597	4.0	2승	0.358	0.437	5승-5패-0무
3	두산	60	35	25	0	0.583	5.0	1승	0.370	0.449	6승-4패-0무
4	KIA	58	32	26	0	0.552	7.0	1승	0.349	0.423	5승-5패-0무
5	LG	60	31	28	1	0.525	8.5	1승	0.342	0.416	3승-6패-1무
6	삼성	61	31	30	0	0.508	9.5	1패	0.338	0.406	4승-6패-0무
7	KT	60	30	30	0	0.500	10.0	1패	0.363	0.457	7승-3패-0무
8	롯데	58	28	30	0	0.483	11.0	1패	0.343	0.399	5승-5패-0무
9	SK	61	19	42	0	0.311	21.5	1패	0.315	0.359	4승-6패-0무
10	한화	61	17	44	0	0.279	23.5	1승	0.315	0.340	5승-5패-0무

힌트:

조건문의 판별식을 제대로 동작하게 하는게 핵심! 조건문 부분에서 에러가 난다면 에러 메시지를 잘 읽어보고 구글링해보세요.

▼ Q. 이렇게 출력하기

- 물론 승률은 매번 바뀌니 꼭 아래와 똑같은 순서로 보이진 않겠죠? 형식만 참고하세요

1	NC	0.672
2	키움	0.597
3	두산	0.583
4	KIA	0.552
5	LG	0.525
6	삼성	0.508

▼ A. 완성코드

```

import requests
from bs4 import BeautifulSoup

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86
data = requests.get('https://sports.news.naver.com/kbaseball/record/index.nhn?category=kbo', headers=headers)

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

# select를 이용해서, 팀순위 표를 가져오기
ranking_table = soup.select('#regularTeamRecordList_table > tr')
# print(ranking_table)

# ranking_table (tr들) 의 반복문을 돌리기
for ranking_info in ranking_table:
    # print('#####', ranking_info)
    rank = ranking_info.select_one('th > strong').text
    name = ranking_info.select_one('td.tm > div > span').text
    win_rate = ranking_info.select_one('td:nth-child(7) > strong').text

    # 문자열(string) 을 부동소수점형(float)으로 강제 형변환
    if float(win_rate) > 0.5:
        print(rank, name, win_rate)

```

후반 3시간



꼭 DB 설치와 동작 미리 확인해주세요! 튜터님은 튜티들의 설치 및 동작 확인을 도와주세요!

확인 방법

크롬 창에 localhost:27017 이라고 쳤을 때, 아래와 같은 메시지가 나오면 mongoDB가 동작하고 있는 것입니다.

It looks like you are trying to access MongoDB over HTTP on the native driver port.

▼ Windows에서 위 메시지가 안 뜰 때 빠른 해결방법

1. 윈도우 키 + R 을 누른 후 cmd 를 입력하고 엔터를 누릅니다. 까만 터미널창이 뜨면, 그리고 다음 명령어를 복사/붙여넣기하여 입력합니다.

```
mongod
```

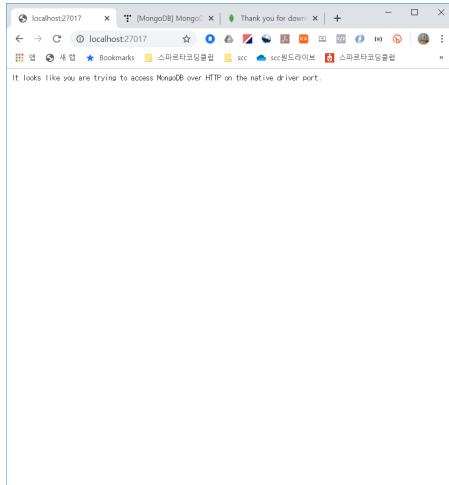
```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\sos89\mongod
2018-06-09T09:59:12.689+0700 I CONTROL  [initandlisten] MongoDB starting : pid=7504 port=27017 dbpath=C:\data\db\ 64-bit
host=BEOULN-PARK
2018-06-09T09:59:12.689+0700 I CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2018-06-09T09:59:12.690+0700 I CONTROL  [initandlisten] db version v3.6.5
2018-06-09T09:59:12.690+0700 I CONTROL  [initandlisten] git version: a20e0d3e3a174162052ff90913bc2ca9a839d618

```

2. 브라우저에서 localhost:27017 주소로 이동한 뒤 다음과 같은 화면이 뜨는 것을 확인하면 실행 완료!



3. 터미널 창을 종료하지 말아주세요. 터미널 창을 끄면 mongoDB 실행도 종료됩니다.

[월수/화목반] : 체크인 & 출석체크



튜터님은 체크인과 함께 출석 체크(링크)를 진행해주세요!

스파르타코딩클럽 출석체크

<http://spartacodingclub.shop/attendance>

▼ "15초 체크인" 진행합니다

- 튜터는 타이머를 띄워주세요! ([링크](#))
- 본인의 감정상태와 오늘 있었던 소소한 일을 공유하는 시간입니다.



배우고 적용하기 - 후반부



5분 꿀팁 - CRUD

▼ CRUD 란?

CRUD(크루드, 씨알유디)라고 읽습니다. 마법 주문이름 같기도 하죠?

- 기본적인 데이터 처리 기능인 **Create, Read, Update, Delete**의 두문자를 따서 **CRUD**라고 합니다. 대부분의 소프트웨어는 이 기능을 가지고 있고요. UI가 갖추어야 할 기본 기능 단위로 CRUD를 묶어 이야기합니다.
- Create (생성) : 예. 사용자가 게시글 쓰기
- Read (읽기) : 예. 게시글 보기
- Update(갱신, 업데이트) : 예. 게시글 수정
- Delete (삭제) : 예. 게시글 삭제
- 여러분들이 나중에 소프트웨어 기능을 개발할 때 CRUD가 내가 구현해야하는 기본 세트구나! 하고 생각하시면 좋겠죠?

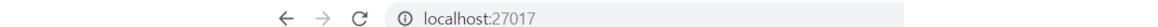
[한 걸음 더!]

- API를 설계하는 방식 중 하나인 RESTful API에서는 HTTP method 와 CRUD 를 하나 하나씩 매핑해서 쓰기도 해요. 우리가 Read 기능은 GET 방식을 쓰자! 라고 한 것처럼요.

[1시간] DB 사용하기 - pymongo로 제어하고 Robo 3T로 확인하기

▼ 13) DB 설치 확인

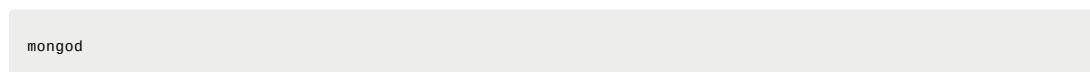
 먼저, 각자 설치해온 DB가 잘 작동하는지 확인합니다. 크롬 창에 localhost:27017 이라고 쳤을 때, 아래와 같은 화면이 나오면 mongoDB가 돌아가고 있는 것입니다.



It looks like you are trying to access MongoDB over HTTP on the native driver port.

▼ Windows에서 위 메시지가 안 뜰 때 빠른 해결방법

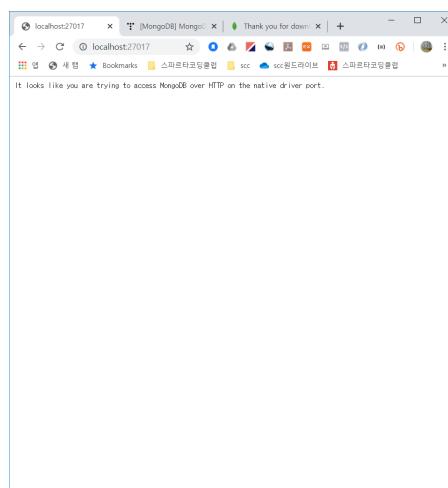
- 원도우 키 + R 을 누른 후 cmd 를 입력하고 엔터를 누릅니다. 까만 터미널 창이 뜨면, 그리고 다음 명령어를 복사/붙여넣기하여 입력합니다.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\scs89>mongod
2018-06-09T09:59:12.689-0700 | CONTROL  [initandlisten] MongoDB starting : pid=7504 port=27017 dbpath=C:\data\db\ 64-bit
host=BEOMUN-PARK
2018-06-09T09:59:12.689-0700 | CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2018-06-09T09:59:12.690-0700 | CONTROL  [initandlisten] db version v3.6.5
2018-06-09T09:59:12.690-0700 | CONTROL  [initandlisten] git version: a20ec03e3a174162052ff99913bc2ca9a839d618
```

- 브라우저에서 localhost:27017 주소로 이동한 뒤 다음과 같은 화면이 뜨는 것을 확인하면 실행 완료!



- 터미널 창을 종료하지 말아주세요. 터미널 창을 끄면 mongoDB 실행도 종료된답니다.

▼ 14) [👏튜터와 함께] robo 3T 준비하기

- robo 3T의 역할



mongoDB라는 프로그램은 참 특이한 친구예요. 눈으로 보이지 않는답니다.
유식한 말로, 그래픽 유저 인터페이스(=GUI)를 제공하지 않는다고 표현합니다.

데이터를 저장했는데 눈으로 보이진 않고.. 답답하겠죠?

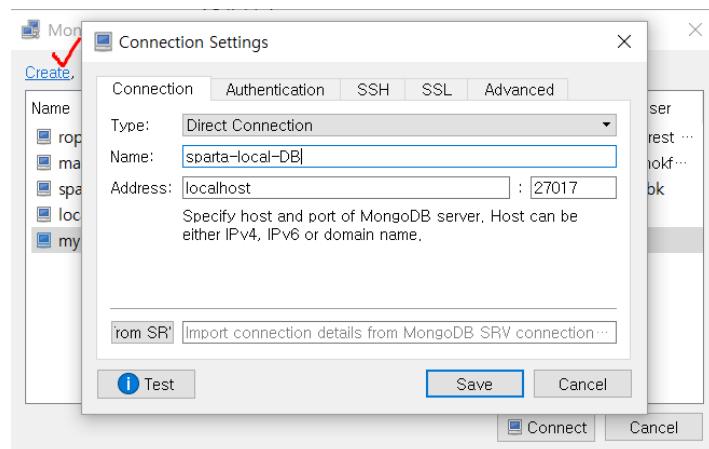
그래서 **DB내부를 살펴보기 위한 프로그램**을 따로 설치해야해요.
→ 이것이 바로 robo3T의 역할!

(참고) 우리 눈에는 안보이지만(=GUI는 없지만) 컴퓨터에서 돌아가는 프로그램들은 무척 많으니, 너무 놀라지 마세요~!

- robo3T 세팅하기



아래처럼 준비해서 robo3T에서 DB 볼 세팅을 미리 해둡니다!
(sparta-local-DB는 아무 이름이나 입력해도 됩니다)



- db, collection, documents(각 데이터들을 지칭)를 확인 가능합니다.



지금은 System, config 외엔 아무것도 없죠?
조금 이따 데이터를 넣으면 아래처럼 보일거예요!

Key	Type
(1) ObjectId("5f0fbf30da1ce10c2349c31c")	Object
_id	ObjectId
name	String
age	Int32
(2) ObjectId("5f0fbf30da1ce10c2349c31d")	Object
_id	ObjectId
name	String
age	Int32
(3) ObjectId("5f0fbf30da1ce10c2349c31e")	Object
_id	ObjectId
name	String
age	Int32
(4) ObjectId("5f0fc07ff11fb45d2c4e0495")	Object
_id	ObjectId
name	String
age	Int32
(5) ObjectId("5f0fc07ff11fb45d2c4e0496")	Object
_id	ObjectId
name	String
age	Int32
(6) ObjectId("5f0fc07ff11fb45d2c4e0497")	Object
_id	ObjectId
name	String
age	Int32

▼ 15) 들어가기 전에 : DB의 두 가지 종류

Database에는, 크게 두 가지 종류가 있습니다.



RDBMS(SQL)

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

ex) PostgreSQL(포스트그레 애스큐엘(시퀄)), MySQL(마이 애스큐엘(시퀄)) 등

No SQL

딕셔너리 형태로 데이터를 저장해두는 DB입니다. 같은 값을 가진 데이터들도 자유롭게 저장이 가능합니다. 자유로운 형태의 데이터 적재에 유리한 대신, 일관성이 부족할 수 있습니다.

ex) MongoDB

▼ 16) [튜터와 함께] pymongo로 mongoDB 조작하기

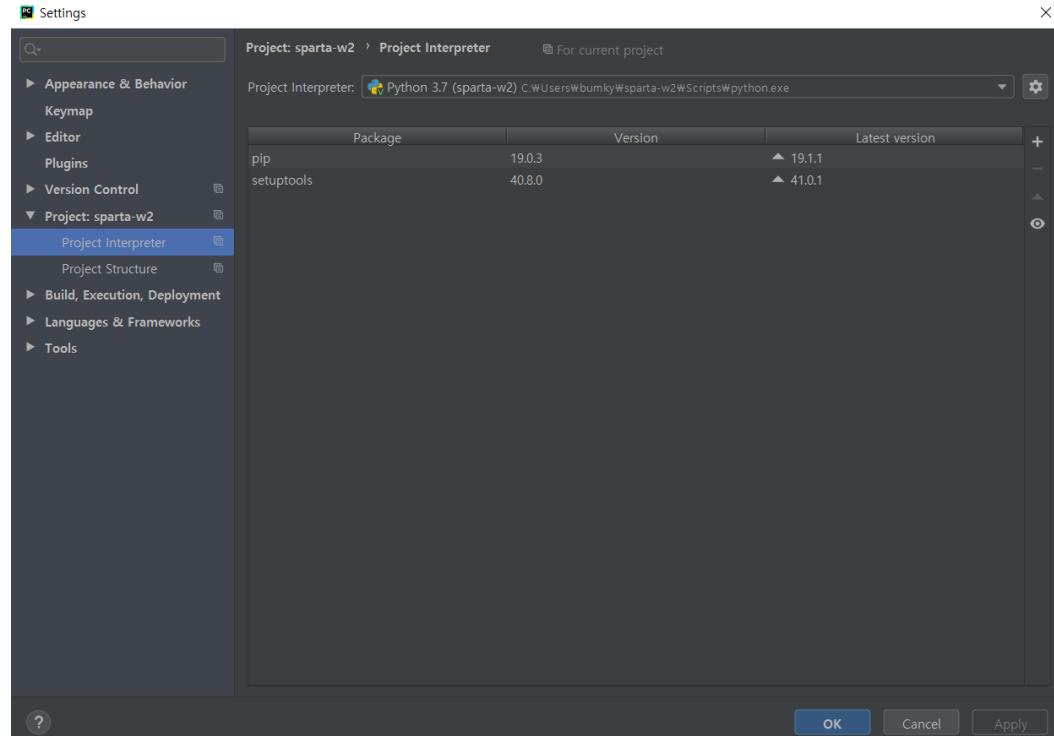
- 패키지 설치

- `pymongo` 는 MongoDB 를 쉽게 조작하기 위한 라이브러리입니다.
- 아래 화면에서 `pymongo` 를 검색해 설치해주세요. 앞서 requests, bs4 패키지 설치한 것과 동일합니다.

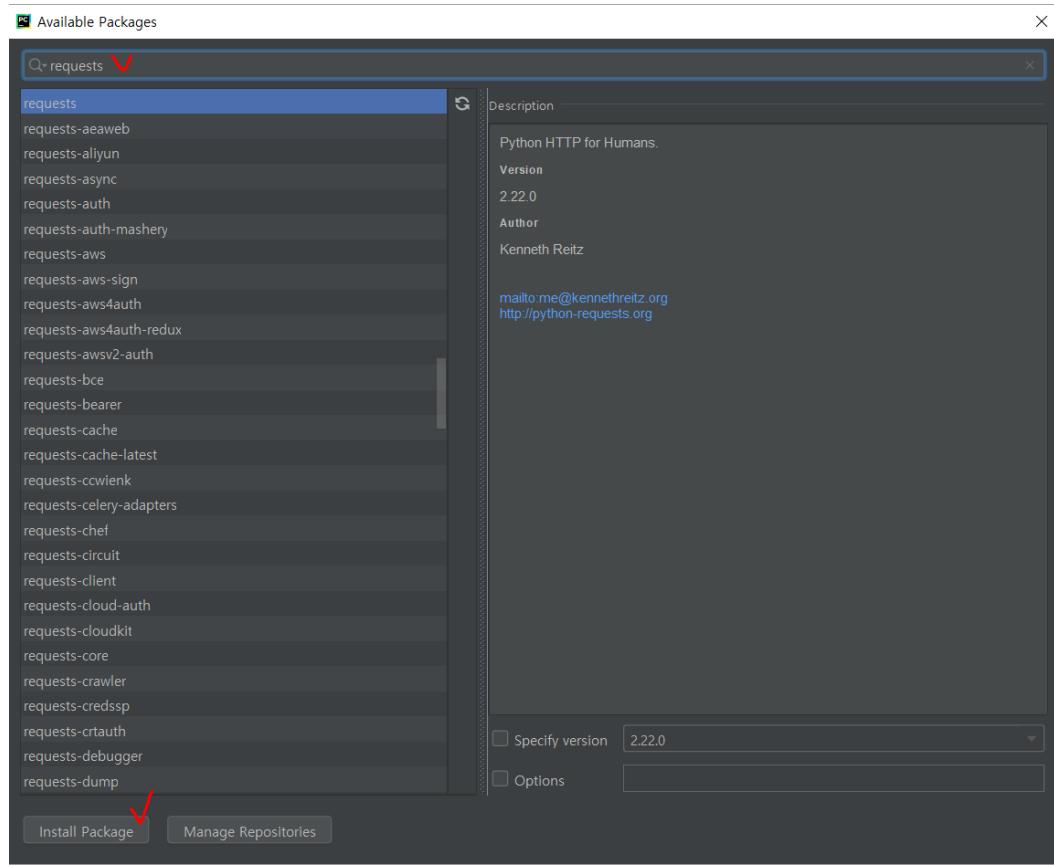
▼ Pycharm 화면

- project interpreter 로 접근해서 + 버튼 누르기

Windows : file → setting → project interpreter
Mac : preference → project interpreter로 접근



- project interpreter 화면에서 + 버튼을 누르면 아래 창이 뜹니다!



- Pycharm에서 `db_practice.py`라는 이름으로 파일을 새롭게 만들고, 아래 코드를 따라하며 배워볼게요. 주석을 적절하게 사용해 보세요.

[💻 코드 pymongo 연습 01] DB연결하기 & 데이터 생성하기(Create)

```
from pymongo import MongoClient # pymongo를 임포트 하기(패키지 설치 먼저 해야겠죠?)

client = MongoClient('localhost', 27017) # mongoDB는 27017 포트로 돌아갑니다.
db = client.dbsparta # 'dbsparta'라는 이름의 db를 만듭니다.

# MongoDB에 insert 하기

# 'users'라는 collection에 데이터를 생성합니다.
db.users.insert_one({'name': '덤블도어', 'age': 116})
db.users.insert_one({'name': '맥고나걸', 'age': 85})
db.users.insert_one({'name': '스네이프', 'age': 60})
db.users.insert_one({'name': '해리', 'age': 40})
db.users.insert_one({'name': '허마이오니', 'age': 40})
db.users.insert_one({'name': '론', 'age': 40})
```

[💻 코드 pymongo 연습 02-1] DB 저장된 값을 조회하기 (Read)

```
from pymongo import MongoClient # pymongo를 임포트 하기(패키지 인스톨 먼저 해야겠죠?)

client = MongoClient('localhost', 27017) # mongoDB는 27017 포트로 돌아갑니다.
db = client.dbsparta # 'dbsparta'라는 이름의 db를 만듭니다.

# MongoDB에서 데이터 모두 보기
all_users = list(db.users.find({}))

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)
```

[💻 코드 pymongo 연습 02-2] DB 저장된 특정값 조회하기 (Read)

```
# MongoDB에서 특정 조건의 데이터 모두 보기
same_ages = list(db.users.find({'age': 40}))

for user in same_ages: # 반복문을 들며 모든 결과값을 보기
    print(user)

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

# 그 중 특정 키 값을 빼고 보기
user = db.users.find_one({'name': '덤블도어'}, {'_id': False})
print(user)
```

[💻 코드 pymongo 연습 03] 값 업데이트하기 (Update)

- `db.people.update_many(값조회_조건, { '$set': 바꿀_값 })`

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

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

[💻 코드 pymongo 연습 04] 삭제하기 (Delete)

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

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

- 실제 개발할 때에는, 데이터를 삭제하면 복구가 어렵기 때문에 '삭제' 기능이라도 실제 DB에서는 데이터 자체를 삭제하는 것보다 '사용하지 않음'으로 처리해두는 경우가 많답니다.

▼ 17) 요약 : pymongo 사용한 CRUD



5분꿀팁에서 배운 CRUD 기억나시죠?

기능 기본 단위! Create(생성), Read(조회), Update(업데이트), Delete(삭제)

딱 네 가지 기능 어떻게 쓰는지 정리하면 아래와 같습니다.

```
# Create(생성)
user = {'name': '론', 'age': 40}
db.users.insert_one(user)

# Read(조회) - 한 개 값만
user = db.users.find_one({'name': '론'})

# Read(조회) - 여러 값 (_id 값은 제외하고 출력)
same_ages = list(db.users.find({'age': 40}, {'_id': False}))

# Update(업데이트)
db.users.update_one({'name': '덤블도어'}, {'$set': {'age': 116}})

# Delete(삭제)
db.users.delete_one({'name': '론'})
```

[0.5시간] 스크래핑 한 결과를 mongoDB에 저장하기

▼ 18) [💡튜터와 함께] insert 연습하기 - 웹 스크래핑 결과를 DB 저장

- 준비하기

- week03 폴더에 `mongo_practice.py` 파일을 만들고 아래 코드에서 시작해봅시다! (복사+붙여넣기)

▼ [💻 코드 pymongo 응용 00-1]

```

import requests
from bs4 import BeautifulSoup

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200716', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만듦
# soup이라는 변수에 "파싱 용이해진 html"이 담긴 상태가 됨
# 이제 코딩을 통해 필요한 부분을 추출하면 된다.
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.
        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)

```

[■ 코드 pymongo 응용 01-1] pymongo 기본 세팅

```

import requests
from bs4 import BeautifulSoup
from pymongo import MongoClient # pymongo를 임포트 하기(패키지 인스톨 먼저 해야겠죠?)

client = MongoClient('localhost', 27017) # mongoDB는 27017 포트로 돌아갑니다.
db = client.dbsparta # 'dbsparta'라는 이름의 db를 만듭니다.

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200716', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만듦
# soup이라는 변수에 "파싱 용이해진 html"이 담긴 상태가 됨
# 이제 코딩을 통해 필요한 부분을 추출하면 된다.
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.
        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)

```

[■ 코드 pymongo 응용 01-2] 도큐먼트 만들어 하나씩 insert 하기

```

import requests
from bs4 import BeautifulSoup
from pymongo import MongoClient # pymongo를 임포트 하기(패키지 인스톨 먼저 해야겠죠?)

client = MongoClient('localhost', 27017) # mongoDB는 27017 포트로 돌아갑니다.
db = client.dbsparta # 'dbsparta'라는 이름의 db를 만듭니다.

# 타겟 URL을 읽어서 HTML를 받아오고,
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36'}
data = requests.get('https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=pnt&date=20200716', headers=headers)

# HTML을 BeautifulSoup이라는 라이브러리를 활용해 검색하기 용이한 상태로 만듦
# soup이라는 변수에 "파싱 용이해진 html"이 담긴 상태가 됨
# 이제 코딩을 통해 필요한 부분을 추출하면 된다.
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.
        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)
        doc = {
            'rank': rank,
            'title': title,
            'star': star # DB에는 숫자처럼 생긴 문자열 형태로 저장됩니다.
        }
        db.movies.insert_one(doc)

```

▼ 19) 🎨 find, update 연습하기 (delete는 연습 안할게요!)

- week03 폴더에 `mongo_practice02.py` 파일을 만들고 아래 코드에서 시작해봅시다! (복사+붙여넣기)

▼ [💻 코드 pymongo 응용 00-2]

```

from pymongo import MongoClient

client = MongoClient('localhost', 27017)
db = client.dbsparta

## 코딩 할 준비 ##

```

- (1) 영화 제목 '월-E'의 평점을 가져오기

▼ Q. 이렇게 되면 완성



9.41
Process finished with exit code 0

▼ A. 완성 코드

```

from pymongo import MongoClient

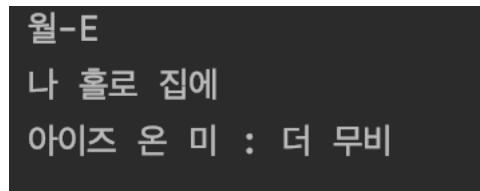
client = MongoClient('localhost', 27017)
db = client.dbsparta

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

```

- (2) '월-E'의 평점과 같은 평점의 영화 제목들을 가져오기

▼ Q. 이렇게 되면 완성



월-E
나 홀로 집에
아이즈 온 미 : 더 무비
런 더 런

▼ A. 완성 코드

```

from pymongo import MongoClient

client = MongoClient('localhost', 27017)
db = client.dbsparta

target_movie = db.movies.find_one({'title': '월-E'})

```

```

target_star = target_movie['star']

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

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

```

- (3) '월-E'의 평점과 같은 평점의 영화 제목들의 평점을 0으로 만들기

▼ Q. 이렇게 되면 완성 (robo3T에서)

▼ A. 완성 코드

```

from pymongo import MongoClient

client = MongoClient('localhost', 27017)
db = client.dbsparta_ninework

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

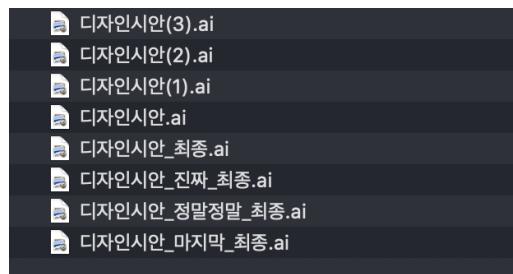
db.movies.update_many({'star': target_star}, {'$set': {'star': '0'}})

```

[0.5시간] 버전 관리 Git 사용하기

▼ 20) Git & Github이란

- 버전 관리가 안되어 힘든 경험 다들 있지 않으신가요? 아래처럼 말이죠!



- **Git** 은 작업 기록을 남기고 이력을 추적해서 코드를 손쉽게 관리하도록 도와줍니다. 버전 관리도 매우 간편합니다.
 - 어떤 버전부터 버그가 생긴 걸까? 어떤 내용이 변경되었는지 한 눈에 보고 싶다!
 - 여러 사람 / 컴퓨터에서 코드 작업을 동시에 하고 있다면 변경된 내역을 어떻게 합칠까?
 - 이력 추적하기 예시 ↪ [현법 개정안 비교](#)
- **Github** 은 작업 기록과 파일을 저장하는 Git 원격 저장소를 지원해주는 사이트입니다.
 - 그 외에도 최근 트렌드인 오픈소스 찾기, 프로젝트 issue 관리처럼 다양한 부가 기능을 제공하고 있어요. (비슷한 곳으로는 Gitlab, bitbucket 등이 있습니다.)

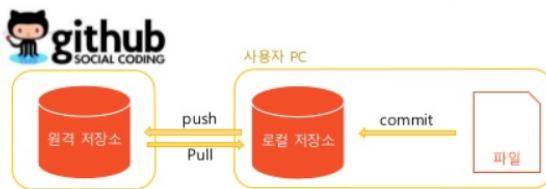
▼ 21) Git의 commit / pull / push 란



혼자 개발 할 때에도 git을 쓰면 무척 편리합니다.

만약 작성 중이던 코드가 한참 잘못돼서 일주일 전 버전으로 돌리고 싶을 때, git이 없다면 난감하겠죠?

- 우리는 앞으로 주로 '내 컴퓨터(로컬 저장소)에 작업 내역을 반영하고, 원격 저장소(예. Github)에 작업 내역을 올리기' 를 많이 사용해볼거에요. 이제 Git 에서 가장 많이 쓰이는 commit / pull / push 를 살펴볼게요!



1. 내 로컬 저장소에 작업 내역을 반영하고 → `commit`

- commit 앞에 파일을 새로 추적하는 `add` 과정이 있습니다. 우리가 쓰는 Github Desktop 처럼 GUI 툴을 사용하게 되면, 보통 commit 작업에 포함되어 있습니다.

2. 원격 저장소에 작업 내역을 업로드하는 것 → `push`

- 보통은 여러 사용자가 올린 것이라도 작업 내역에 겹치는 게 없다면 자동으로 합쳐줍니다.
- 여러 사용자가 하나의 파일의 동일한 부분을 고쳤다면, 어떤 부분을 어떻게 반영해야 할까요? Git이 충돌(conflict)났다고 알려주고 사용자에게 직접 수정하라고 알려줍니다.

3. 원격 저장소 작업 내역을 내 로컬 저장소로 가져오는 것 → `pull`

- 다른 사람의 작업내역을 가져오거나, 다른 컴퓨터에서 작업한 내용을 가져올 수 있습니다.
- 보통 나의 작업 내역을 `commit` 하고, `pull` 해으면 좋습니다. 그렇지 않으면 다른 사람의 작업 내역으로 내 작업 내역이 덮어쓰기가 되어버립니다.
- 임시로 파일의 작업 내역을 보관해두는 `stash` 도 있습니다. 하지만 우리는 처음 배우니까 요건 나중에 git에 익숙해지면 써 보세요!

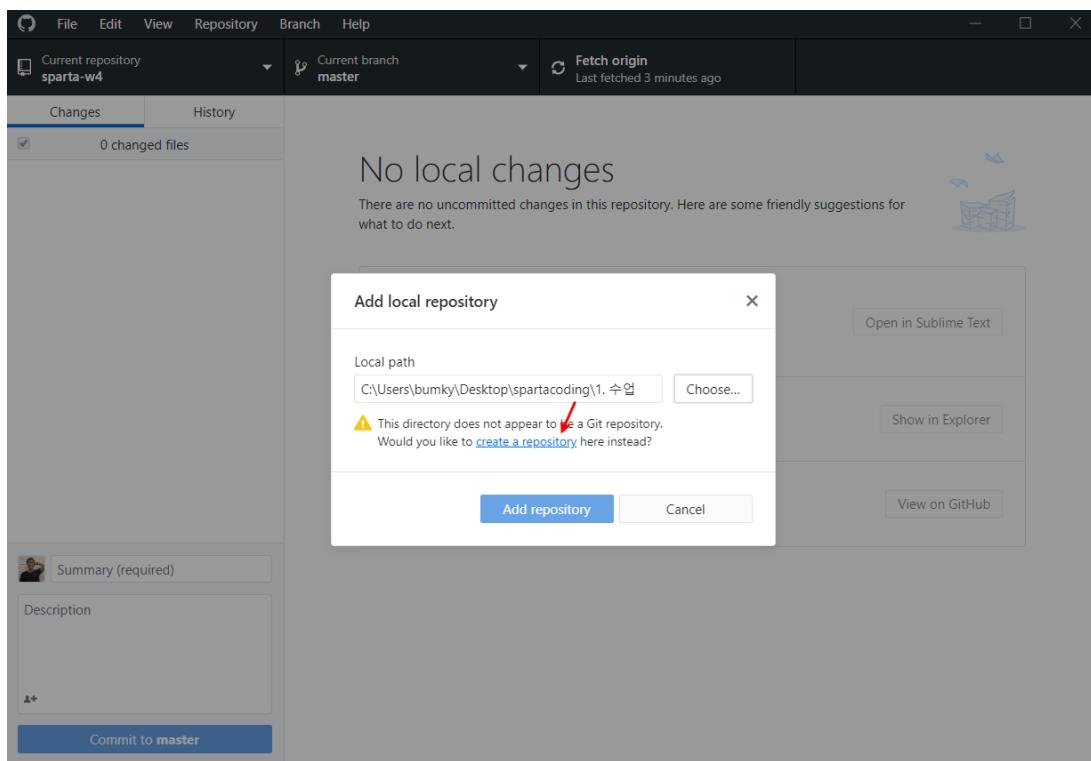
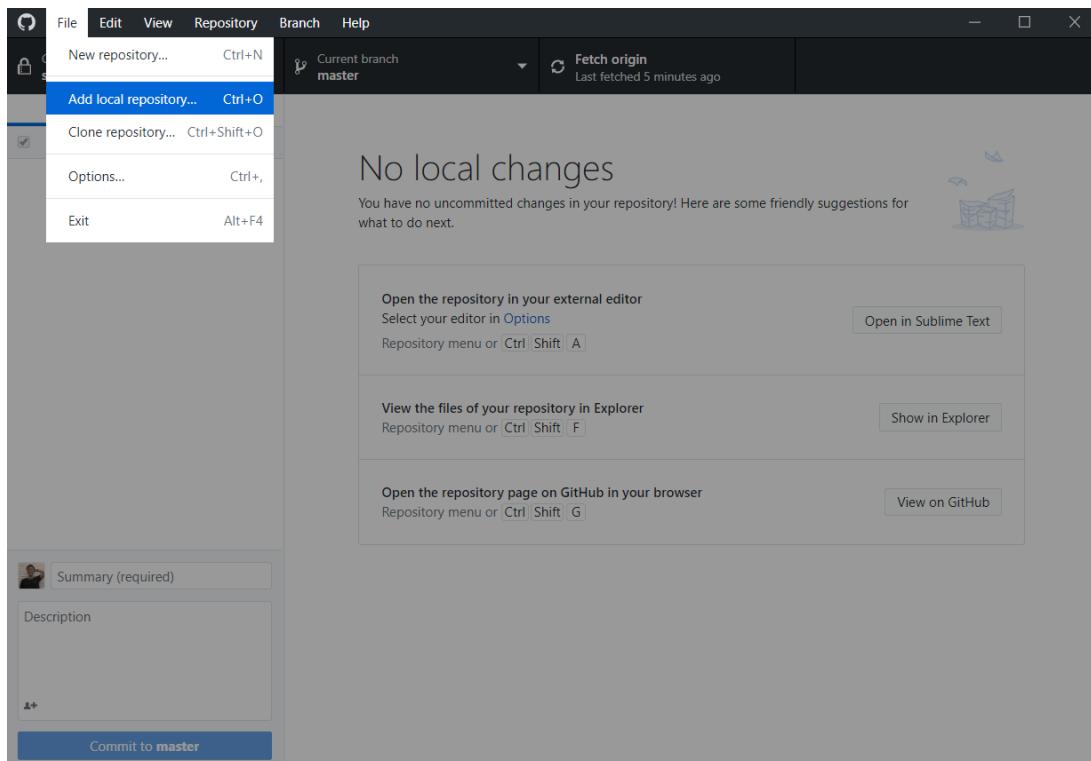
▼ 22) [👉튜터와 함께] 숙제를 git에 올려보기 - Github Desktop 사용

- Github Desktop이란? 그래픽 인터페이스(클릭, 클릭 / GUI)로 Git을 쓸 수 있게 하는 프로그램!

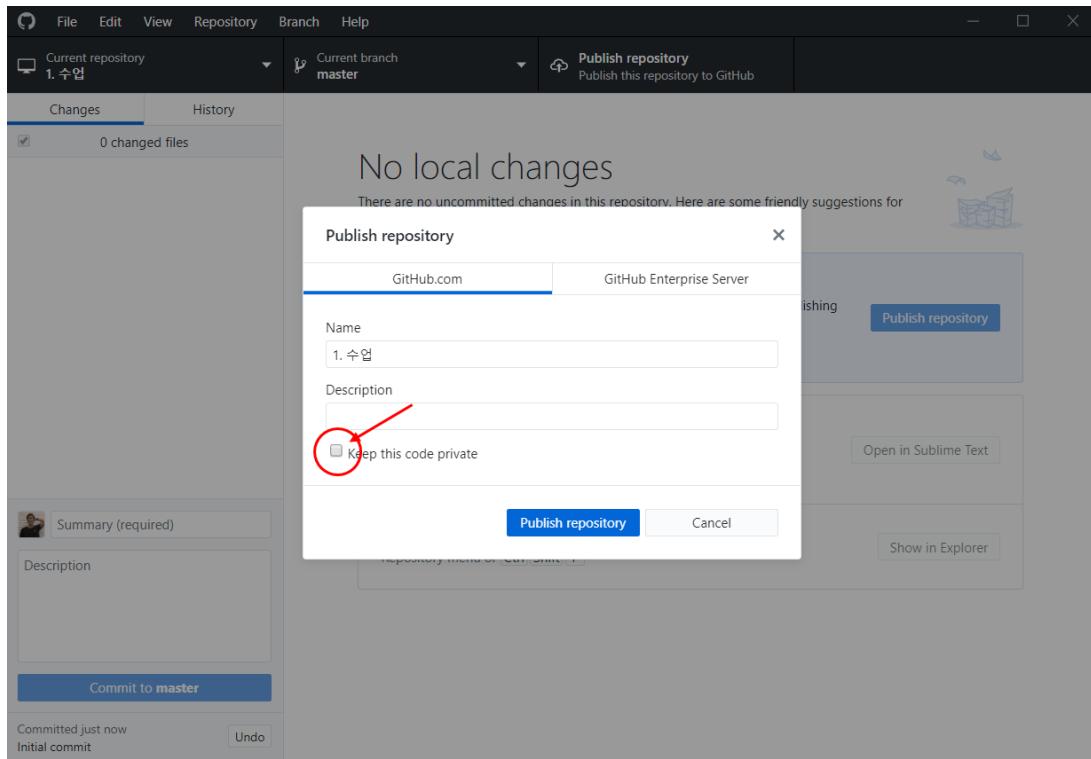
1. 내 로컬 저장소(Repository) 만들기

- **잠깐!** homework 폴더에 파일이 있는지 확인해주세요.
 - 숙제를 아직까지 못해서 homework에 아무 파일도 없으신 분들은 오늘 수업시간에 한 아무 파일이나 복사 - 붙여넣기해서 homework 에 파일을 넣어주세요!
- Github Desktop 을 켜고 로그인 후, add local repository 클릭, 바탕화면에 있는 sparta/homework 폴더를 클릭하고, 파란색 글자/버튼을 따라 누르면 repository가 완성됩니다.
- 기존에 존재하던 폴더(homework)를 git 이 추적(tracking)할 수 있게 설정했다고 생각하면 됩니다. (homework 폴더 안에 git 설정 내용을 담고 있는 `.git` 이라는 폴더가 생성되었을 거에요.)

▼ 화면



반드시 'keep this code private'을 해제해주세요!
공개되어 있어야 투터도 숙제 확인이 가능하답니다.



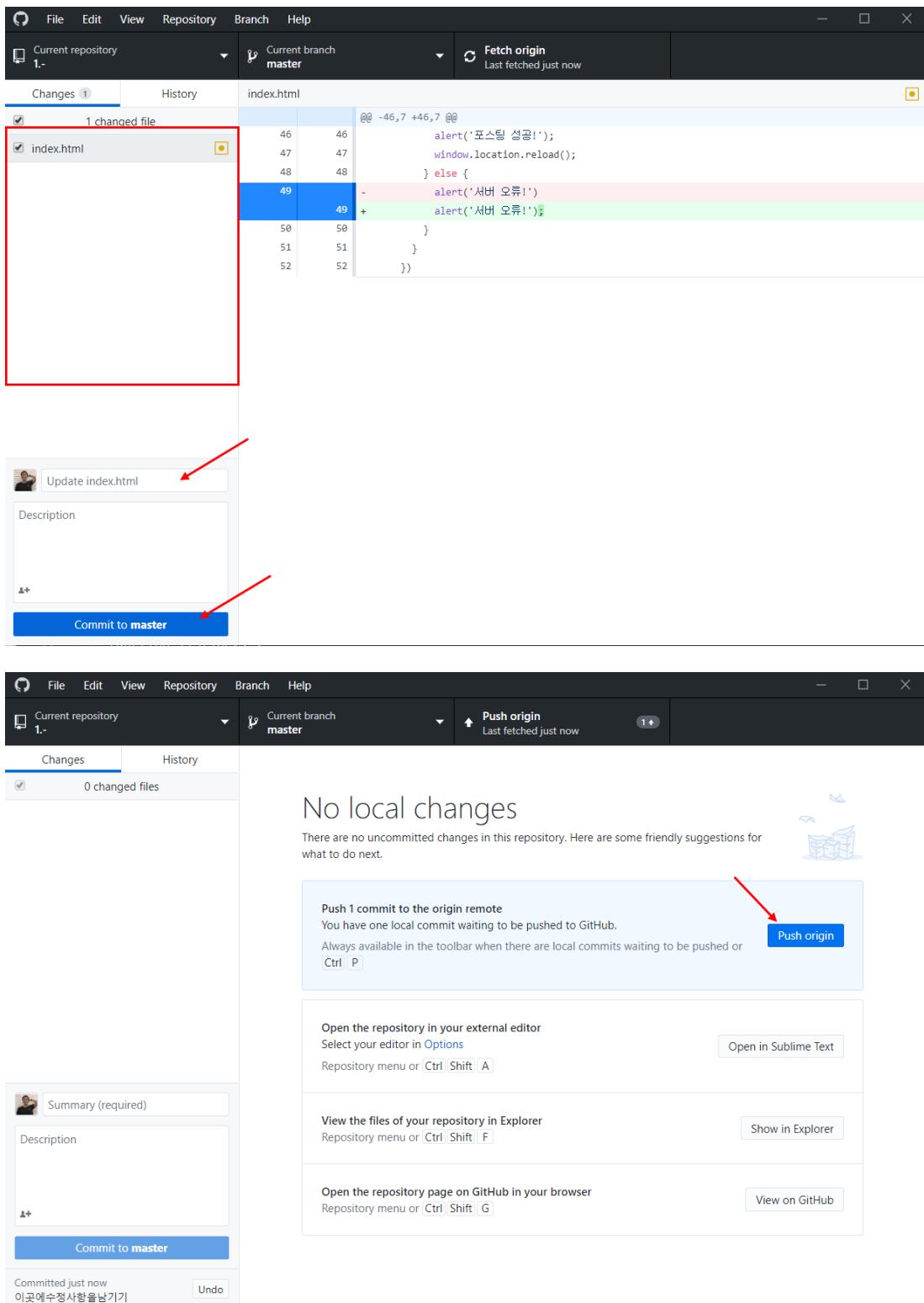
2. github.com에 로그인 후, 원격 저장소(repository)가 생성된 것 확인

▼ 화면

3. 코드를 수정하고, `commit` 하고 `push` 해보기

 우리는 작업 내역을 잘 관리하기 위해 Git을 사용하는 거니까,
작업 내역을 알기 쉽게 commit message 를 적어주어야겠죠? (변수/함수 이름 짓기(naming)처럼 중요하답니다!)

▼ 화면



4. github.com 에 내 원격 저장소 페이지에서 코드가 바뀐 것 확인

▼ 화면

No description, website, or topics provided.

Manage topics

3 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

bumkyulee 이곳에수정사항을남기기
Initial commit 4 minutes ago
.gitattributes 4 minutes ago
index.html 1 minute ago

Latest commit 348894d 1 minute ago

Add a README

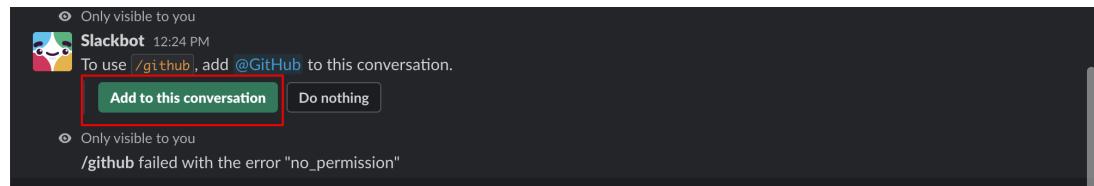
▼ 23) [💡튜터와 함께] Github 수정 내역을 Slack 에 알림오게 만들기

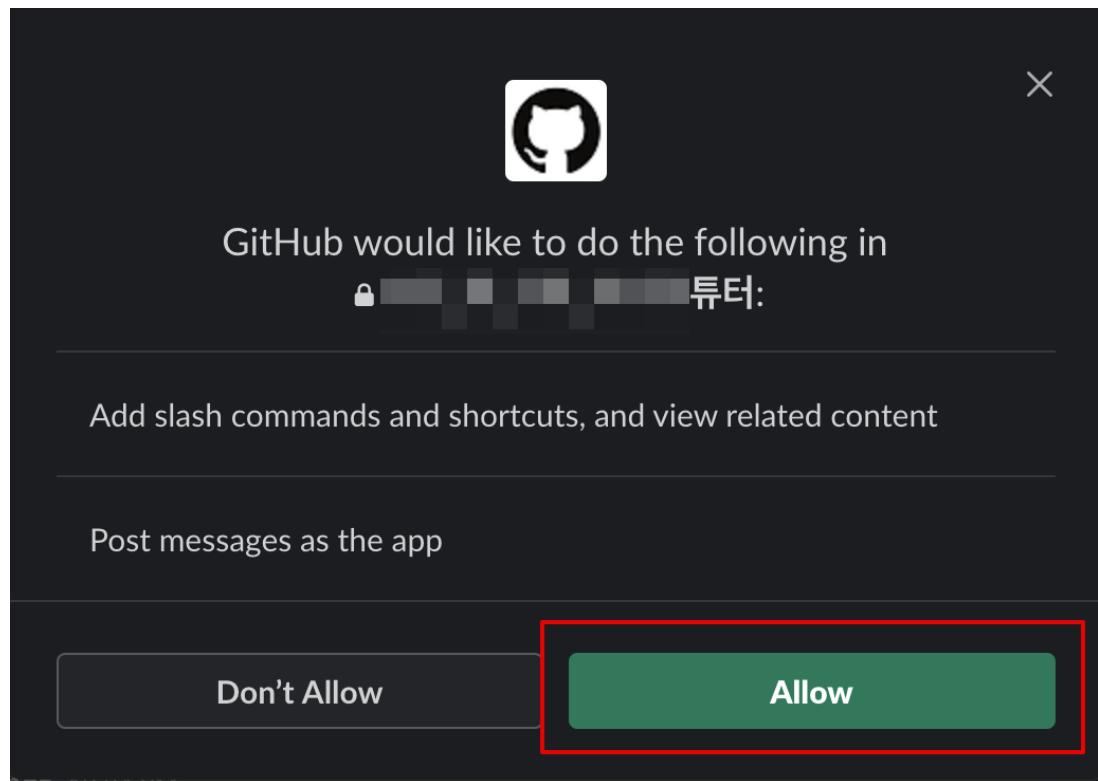
- 우리반 슬랙 채널에 나의 원격 저장소(Github repository) 수정내역을 알림오게 설정해볼게요.
- 원리는 슬랙에 있는 github 봇을 사용하는 겁니다. 슬랙 채널 메시지창에 아래처럼 입력해보세요.

```
/github subscribe 내_github_사용자명/repository명
```

- 만약 아래와 같은 메시지가 나온다면 초록 버튼 클릭!

▼ 화면





- 그리고 github 에 권한을 주세요.

▼ 화면보며 따라하기





Finish connecting your GitHub account to your Slack account
on the [spartacodingclub.slack.com](#) workspace

This will connect your accounts so that you can use Slack to view rich previews for private GitHub links, open and close issues, and other features that depend on your access to GitHub. [Learn more](#)

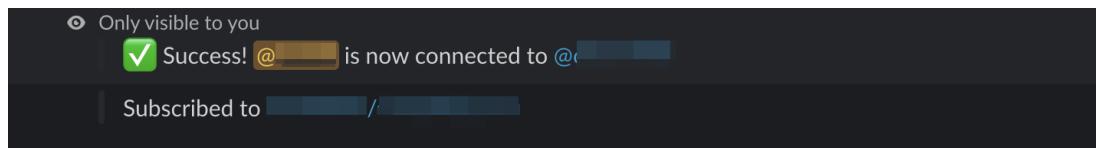
[Connect GitHub account](#)

클릭

GitHub Inc. ©2020

[Terms / Privacy](#)

- 아래와 같이 완료메시지가 뜨면 성공



- 슬랙 메시지 창에 `/github` 이라고 쓰고 엔터를 누르면 더 많은 github 슬랙 앱의 명령어들을 보실 수 있어요. 더 이상 알림이 오지 않게 하는 것(unsubscribe)도 가능!

▼ 화면

Only visible to you

GitHub APP 12:26 PM

Need some help with `/github?`

Subscribe to notifications for a repository:
`/github subscribe owner/repository`

Unsubscribe from notifications for a repository:
`/github unsubscribe owner/repository`

Subscribe to notifications for all repositories in an organization:
`/github subscribe owner`

Unsubscribe from notifications for an organization:
`/github unsubscribe owner`

Subscribe to additional features and adjust the configuration of your subscription ([Learn more](#)):
`/github subscribe owner/repository reviews,comments`

Unsubscribe from one or more subscription features:
`/github unsubscribe owner/repository commits`

Create required-label. Issues, Comments, PRs without that label will be ignored:
`/github subscribe owner/repository +label:my-label`

Remove required-label:
`/github unsubscribe owner/repository +label:my-label`

List all active subscriptions in a channel:
`/github subscribe list`

List all active subscriptions with subscription features:
`/github subscribe list features`

Close an issue:
`/github close [issue link]`



와, mongoDB 데이터베이스에 저장하면 정보를 언제나 가져다가 쓸 수 있겠군요!
Git 을 사용하니 내 코드도 관리하기 쉬워졌고요.
신난다~!

😎 문제뱅크

- 오늘은 쉬어 갈게요!
- 만약 같은 반 동료 튜티들을 다 도와주고도 실습 시간이 남는다면, '공통 숙제 - 내 프로젝트에 사용할 정보 찾기' 를 해보세요!

💡 소화 타임



소화타임 = 복습 & 숙제 시작 타임!

인생은 실전! 튜터에게 질문하면서 숙제를 시작해보세요.

숙제 설명



달달한 맛 vs 짭짤한 맛 중 하나만 골라서 하시면 됩니다! :-)
숙제는 되도록 수업 D-1 까지 하기!

1. 공통 숙제 - 내 프로젝트에 사용할 정보 찾기

▼ 1주차에 내 프로젝트 아이디어 상상해봤었죠?

1. 프로젝트 이름
2. 기능 설명
3. 참고한 프로젝트/웹사이트가 있다면 링크
4. (선택)프로젝트 생김새

• 내 프로젝트에 데이터가 필요하다면 어떤 사이트에서 정보를 얻을 수 있을지 사이트 링크를 조사해주세요. 스크래핑이 될 수도 있고, openAPI 가 될 수도 있어요. 그리고 링크들을 슬랙에 메시지로 남기기!

- 예를 들어,
 - '한국의 100대 명산' 알려주는 사이트를 만들고 싶다면 산의 정보를 어디선가 가져와야겠죠?
 - 100대 명산 리스트 : <https://100mountain.tistory.com/431>
 - 등산 후에 파전 먹을 맛집 정보를 블로그에서 검색! 네이버 블로그 검색
 - <https://developers.naver.com/docs/search/blog/>
 - 산정보 공공 API (국내 소재 3,368개 설명) : <https://www.data.go.kr/data/15058662/openapi.do>
- 한 걸음 더
- 정말 정말 더 해보고 싶으신 분들은 조사한 사이트의 웹 스크래핑을 시도해보세요.
(참고. 웹 스크래핑하는 것보다 API를 사용하는게 훨씬 더 편하기도 하고, 정확한 정보를 얻을 수 있는 경우도 많습니다!)

[숙제 - 달달한 맛 🍫] 지니뮤직의 1~50위 곡의 정보를 스크래핑 (30분 예상)



아래 코드에 빈 칸을 채워서 순위 / 곡 제목 / 가수를 스크래핑해 출력하면 됩니다.
<https://www.genie.co.kr/chart/top200?ditc=D&rtm=N&ymd=20200713>

▼ 💻 코드! 여기서부터 시작하세요

```
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 Safari'
# 아래 빈 칸('')을 채워보세요
data = requests.get('', headers=headers)
soup = BeautifulSoup(data.text, 'html.parser')

trs = soup.select('#body-content > div.newest-list > div > table > tbody > tr')
# 아래 빈 칸('')을 채워보세요
for tr in trs:
    rank = tr.select_one('').text[0:2].strip()
    title = tr.select_one('').text.strip()
    artist = tr.select_one('').text
    print(rank, title, artist)
```

▼ 완성 모습

- 1 여름 안에서 (Covered by 싹쓰리) (Feat. 황광희) 싹쓰리 (유두래곤 & 린다G & 비룡)
- 2 아직 너의 시간에 살아 이수현
- 3 How You Like That BLACKPINK
- 4 마리아 (Maria) 화사 (Hwa Sa)
- 5 Summer Hate (Feat. 비) 지코 (ZICO)
- 6 PLAY (Feat. 창모) 청하
- 7 Downtown Baby 블루 (BL00)
- 8 보라빛 밤 (pporappippam) 선미
- 9 Apple 여자친구 (GFRIEND)
- 10 에잇 (Prod. & Feat. SUGA of BTS) 아이유 (IU)
- 11 아로하 조정석
- 12 Monster 레드벨벳-아이린 & 슬기
- 13 Into the I-LAND 아이유 (IU)
- 14 Dolphin 오마이걸 (OH MY GIRL)
- 15 사랑하게 될 줄 알았어 전미도
- 16 살짝 설렜어 (Nonstop) 오마이걸 (OH MY GIRL)
- 17 10억뷰 (Feat. MOON) 세훈 & 찬열
- 18 어떻게 지내 (Prod. by VAN.C) 오반
- 19 MORE & MORE TWICE (트와이스)
- 20 Don't Start Now Dua Lipa
- 21 Dance Monkey Tones And I
- 22 흔들리는 꽃들 속에서 네 삼푸향이 느껴진거야 장범준
- 23 METEOR 창모 (CHANGMO)
- 24 Memories Maroon 5
- 25 Blueming 아이유 (IU)
- 26 아무노래 지코 (ZICO)
- 27 너에게 난, 나에게 넌 미도와 파라솔
- 28 썸 타간 월 타 백아연
- 29 Left & Right 세븐틴 (Seventeen)
- 30 시작 가호 (Gaho)
- 31 늦은 밤 너의 집 앞 골목길에서 노을
- 32 좋은 사람 있으면 소개시켜줘 조이 (JOY)
- 33 2002 Anne-Marie

[숙제 - 짭짤한 맛 🍜] - 지니뮤직의 1~50위 곡의 정보를 스크래핑 (처음부터 내 손으로)



순위 / 곡 제목 / 가수를 스크래핑해 출력하면 됩니다.

<https://www.genie.co.kr/chart/top200?ditc=D&rtm=N&ymd=20200713>



힌트:

순위와 곡제목이 깔끔하게 나오지 않을거예요. 옆에 여백이 있다던가, 다른 글씨도 나온다던가.. python 내장 함수인 strip()을 잘 연구해보세요!

▼ 완성 모습

- 1 여름 안에서 (Covered by 싹쓰리) (Feat. 황광희) 싹쓰리 (유두래곤 & 린다G & 비룡)
- 2 아직 너의 시간에 살아 이수현
- 3 How You Like That BLACKPINK
- 4 마리아 (Maria) 화사 (Hwa Sa)
- 5 Summer Hate (Feat. 비) 지코 (ZICO)
- 6 PLAY (Feat. 창모) 청하
- 7 Downtown Baby 블루 (BL00)
- 8 보라빛 밤 (pporappippam) 선미
- 9 Apple 여자친구 (GFRIEND)
- 10 에잇 (Prod. & Feat. SUGA of BTS) 아이유 (IU)
- 11 아로하 조정석
- 12 Monster 레드벨벳-아이린 & 슬기
- 13 Into the I-LAND 아이유 (IU)
- 14 Dolphin 오마이걸 (OH MY GIRL)
- 15 사랑하게 될 줄 알았어 전미도
- 16 살짝 설렜어 (Nonstop) 오마이걸 (OH MY GIRL)
- 17 10억뷰 (Feat. MOON) 세훈 & 찬열
- 18 어떻게 지내 (Prod. by VAN.C) 오반
- 19 MORE & MORE TWICE (트와이스)
- 20 Don't Start Now Dua Lipa
- 21 Dance Monkey Tones And I
- 22 흔들리는 꽃들 속에서 네 삼푸향이 느껴진거야 장범준
- 23 METEOR 창모 (CHANGMO)
- 24 Memories Maroon 5
- 25 Blueming 아이유 (IU)
- 26 아무노래 지코 (ZICO)
- 27 너에게 난, 나에게 넌 미도와 파라솔
- 28 썸 타간 월 타 백아연
- 29 Left & Right 세븐틴 (Seventeen)
- 30 시작 가호 (Gaho)
- 31 늦은 밤 너의 집 앞 골목길에서 노을
- 32 좋은 사람 있으면 소개시켜줘 조이 (JOY)
- 33 2002 Anne-Marie



한 걸음 더!

스크래핑한 결과를 mongoDB에 저장해보세요.

▼ 완성 모습 (Robo 3T에서 봤을 때)

db.getCollection('musicchart').find()		
sparta-local-db localhost:27017 dbsparta		
db.getCollection('musicchart').find()		
musicchart 0.000 sec.		
Key	Value	Type
▼ (1) ObjectId("5f0e96babaed3a987d3acea8")	{ 4 fields }	Object
_id	ObjectId("5f0e96babaed3a987d3acea8")	ObjectId
rank	1	String
title	여름 안에서 (Covered by 썩쓰리) (Feat. 황광희)	String
artist	翕쓰리 (유두래곤 & 린다G & 비룡)	String
▼ (2) ObjectId("5f0e96babaed3a987d3acea9")	{ 4 fields }	Object
_id	ObjectId("5f0e96babaed3a987d3acea9")	ObjectId
rank	2	String
title	아직 너의 시간에 살아	String
artist	이수현	String
► (3) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (4) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (5) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (6) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (7) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (8) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (9) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (10) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (11) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (12) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (13) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (14) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (15) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (16) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (17) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (18) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (19) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (20) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (21) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (22) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (23) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (24) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object
► (25) ObjectId("5f0e96babaed3a987d3acea")	{ 4 fields }	Object

【속제 제출】

- 내 Github 에 올리면(push) 됩니다. 슬랙에 github 봇을 제대로 붙여두었다면 자동으로 알람이 갈 거에요. 만약 제대로 안 뜬다면, Github Repository URL 을 슬랙에 공유해주세요.

복습 도우미

- 이번주 배운 내용을 빠르게 복습할 수 있게 준비했어요! 이번주에도 새로운 재밌는 것들 많이 배웠죠! 😊 다시 한번 복습하면서 빠르게 되짚어보세요.

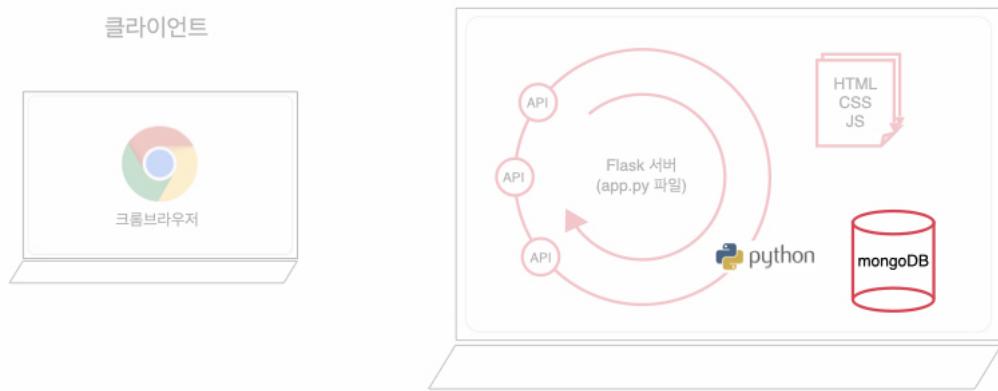
▼ 웹 기초 동작 원리

- 오늘은 우리가 앞으로 만들 API 에서 사용하는 프로그래밍 언어인 파이썬을 배워봤습니다.
- HTTP: 웹은 HTTP라는 규약(규칙)을 따릅니다. url에서 `http://` 가 바로 HTTP 라는 규약을 따른다는 표시예요.
- 클라이언트 : HTTP에서 요청을 하는 쪽
- 서버 : HTTP에서 요청을 받아 응답하는 쪽
- API 를 사용할 땐, 미리 정해둔 **약속** 을 따라야 작동합니다. 약속들은 API 페이지(문서)에 적혀있습니다.

 우리가 앞으로 만들 API 에서 사용하는 프로그래밍 언어인 파이썬(Python)을 배웠습니다. 먼저 문법을 연습하고, 라이브러리를 활용하여 웹 스크래핑을 했죠.

그리고, 우리의 인생 첫 데이터베이스. MongoDB의 데이터를 CRUD 기능을 배웠습니다.
(API를 만드는 건 다음주에!)

서버



▼ 전반부 주요 키워드

- [5분 꿀팁] 질문 잘하기 : 내가 해결하고 싶은 문제를 정확히 알려주기, 지금 내가 무엇을 알고 있는지 알려주기, 어디에 어떤 시간에 질문할지, 질문 해결방법에 대해 피드백하기
- API 를 화면 코드에 사용하기 : '나홀로 메모장 포스팅 가져오기 API' 를 사용해 화면에 포스트를 보여줬습니다. 서버에서 JSON 형식으로 아티클 정보(articles)를 응답(response) 데이터로 보내주죠!
- 파이썬: 파이썬은 매우 직관적인 언어이고, 할 수 있는 것도 많습니다. 필요한 것들은 구글링해서 찾아보면 됩니다.
- 프로그래밍에 도움되는 주요 스킬!

1. 출력하기

값을 확인하거나, 에러를 찾을 때 자주 쓰입니다. 출력해서 버그찾기(Debugging By Printing) 스킬! Python 에서는 출력할 때, `print('출력할 값!')` 사용합니다.

2. 이름 짓기 (naming)

담고 있는 데이터, 기능을 잘 나타낼 수 있는 이름으로 지어주세요. 보통 변수는 명사형으로, 함수는 동사형으로 많이 짓습니다.

파이썬에서는 이름짓기 규칙(naming convention)은 snake style 을 사용합니다. (下划线로 단어 연결하기. 예. `first_name`)
👉 파이썬 공식 스타일 가이드([PEP 8 링크](#) / [파이썬 자습서-코딩 스타일](#))

- 파이썬 패키지(Python Package) : 외부 라이브러리를 사용하기 위해서 패키지를 설치했습니다. requests, BeautifulSoup4, pymongo 모두 외부 라이브러리입니다.
- 파이썬 가상환경(Python Virtual Environment) : 프로젝트별 공구함으로, 같은 시스템에서 실행되는 다른 파이썬 응용 프로그램들의 동작에 영향을 주지 않기 위해, 파이썬 배포 패키지들을 설치하거나 업그레이드하는 것을 가능하게 하는 격리된 실행 환경입니다. (출처 : [파이썬 공식 용어집- 가상환경](#))
- 웹 스크래핑(web scraping): 웹 페이지에서 우리가 원하는 부분의 데이터를 수집해오는 것. 한국에서는 같은 작업을 크롤링(crawling) 이라는 용어로 혼용해서 씁니다.

▼ 후반부 주요 키워드

- CRUD : 기본적인 데이터 처리 기능인 **Create, Read, Update, Delete** 의 두문자를 따서 **CRUD** 라고 합니다.
- MongoDB : No SQL 의 한 종류인 MongoDB를 pymongo 패키지를 이용해 python으로 조작해보았습니다. mongoDB 에 데이터를 CRUD 하는 것을 배웠습니다.
- Git : 작업 기록을 남기고 이력을 추적해서 코드를 손쉽게 관리하도록 도와줍니다.

[1시간] 스스로 소화 타임 시작

- 오늘 배운 것을 복습하고, 본격적으로 숙제를 시작합니다. 모르는 것이 있으면 튜터에게 질문합니다.

✓ 체크아웃

▼ "15초 체크아웃"을 진행합니다.

- 튜터는 타이머를 띄워주세요! ([링크](#)).
- 체크인처럼, 현재 본인의 감정상태와 수업후기에 관해 이야기합니다.
 - 예. 오늘 새로운 걸 배워서 재밌었어요! 중간 중간 놓치는 것들은 집에 가서 복습하면 될 거 같아요!
 - 자유롭게 해주셔도 좋고, **KPT**에 맞춰해주셔도 좋아요.
 - Keep : 오늘 수업을 하면서 좋았던 것, 앞으로도 할 행동 / Problem : 아쉬워서 고쳐보면 좋을 것 / Try : 고치기 위해 내가 할 시도
 - 예) (Keep)오늘 다같이 자기소개를 해서 좋았어요. 친해진 기분! (Problem) 그런데 사전과제 중에 안한 게 있어서 쉬는 시간에 했어요. (Try)수업 전날 지난주 강의자료 확인하고 숙제할게요!
- 튜터님은 체크아웃과 함께 **출석 체크(링크)**를 진행해주세요!
- 출석체크 - (변동이 있다면 다시 제출해주세요!)

스파르타코딩클럽 출석체크
<http://spartacodingclub.shop/attendance>

[설치] - 다음 시간을 위해 미리 설치해와야 할 것들

- 없음!