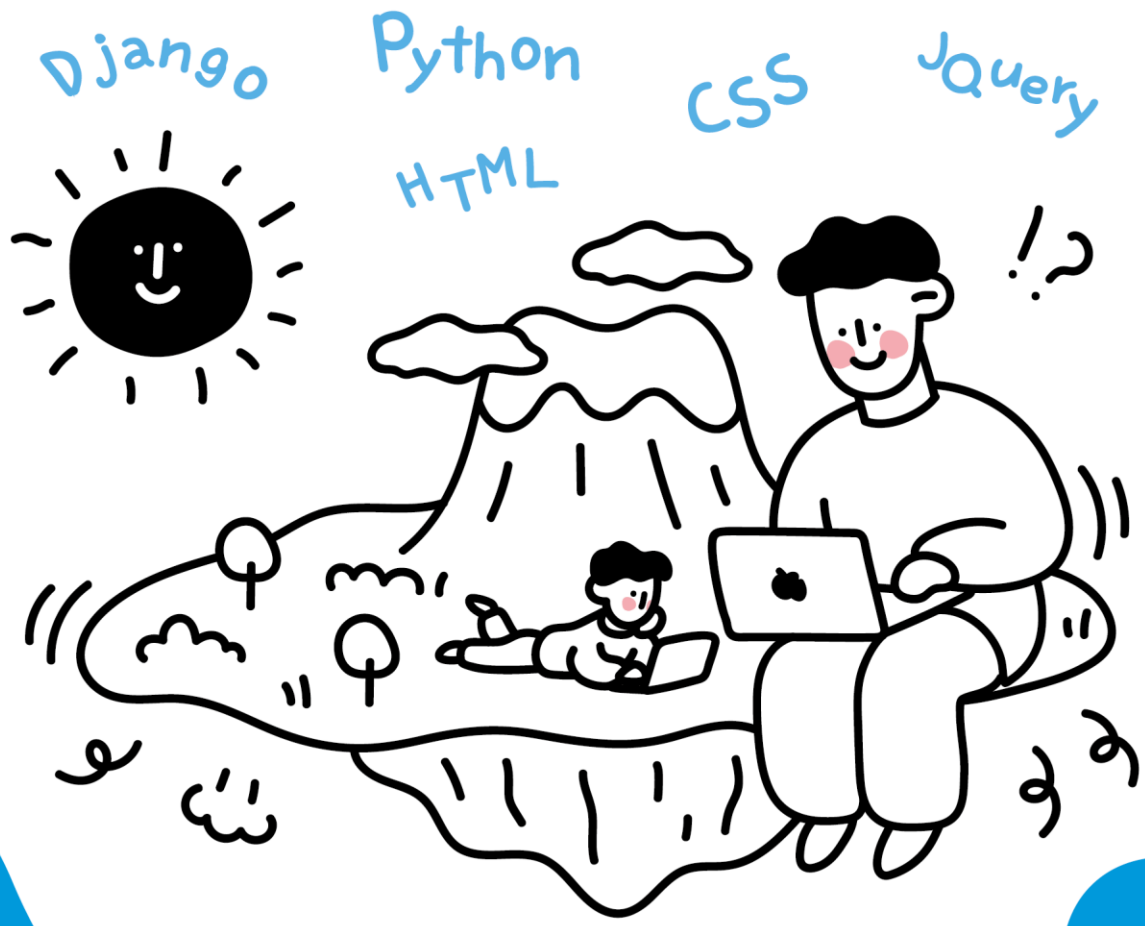
2021

JEJU 코딩 베이스캠프

With. **Django 3.x**

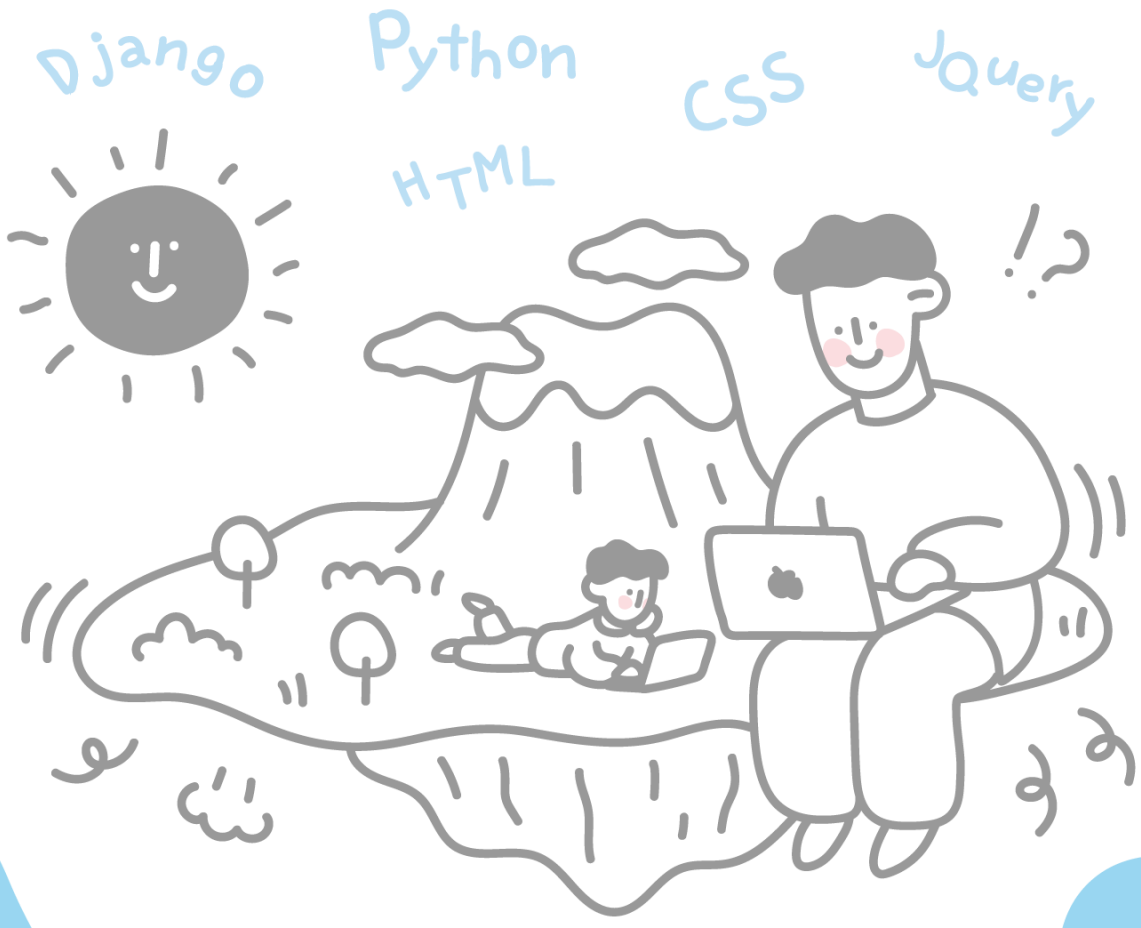


weniv

2021

JEJU 코딩 시 베이스캠프

With. **Django 3.x**



2021 JEJU CODING BASECAMP

개발자는 단기간에 양성될 수 없습니다. 하지만 개발 과정은 단기간에 훑어볼 수 있어요. 서비스 기획부터 서비스 배포까지. 우리가 청년들에게 보여주려는 것은 단순 지식보다는 '당신을 위한 정성'일지도 모르겠습니다.

13기입니다. 사실 앞으로 더 할 수 있을지 모르겠어요. 매번 이번이 마지막일 것 같으니 제대로 해보자는 다짐으로 여기까지 왔습니다. 마음으로, 여러분이 써주신 10장의 지원서 만큼이나 간절한 마음으로 캠프를 준비해 갑니다.

서비스 기획부터 론칭까지 같이 해봅니다. 에러 잡아드려요. 대표님이거나 디자이너이신가요? 개발자와 소통할 수 있습니다. 론칭하고 싶은 서비스가 있으신가요? 완전하지는 않지만 애자일하게 론칭할 수 있어요. 코딩 처음 해보시나요? 괜찮습니다. 여기 기초부터 합니다. 더 낮은 단계도 없습니다.

같이 몰입해봐요. 우리는 어쩌면 너무도 많은 잡음으로 인해 하나의 결과를 못 만드는 것은 아닐까요? 함께 합시다. 또 교육은 교육으로 끝나지 않고, 여러 좋은 기회들도 마주하게 되실 겁니다.

평안하세요. 캠프에서 뵙겠습니다.

운영자 **이호준**

노션으로 바로가기

아래 링크에서 노션(Notion)으로 된 Code Page를 보실 수 있습니다.
실습을 하실 때에는 노션에서 Code를 복사해 사용하세요.

[2021 JEJU CODING BASECAMP] 노션 페이지 링크 : http://bitly.kr/asmXMBfYMcn

이 책에 있는 'JEJU CODING BASECAMP' 강좌는 인프런에서 만나보실 수 있습니다.

저자소개

이호준

주식회사 위니브와 바울랩이라는 회사를 이끌고 있어요.
청소년에게는 ICT 관련 진로와 직업을 찾을 수 있도록, 청년에게는 ICT 업을 찾아주는 일을 하고 있습니다.

유준모

스튜디오밀(Studiomeal.com)을 운영 중이며 유튜브/페이스북에서 '1분코딩'을 운영하고 있습니다.

이범재

스페이스클라우드 개발팀장으로 Ruby on ruby rails에 깊은 애정을 가지고 있습니다.

김혜원, 김유진, 차경림

주식회사 위니브의 COO, CTO, CDO입니다.

김난화, 김진, 김승민, 이진우, 이숭신, 현지연

주식회사 위니브의 연구원입니다.

강민정, 정윤하

바울랩의 연구원으로 많은 행사와 출판을 함께 하였습니다.

정승한

주식회사 위니브의 Back-end 개발자입니다.

제주코딩베이스캠프 로드맵

junior basic	basic	advanced	total advanced	cet
학생을 위한 코딩교육	입문자를 위한 코딩교육	중급자를 위한 코딩교육	입문자~중급자 코딩교육	기타 강의들
프론트엔드				
처음하는 코딩 - 플레이봇(Javascript)	30분 요약강좌 : HTML,CSS,JS,Python	Python 100제	코알못에서 웹서비스 런칭까지 : 제주코딩베이스캠프	직접 만드는 노션 템플릿 BEST 7선
백엔드				
처음하는 코딩 - 러플(Python)	30분 요약강좌 시즌2 : 파이썬 활용편	JS 100제	Python all-in-one	깔끔한 협업, 슬기로운 슬랙생활 시작하기
디지털 리터러시 With 제주코딩베이스캠프	웹 개발의 시작 : HTML,CSS,JS,Bootstrap	Django Mini project BEST 3	인스타그램 클론 강의	프로그래머의 콘텐츠 생산 가이드
파이콘 2018 영코더 with 바울랩	웹 개발의 시작 : JS,Jquery,ajax	Node,JS,Express	페이스북 클론 강의	나도 할 수 있다! SW강사
	Python 부트캠프	웹 해킹의 시작 : SQL injection	제주 하긴디 이신 데이터들 Python으로 몬딱 분석해불게	온라인 개학에 대한 모든 것
	생애 첫 SQL With 제코베	Python 엑셀 프로그래밍 - with xlsxwriter	캐글 데이터로 살펴보는 데이터분석 (With Python and SAS)	Microcosm SW Fair -경쟁없는, 나만의 시빅해킹
	나의 개발 유형을 테스트 해보자! <M.B.I.T>	실전 예제로 살펴보는 PyQt Tutorial	정보 보안 패스파인더	
		논떠보니 코딩 테스트 전날	메모혁신 노션(Notion) 가이드	
		PyGame 어드벤쳐 : 위니브 월드를 수복하라!	스타트업 A to Z	
		테스트 페이지 만들기		

jeju coding basecamp

수료증 제도

제코베 수료증 제도란?

제주코딩베이스캠프 강좌 수강(90% 이상) 및 아래 안내된 IT 관련 활동을
30학점 이상 수료 시 **수료증 및 각종 혜택을 제공해 드립니다.**

제주코딩베이스캠프 온라인 강의

1학점	2학점	3학점	5학점	1학점

기타 강의 및 행사

제코베 강좌를 제외한 IT강의(플랫폼 제한 없음)	1학점·최대 5학점
제코베 오프라인 강의 및 행사 참여	5학점

기타 IT 관련 활동

개인 유튜브 강의 및 유료 강의 콘텐츠 제작	2학점·최대 2학점
IT 관련 서적 집필	3학점·최대 3학점
IT 관련 서적 서평 작성	2학점·최대 4학점

혜택 1

인재풀 등록 및 취업 지원

혜택 2

수료증 및 제코베 굿즈 키트

혜택 3

강의 공동 제작 책 공동 집필 기회

참여 방법

1. 제코베 온라인 강좌, 기타강의 및 행사를 20학점 이상, 90% 이상 수강/참여한다.
2. 증빙 자료(캡쳐)를 구글 설문지를 통해 제출한다.
3. 취업 지원 또는 포트폴리오 인증서 발급을 원할 경우 포트폴리오/Github/블로그 URL 등을 구들 설문지를 통해 제출한다.

공지 사항

- 제코베 기본 강좌인 '코알못에서 웹서비스 런칭까지 : 제주코딩베이스캠프'는 필수로 수강해야 합니다.
- 포트폴리오 인증서는 수료증을 발급 받게 되는 분들을 대상으로 하며, 원할 경우에 포트폴리오 검수 후 발급해 드립니다.

수료증 신청 구글 설문지 〉〉〉

Contents

Day1

소개 및 학습 내용 개요

1. 소통과 목표치 설정
2. 자문자답!
3. 인터넷은 어떻게 구성되어 있을까요?
4. 무엇을 어떻게 이용하여 개발할 것인가요?
5. 프로젝트는 어떻게 설계할 것인가요?
6. 왜 이러한 교육과정으로 배워야 하죠?
7. 왜 Django인가요?

환경세팅

1. 왜 환경을 세팅해야 할까요?
2. TextEditer와 IDE의 차이점
3. 로컬 환경설정
  3.1 Atom 설치 및 사용방법
  3.2 테마 변경
  3.3 유용한 기능
  3.4 Emmet 패키지 사용법
4. 서버 환경설정
5. Ngnix 설치 및 구동(선택사항)

HTML & CSS

1. HTML의 기초
2. HTML5
3. CSS
4. Code
5. CheatSheet

Day2

JavaScript

1. Javascript의 기초
2. 변수
3. Javascript의 연산
4. 조건문
5. 함수
6. DOM(Document Object Model)
7. HTML FORM
8. 이벤트
9. 정규표현식
10 Javascript CheatSheet

jQuery

1. 라이브러리
2. jQuery의 필터와 실행방법
3. 이벤트
4. jQuery CheatSheet

NotionDB

Day3

Bootstrap

1. 부트스트랩
2. 부트스트랩 적용
3. 그리드
4. 이미지
5. 테이블
6. 버튼
7. CARD
8. 캐러셀
9. 마무리하며

Contents

웹서비스 개발 기획

1. 왜 홈페이지 기획을 하는가?
2. 무엇을 이용해서 기획을 하는가?
3. 마인드맵(FREEMIND)
4. 화면설계(KAKAO OVEN)
   카카오 오븐(https://ovenapp.io)

Git & Github

1. 소스코드 버전관리를 사용하지 않았을때 발생했던 문제점
2. 소스코드 버전관리를 통해 문제가 해결되는 사례
3. 소스코드 버전관리란?
4. Git이란?
5. Git 설치하기
6. 저장소 만들기
7. 수정하고 저장소에 저장하기
8. Branch
   8.1 브랜치의 사용예
9. Github와 Bitbucket
10. Atom을 이용한 Git 사용하기
11. Git에 관한 잡다한 이야기들

Day4 Python

1. 얕은물 강좌 스토리 미리보기
2. 인쇄용 PDF 파일 다운로드 링크
3. 환경설정
   (1) 라이캣의 탄생!
   (2) 생선을 잡아라
   (연습문제) 캣의 해골섬 출항!
   (3) 효율적으로 생선 잡기
   (4) 생선 회사를 운영하라
   (연습문제) 캣의 고객 관리
   (5) 라이캣의 EXIT
   (연습문제) 사자탈을 쓴 캣
   (스토리) 쏘아진 화살!
   (6) 파이와 썬의 알고리즘 7원석을 찾아서
   (연습문제) 캣의 모험 자금을 모아라!
   (연습문제) 파이와 썬이 남긴 단 하나의 단서
   (7) CheatSheet

Day5 Django 3.1

1. 장고 소개
2. 구름 IDE 환경설정
3. 장고 프로젝트 생성
4. 프로젝트 개요
5. 메인 페이지
6. cafelist 페이지
7. 모델 작성
8. cafedetails 페이지
9. 이미지 넣기
10. 댓글기능 추가하기
11. 게시글 작성일/수정일 추가
12. 카페 위치 추가
13. 모델에 정보 추가하기
14. 템플릿 적용 및 검색 기능 구현
15. 카페 등록 페이지
16. 서비스 배포
17. 구글 맵 API
18. 번회 튜토리얼
19. 요약정리

제주코딩베이스캠프 일정

시간	Front-end		Back-end		
	8/10(월)	8/11(화)	8/12(수)	8/13(목)	8/14(금)
10:00 ~ 13:00	소개 및 학습내용 개요 설명	Javascript & JQuery	Bootstrap	Python 문법	Django 활용
점심시간					
14:00 ~ 17:00	HTML, CSS	Javascript & JQuery	Python 문법	홈페이지 기획, 개발자 도구	Django 활용, 팀빌딩

시간	해커톤	
	8/17(월)	8/18(화)
10:00 ~ 13:00	해커톤 진행	해커톤 진행
점심시간		
14:00 ~ 17:00	해커톤 진행	결과 발표 및 심사

※ 제주코딩베이스캠프는 제주에 다양한 코딩 프로그램, 캠프, 아카데미 등이 절대적으로 부족하다고 느낀 IT 분야의 전문가들이 모여 만든 교육 과정으로, 코딩을 배우고자 하는 사람들을 중심으로 진행되는 5~7일간의 오프라인 교육과정 + 해커톤을 의미합니다. 위 일정표는 제주코딩베이스캠프 오프라인 과정의 일정표입니다.

Day 1

소개 및 학습 내용 개요

1. 소통과 목표치 설정
2. 자문자답!
3. 인터넷은 어떻게 구성되어 있을까요?
4. 무엇을 어떻게 이용하여 개발할 것인가요?
5. 프로젝트는 어떻게 설계할 것인가요?
6. 왜 이러한 교육과정으로 배워야 하죠?
7. 왜 Django인가요?

1. 소통과 목표치 설정

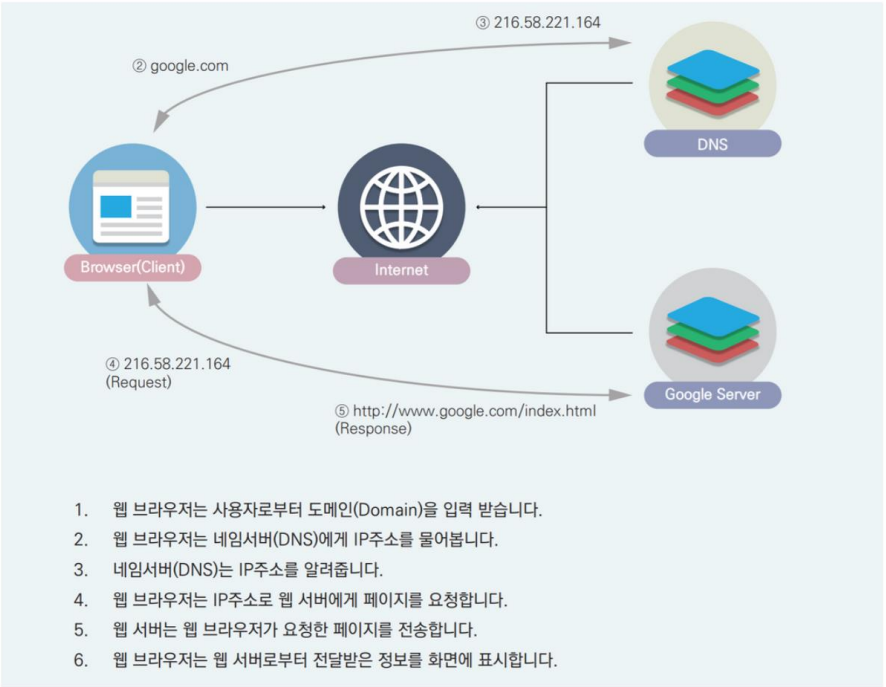
- 어디서 오셨나요?
- 기대하는 것은 무엇인가요?
- 어떤 서비스를 만들고 싶으신가요?
- 이런 것이 가능해요.
- 이런 공모전에도 출품할 수 있어요.

2. 자문자답!

코딩이 뭔가요? 노트북은 뭘 사야 하죠? 제주 코딩 베이스캠프는 무엇인가요? 등에 여러 질문들이 있으셔서, 동영상으로 만들어 놓았습니다. 아래 영상을 참고해주세요.

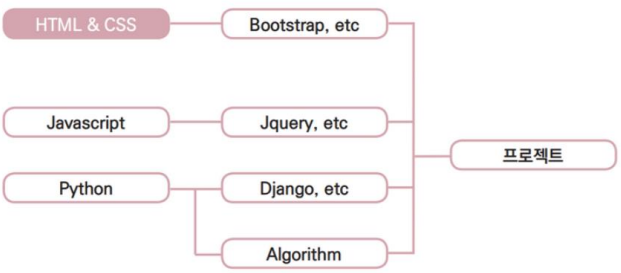
혹시 PDF로 이 강좌를 보고 있으시다면, **youtube에서 '제주 코딩 베이스캠프'**를 검색해주세요.

3. 인터넷은 어떻게 구성되어 있을까요?



1. 웹 브라우저는 사용자로부터 도메인(Domain)을 입력 받습니다.
2. 웹 브라우저는 네임서버(DNS)에게 IP주소를 물어봅니다.
3. 네임서버(DNS)는 IP주소를 알려줍니다.
4. 웹 브라우저는 IP주소로 웹 서버에게 페이지를 요청합니다.
5. 웹 서버는 웹 브라우저가 요청한 페이지를 전송합니다.
6. 웹 브라우저는 웹 서버로부터 전달받은 정보를 화면에 표시합니다.

4. 무엇을 어떻게 이용하여 개발할 것인가요?



Front-end로 HTML, CSS, Javascript, jQeury를 배워 Bootstrap으로 빠르게 UI를 만들고, Back-end로 Python를 배워 Django로 웹 서비스를 구축하는 법을 배웁니다.

그런데, 이것들이 다 뭔가요? 아래 영상을 참고해주세요.


HTML, CSS, JS와 Front-end! 무엇을 할 수 있나요?
나중에 시...
공유
HTML
CSS
JS
Front-end


Python과 Back-end 개발자! 무엇을 할 수 있나요?
나중에 시...
공유
Python
Back-end

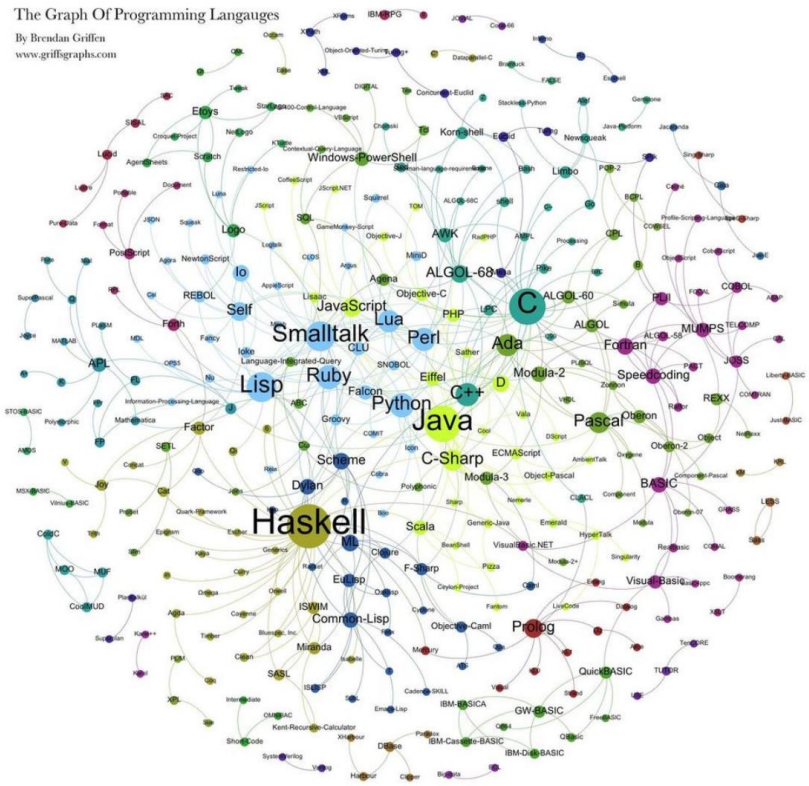
5. 프로젝트는 어떻게 설계할 것인가요?

아이디어를 가장 빠르게 구축해볼 수 있기 때문이죠! 한 Circle! 우리가 추구하는 방향입니다. 앞으로 여러분들은 다른 스택을 공부하게 되실 수도 있어요. 예를 들어 우리는 Python을 배웠지만, 취업을 위해, 또는 직장에서 Java - Spring으로 이뤄지는 패턴을 많이 만나게 되실 텐데요. 그렇다 하더라도, 지금 배우셨던 것이 많은 도움이 되실겁니다!

이렇게 빠르게 아이디어를 구축할 수 있다는 것은 큰 장점인데요. 어떤 서비스를 만들어왔고, 어떤 것이 필요한지 이해해봅시다. (오프라인 강좌에서는 지금까지 해커톤에서 나왔던 아이디어, 문서들을 보여드립니다.)

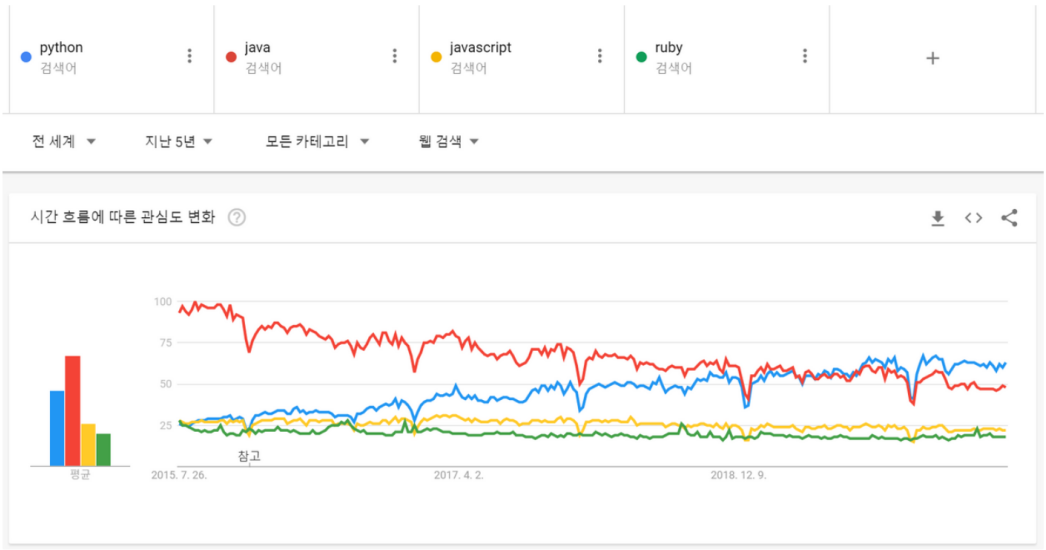
1. 프로젝트 요구사항 정의
    - 서비스 기획
    - usecase
    - 프로젝트 문서
2. 인프라 구축(서버 세팅) URL 구매 후 연결!
3. 애자일한 개발
4. 사용자 피드백 후 다시 1번으로

6. 왜 이러한 교육과정으로 배워야 하죠?



출처 : https://namu.wiki/w/프로그래밍 언어/종류

수 많은 언어 중 빠르게 서비스를 구축할 수 있으면서도, 커뮤니티가 활성화 되어있고, 직관적이고 배우기 쉬운 언어이기 때문입니다! 여러분의 약간의 노력만 더해지면 홈페이지를 쉽고 빠르게 구축할 수 있어요! 물론, 실 서비스를 할 정도에 퀄리티를 지니려면 좀 더 노력해야 한답니다!



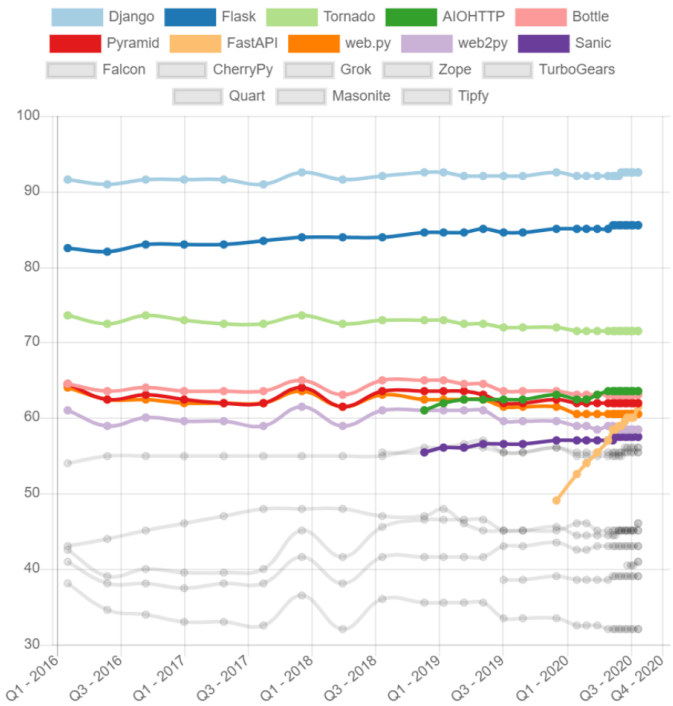
출처 : google trend 전 세계 - C의 점유율이 워낙 높아 제거

- 지속적으로 공부를 하실 것이라면 Java나 C의 점유율을 무시해서는 안됩니다.

7. 왜 Django인가요?

Python

Framework	Score
Django	92
Flask	85
Tornado	71
AIOHTTP	63
Bottle	63
Pyramid	62
FastAPI	61
web.py	60
web2py	58
Sanic	57
Falcon	56
CherryPy	55
Grok	46
Zope	45
TurboGears	43
Quart	41
Masonite	39
Tipfy	32



출처 : https://hotframeworks.com/languages/python

Full-Stack Framework로 여러분이 원하는 모든 기능이 갖춰져 있습니다. 심지어 DB까지요! 다른 Framework는 그렇지 않은 경우도 있어요. 그리고 점유율이 높기 때문에 커뮤니티가 활성화 되어있고 공식문서도 아주 잘 되어 있답니다. 아래 사이트에서 공식문서를 확인해주세요.

Day1 - 소개 및 학습 내용 개요

PDF파일은 무려 1998페이지입니다. 여러분이 원하는 내용이 무엇이든, 여기에 거의 다 담겨있어요!

공식문서 :

Django documentation | Django documentation | Django

Everything you need to know about Django. Are you new to Django or to programming? This is the place to start! Having trouble? We'd like to help! Django has a lot of documentation. A high-level overview of how it's organized will help you know

https://docs.djangoproject.com/en/3.0/

PDF 파일 :

buildmedia.readthedocs.org

https://buildmedia.readthedocs.org/media/pdf/django/3.0.x/django.pdf

Day 1

환경세팅

1. 왜 환경을 세팅해야 할까요?
2. TextEditer와 IDE의 차이점
3. 로컬 환경설정
   - 3.1 Atom 설치 및 사용방법
   - 3.2 테마 변경
   - 3.3 유용한 기능
   - 3.4 Emmet 패키지 사용법
4. 서버 환경설정
5. Ngnix 설치 및 구동(선택사항)

1. 왜 환경을 세팅해야 할까요?

메모장에서도 충분히 코딩할 수 있어요. 그런데 왜 사용하기 어려운 여러 소프트웨어를 사용하여 편집해야 할까요? 메모장은 요리에 비유하자면 업무용 칼입니다. 업무용 칼로 물론 요리를 할 수 있지만, 여러가지 재료가 잘 썰리지 않겠죠. 생선회를 뜰 때 얼마나 다양한 칼을 사용하는지 아시나요? 이처럼 여러 기능을 제공하는 Coding용 TextEditer나 IDE를 사용하는 것은 프로그래머에게 필수 입니다!

2. TextEditer와 IDE의 차이점

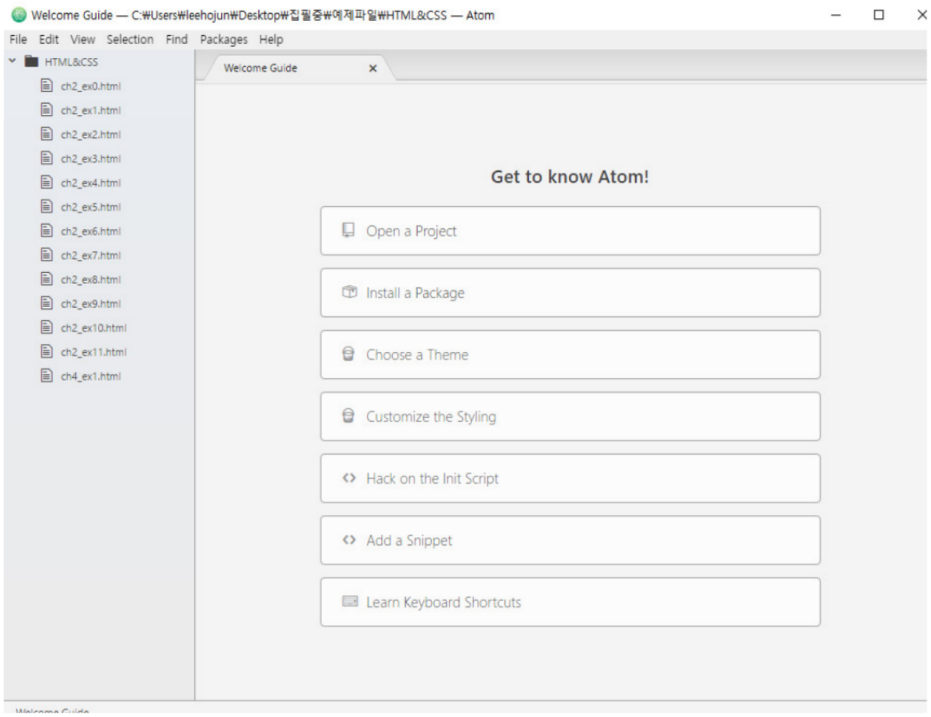
자, 그럼 구체적으로 TextEditer와 IDE 차이점에 대해 알아봅시다. 그런데, 이 챕터는 넘어가셔도 좋습니다. 지금은 구분을 하지 않으시고, 추천하는 에디터를 사용하세요! 3가지를 추천합니다.

1. Visual Studio Code(VSC)
2. Sublime Text나 Atom

여기서는 Atom을 사용하는데요. 이유는 더 빠르다거나, 기능적으로 좋아서는 아닙니다. '**초보자가 사용하기 쉬운 에디터**'라서 선택을 하였습니다.

따라서 VSC를 사용하실 분들은 아래 영상을 참고해주세요. 총 3편까지 있지만, 1편만 첨부합니다.

3. 로컬 환경설정

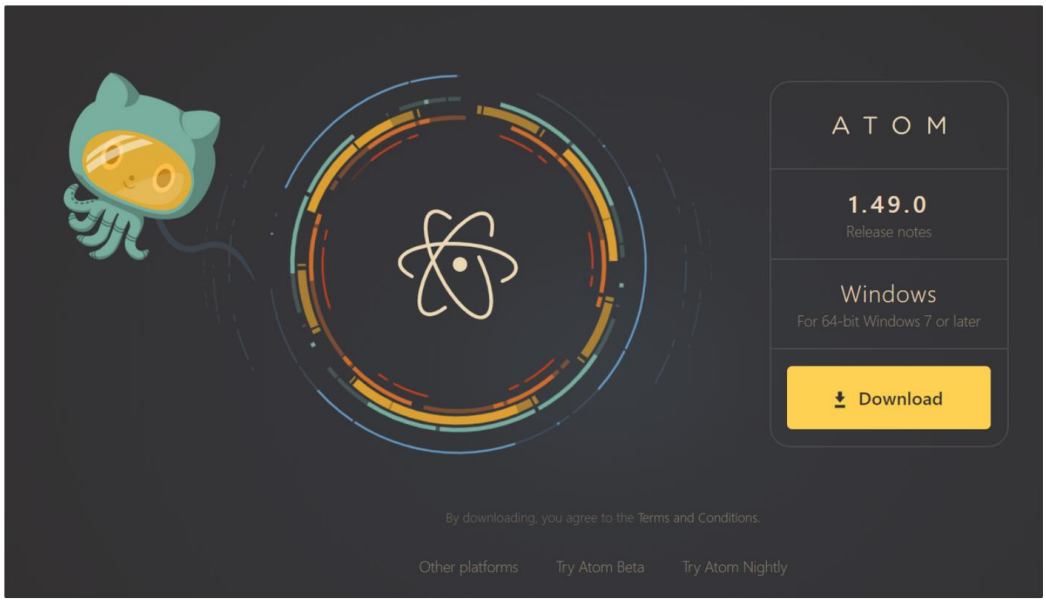


Atom은 github에서 제작한 에디터 입니다. git을 쓰시는 분들에게는 이보다 좋은 에디터는 없을 것으로 보입니다. 아직 git을 사용해보시지 않으셨다면 한 번 사용해 보시길 권해드립니다. 추가 기능이 필요하시다면 패키지를 인스톨하여 원하는 에디터를 만들어갈 수 있습니다.

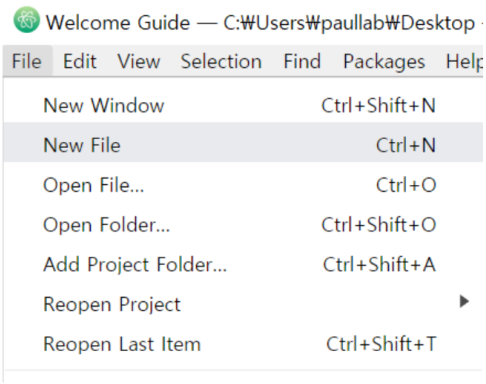
이 역시 영상으로 찍어두었는데요. 아래 텍스트로도 준비되어 있으니 편하신 콘텐츠로 보시길 권해드립니다.

3.1 Atom 설치 및 사용 방법

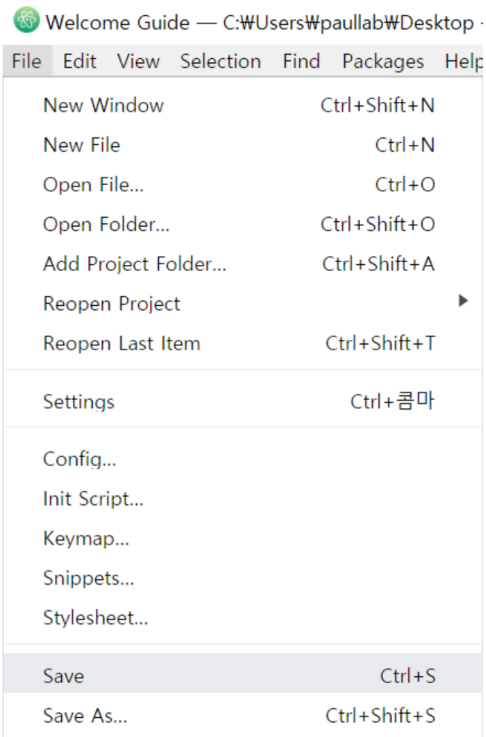
1. https://atom.io 홈페이지에 접속합니다. 구글에서 Atom을 검색하면 최상단에 있습니다. 설치 파일을 다운로드 받고 실행하셔서 설치를 진행하세요.



2. Atom을 실행하고 File > New File을 선택합니다. 단축키는 `Ctrl + N` 입니다.

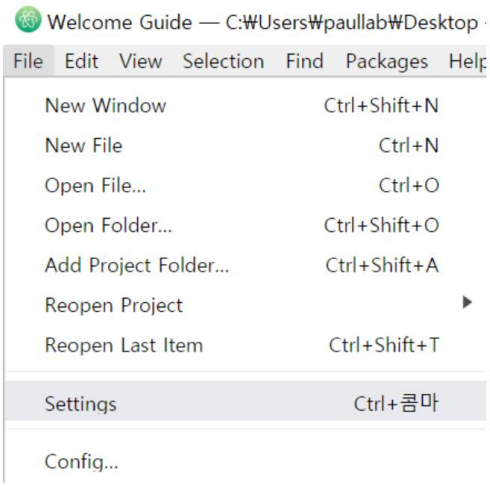


3. untitled 창에서 File > Save 를 클릭해 저장합니다. 저장할 때는 파일명 `.html`로 저장해야 합니다. `.html`로 저장한 파일에서는 코드하이라이팅이 적용됩니다.

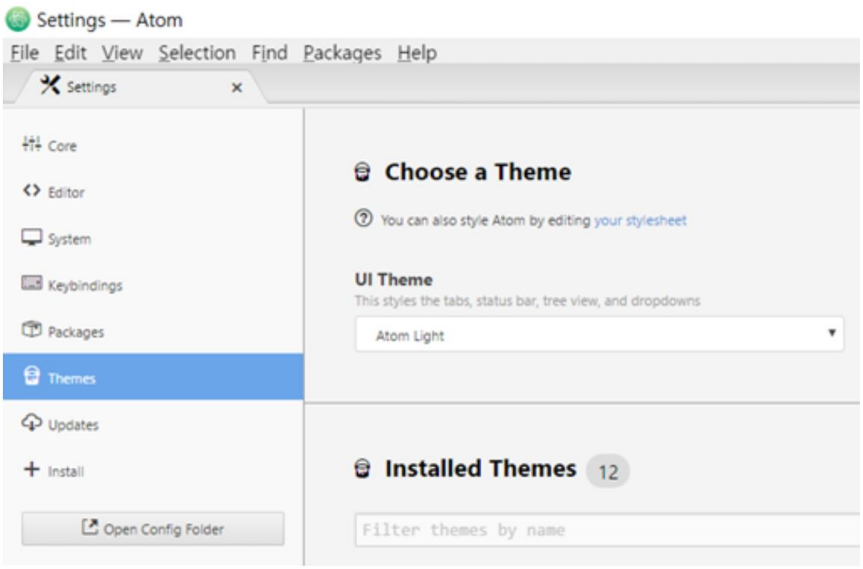


3.2 테마 변경

1. File을 누르고 Settings를 누릅니다. 단축키는 `Ctrl + 콤마` 입니다.

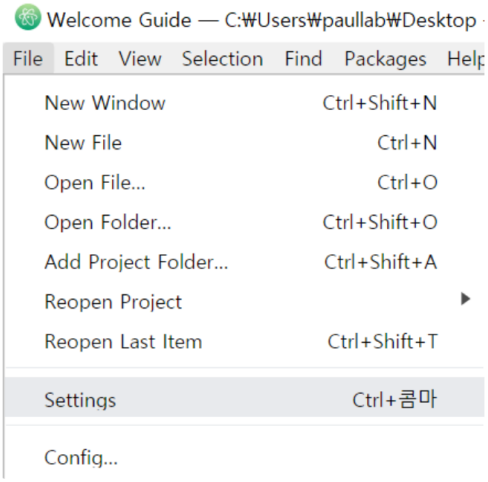


2. UI Theme에서 테마에 대한 설정을 변경할 수 있습니다. 이 책에서는 Atom Light 테마를 사용하도록 하겠습니다.

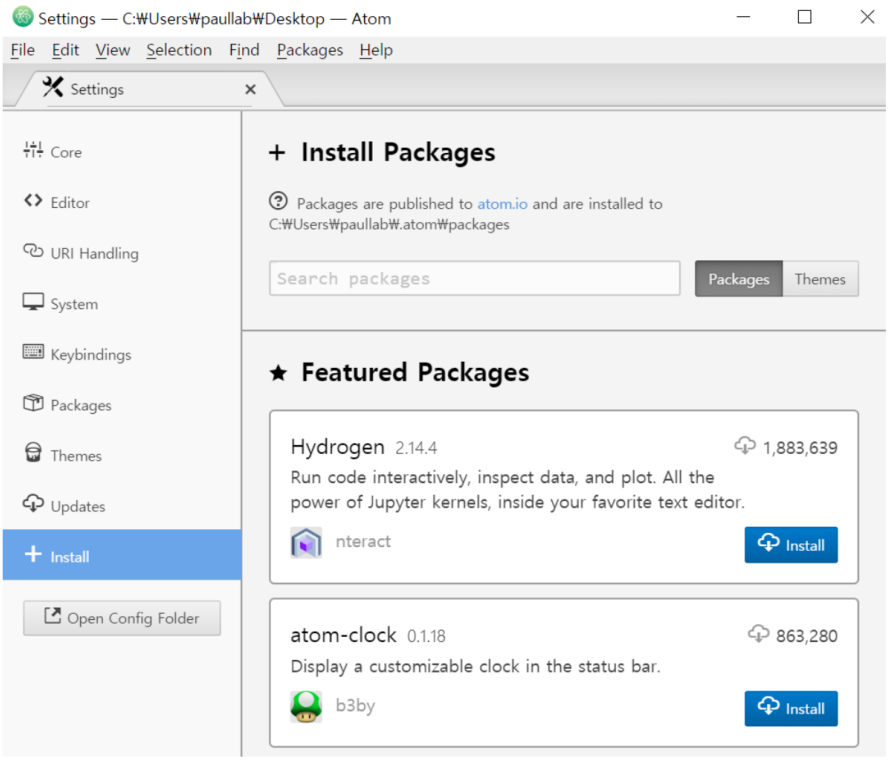


3.3 유용한 기능

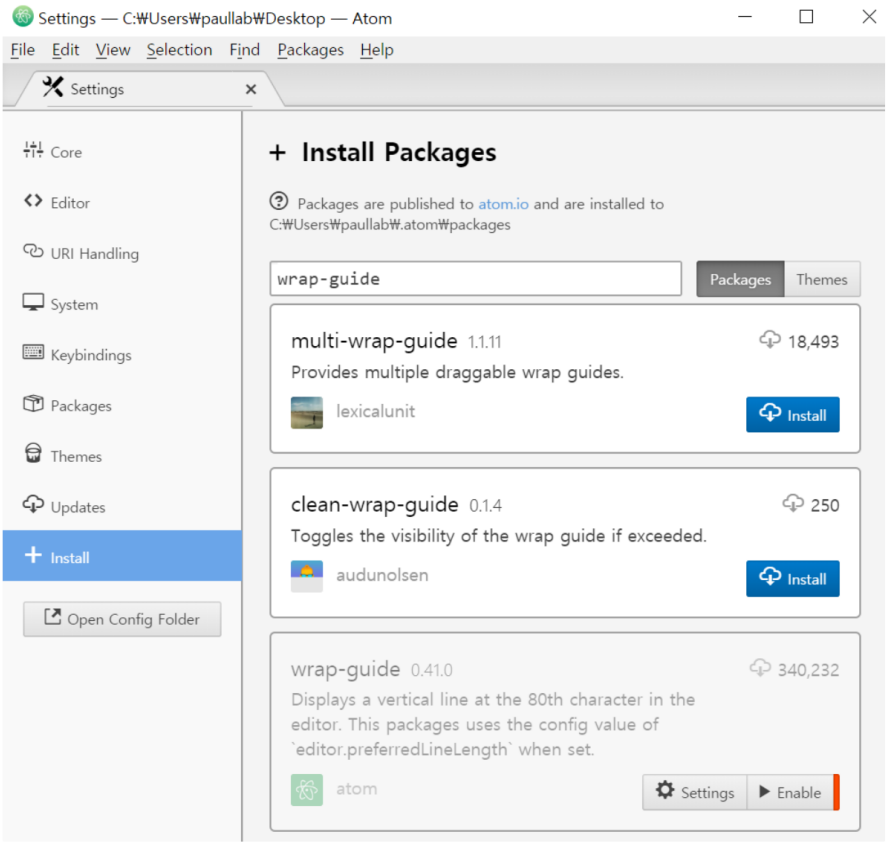
1. File을 누르고 Settings를 누릅니다.



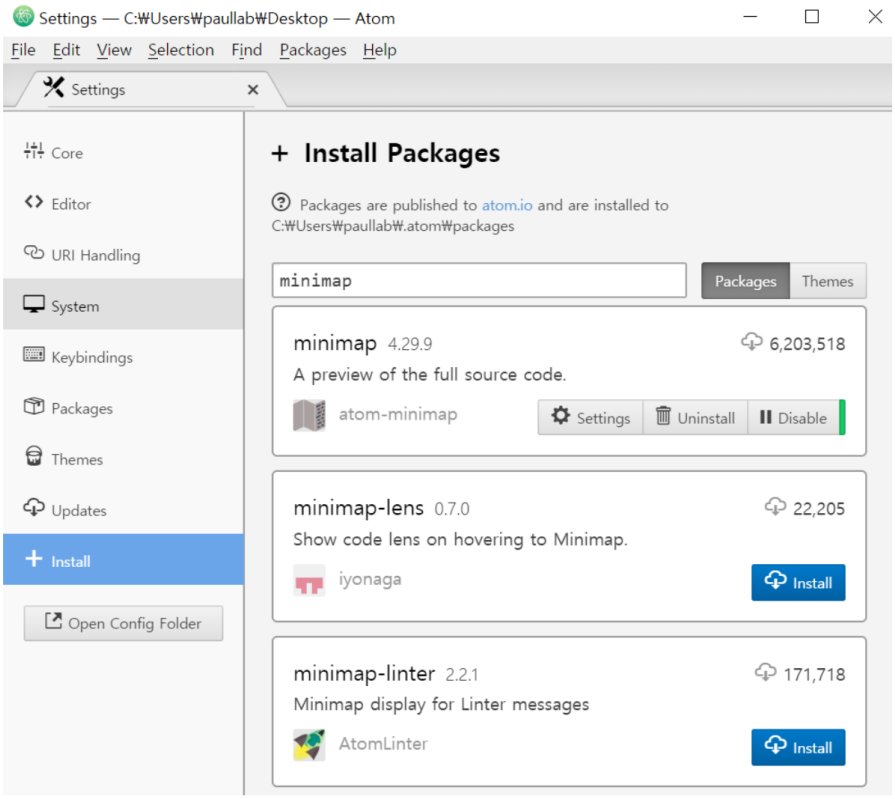
2. Install에서 Editor를 Customizing할 수 있는 몇 가지 Packages를 설치하도록 하겠습니다.



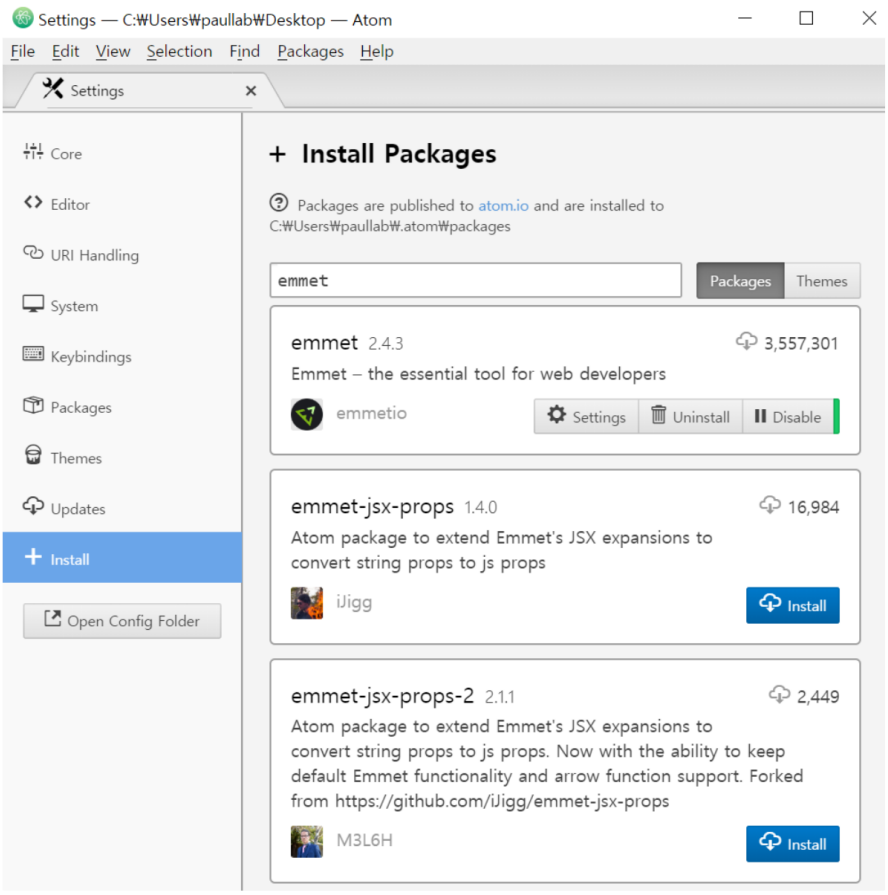
3. `wrap-guide`로 검색하면 wrap-guide package가 나올 것입니다. 에디터 안에서 오른쪽에 라인을 그려주는 Package인데 지워 사용하는 편이 더 깔끔하기 때문에 Disable 시키도록 하겠습니다. `Disable` 버튼을 눌러주시면 아래와 같이 화면이 바뀝니다.



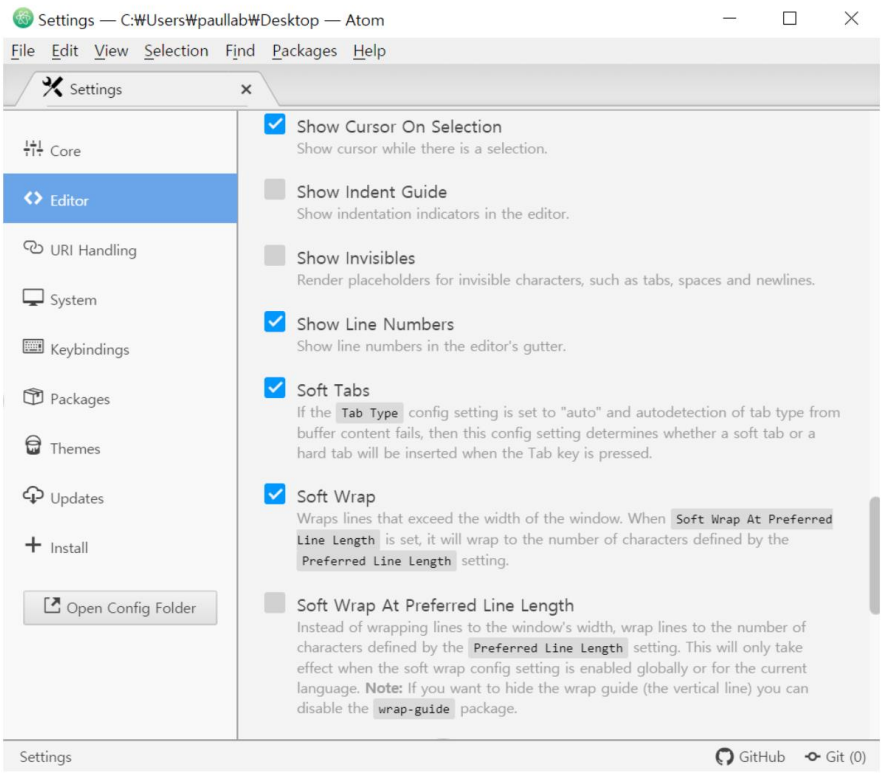
4. 이번에는 `minimap` 이라는 패키지를 설치하도록 하겠습니다. minimap이라고 검색하고 minimap에 install 버튼을 눌러주세요! Code의 긴 줄을 한눈에 보기 편하도록 오른쪽에 작은 Map을 그려주는 유용한 Package입니다.



5. emmet 패키지는 웹 관련 코드를 자동완성 시켜주는 패키지 입니다. 상당히 유용하지만 사용법을 어느정도 숙지해야 하기 때문에 뒷장부터 사용법을 언급하도록 하겠습니다. 가능하다면 YouTube 영상을 시청해주세요!

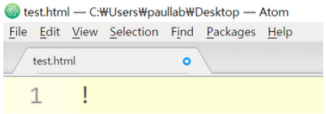


6. 이번에는 Editor로 들어와서 Soft Wrap에 체킹을 해줍니다. 라인이 허용치를 넘어가면 옆으로 스크롤이 생기는데 스크롤이 생기지 않고 아래로 넘어가도록 하는 옵션입니다.



3.4 Emmet 패키지 사용법

1. emmet 패키지 사용법을 어느정도 숙지해 두는 것은 생산성을 높여줍니다. 가장 먼저 파일을 `.html`로 저장해주세요. 저장하지 않으면 emmet문법이 작동하지 않습니다! 자, 이제 `!`를 입력해주시고 `Tab`을 눌러보세요. 아래와 같이 HTML의 기본 골격이 한번에 만들어지는 것을 볼 수 있습니다.



2. 이번에는 위와 같이 입력하고 하나의 라인 끝에서 Tap을 눌러보세요. 예를 들어 아래와 같이 h1 뒤에서 Tab을 눌러보십시오. 그러면 그 아래 화면과 같이 변하는 것을 볼 수 있습니다. 위의 예제를 모두 실습하면 그 원리에 대해 이해하실 수 있으실 겁니다.

```
1   h1
```

```
1   <h1></h1>
```

```
<body>
  h1
  h1+h1
  h1*3
  h1{hello world}*3
  table>(tr>th*3)+(tr>td*3)
  h1#one+div.two+div.three.four.five
  h1.test$$*3
</body>
```

HTML

4. 서버 환경설정

서비스를 운영하기 위해서는 서버가 필요합니다. 그러나 서버를 직접 사기에는 너무 고가이고, 또 배송이 온다고 해서 끝이 아니라 설치를 하는 것도 매우 큰 리소스가 드는 작업입니다. 우리는 그래서 이 서버를 빌려서 사용합니다. 빌리는 것을 클라우드 서비스라 하고, 이 서비스는 크게 3가지로 나눌 수 있습니다.

1. **SaaS**(Software as a Serviece)

   SaaS는 가장 사용자 단에 친밀한 서비스이며 네트워크를 통해 애플리케이션 기능을 이용할 수 있는 서비스입니다.

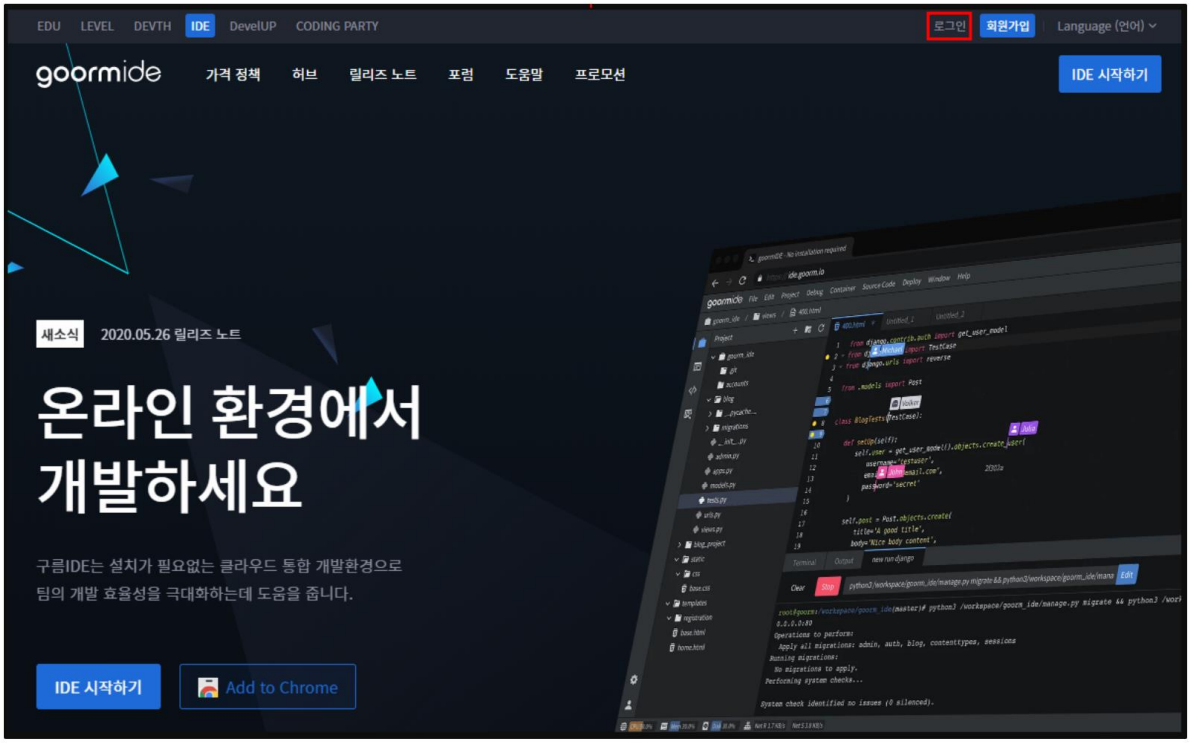
2. **PaaS**(Platform as a Service)

   PaaS는 윈도우와 리눅스 같은 운영체제를 제공하고 개발 가능한 플랫폼도 함께 제공되는 클라우드 서비스입니다.

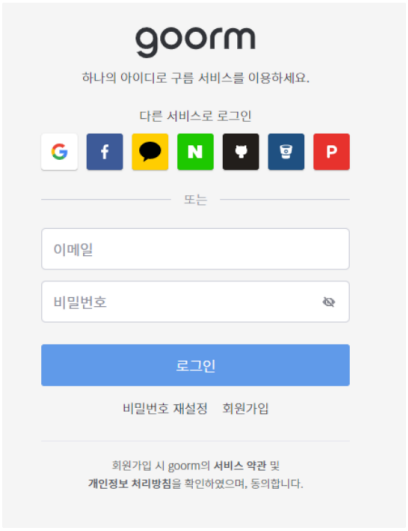
3. **IaaS**(infrastructure as a Service)

   IaaS는 인프라를 제공하는 클라우드 서비스입니다. 기업에서 특히 많이 쓰입니다.

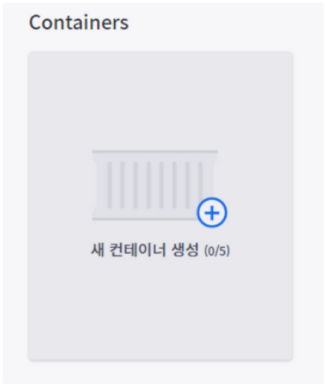
구름IDE 환경설정을 해보도록 하겠습니다. 아래 화면을 보시면서 빨간색 네모를 따라 그대로 따라하시면 됩니다.

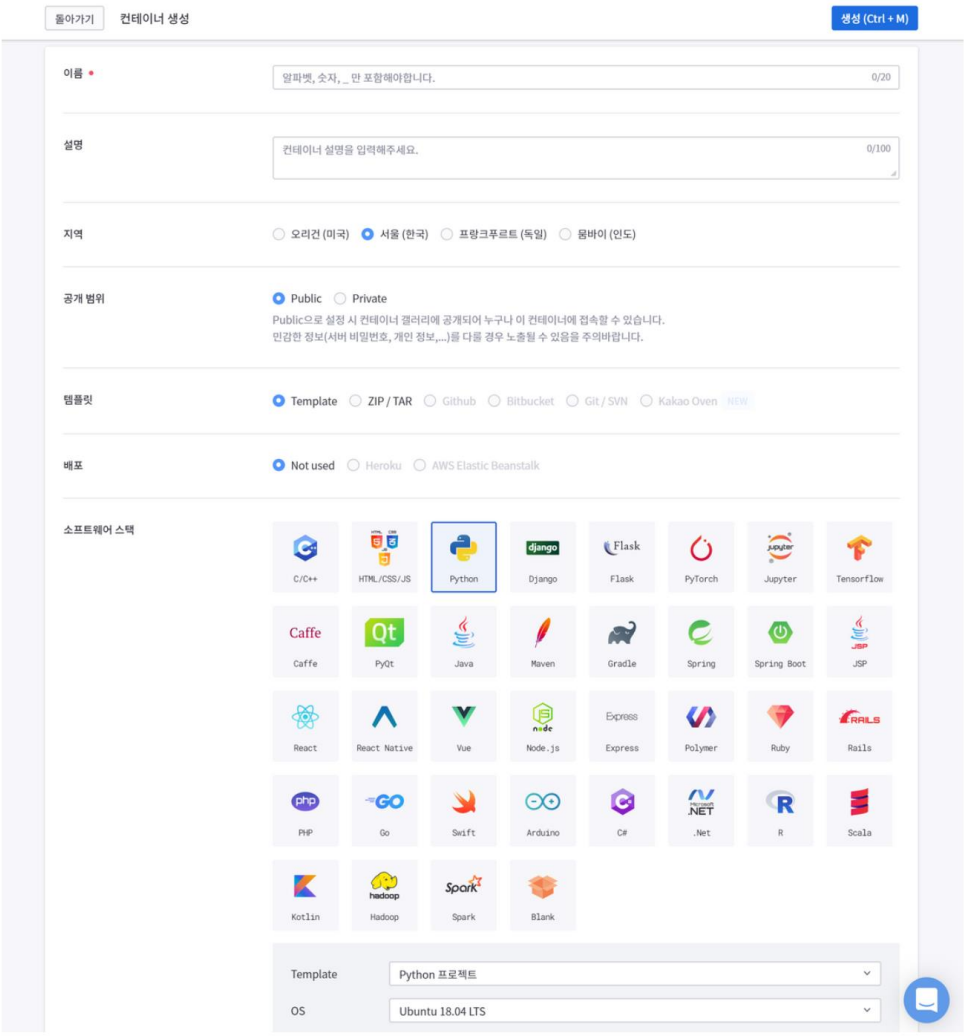


회원가입을 해야 합니다. 무료계정도 컨테이너 5개를 사용할 수 있는 강력한 서비스입니다. 회원가입을 하면 자동적으로 대시보드에 들어갑니다. 만약 메인화면으로 나오셨다면 대시보드를 눌러주세요.

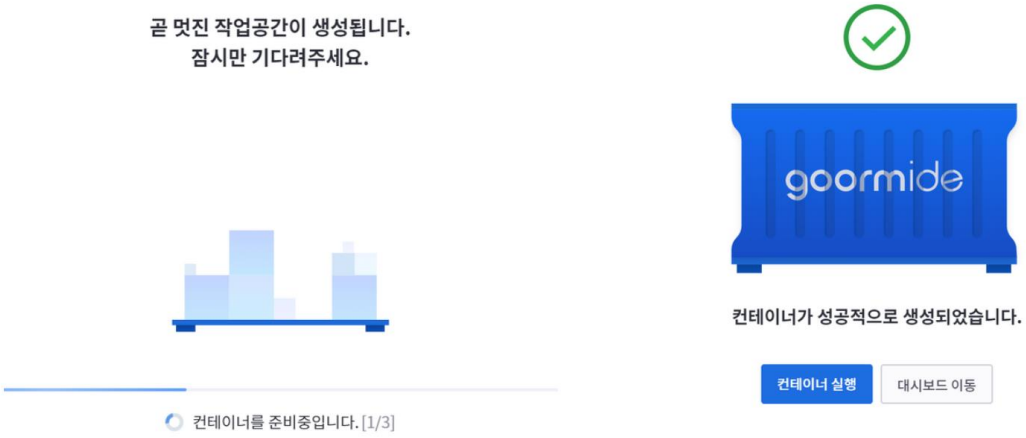


다음은 대시보드 화면입니다. 새 컨테이너 생성을 누르셔서 컨테이너를 생성해 주세요. 하나의 컨테이너는 하나의 컴퓨터를 세팅하는 것과 같습니다. Python을 선택하신 다음 생성하기를 눌러주세요.

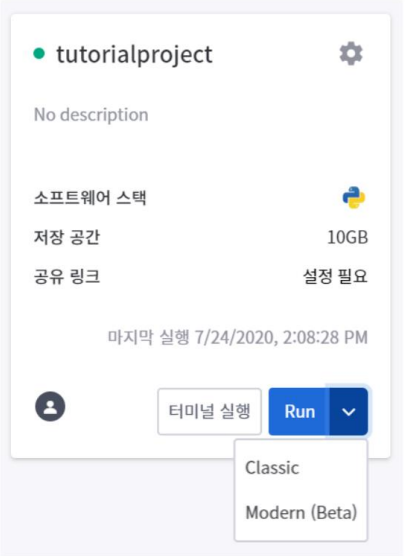




컨테이너가 만들어지고 있다는 화면이 나오고 곧 컨테이너 생성이 완료되었다고 뜹니다. 컨테이너 실행을 누르지 마시고 대시보드로 이동해주세요.



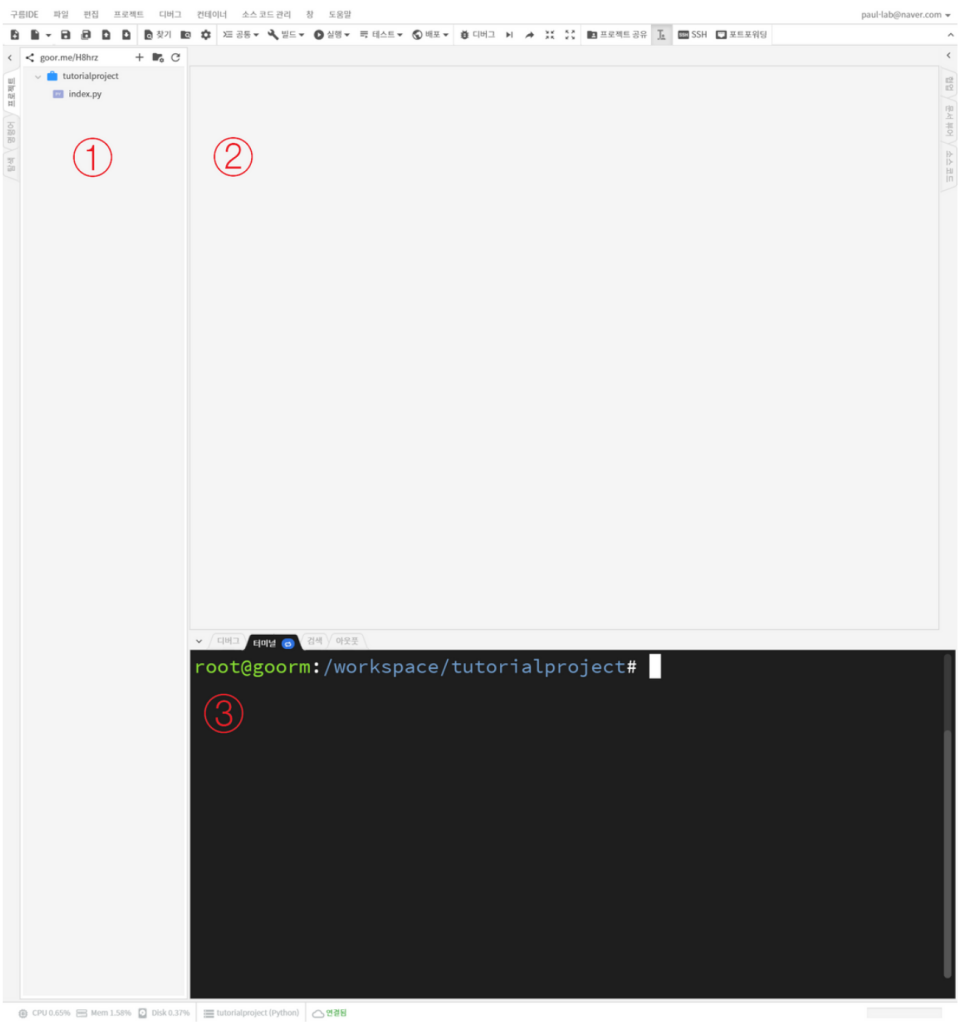
대시보드로 이동한 다음에 `Run 옆에 있는 버튼`을 눌러주세요. 우리는 `Classic`으로 실행합니다. Modern 버전이 가끔 애러가 나기 때문입니다. 컨테이너 실행을 누르셔서 컨테이너 안으로 들어가주세요.



Classic 선택화면

로딩화면

자, 로딩이 완료되었습니다. 여러분만의 클라우드 컴퓨터입니다!



1. 폴더 구조를 볼 수 있는 공간입니다. 우측 상단에 보시면 새로고침 버튼이 있어요. 콘솔창에서 뭔가 작업을 했는데 보이지 않는다면 새로고침을 해보시기 바랍니다.
2. txt, html, py파일 등 다양한 파일을 edit 할 수 있는 공간입니다.
3. 콘솔창입니다. 우리는 대부분의 명령어를 이곳에 입력하게 됩니다.

- 오른쪽에는 협업을 위한 공간이 하나 열렸을 텐데, 우리는 사용하지 않을 것이니 접어둡시다! 이 공간은 채팅 등 협업을 위한 다양한 기능을 제공합니다.

자, 이제 간단한 실습을 해보도록 합시다.

```python
print('Welcome to jejucodingbasecamp!! :)')
```

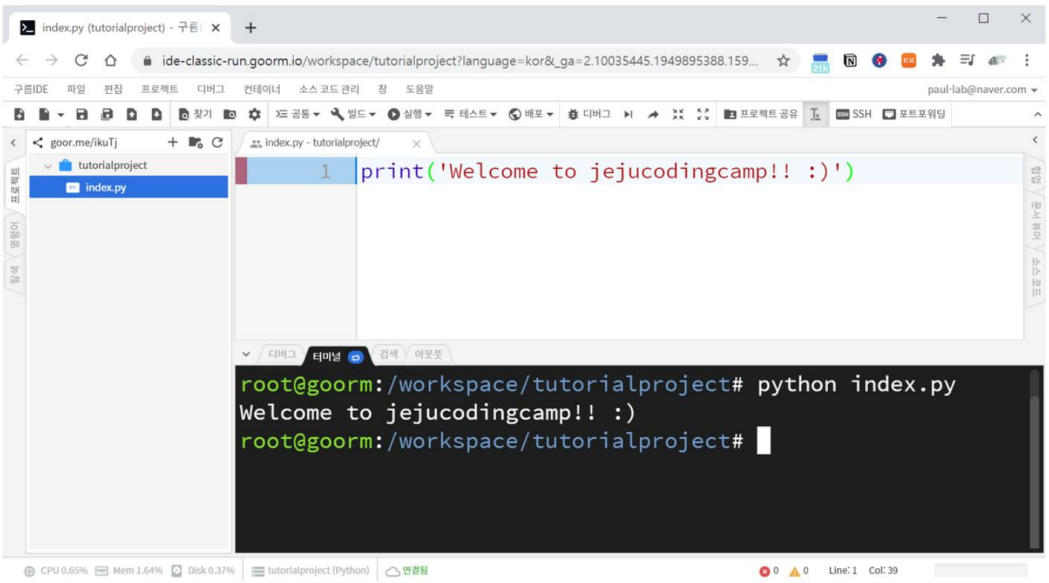
Python

```
python index.py
```

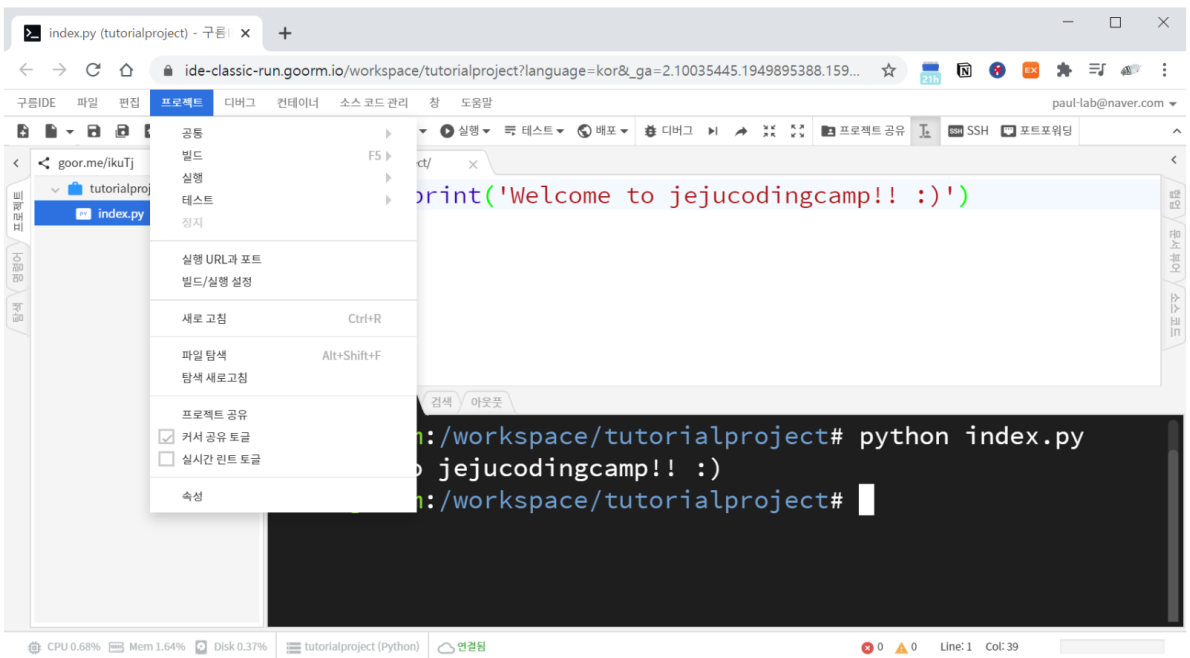
Plain Text

1. 왼쪽 프로젝트 바에서 `index.py`를 클릭합니다. 그러면 오른쪽 창에 해당 파일의 소스코드가 보입니다. 안에 있는 내용을 수정해보세요.
2. 안에 있는 내용을 수정하면 상단 화살표로 그려 넣은 곳이 `*`로 바뀝니다. 해당 표시는 저장이 안 되었다는 표시입니다. `Ctrl + S`를 누르시면 저장이 되고 다시 `X`로 돌아옵니다.
3. 저장이 되신 다음 아래 콘솔창에서 python index.py를 입력해보세요. 작성하신 문서가 실행됩니다!

실행되는 화면입니다.



프로젝트를 누르시고 실행 URL과 포트를 눌러보세요. 앞으로 사용될 URL과 포트입니다!



- 만약 등록이 되어 있지 않다면 위에서 원하는 url을 입력하고 PORT는 80으로 하여 등록버튼을 눌러주세요.
- 프로젝트를 클릭한 화면인 위 화면에서 `실시간 린트 토글` 체크박스 버튼이 빠져있죠? 이렇게 빼주시면 Error를 잡지 않습니다. 구름IDE에서 아직은 제대로 Error를 잡지 못하니 이걸 빼줍시다!

5. Ngnix 설치 및 구동(선택 사항)

서버가 어떻게 작동하는지 알기 위해서 웹 서버(HTML, CSS, JS, PHP Serving)를 서버로 구동시켜 보겠습니다. 동작 방식은 이러합니다. 누군가 우리 URL로 접속하면, 해당되는 페이지를 서빙하는거죠.

* 여기서 부터는 선택사항입니다. 안해보셔도 괜찮습니다.

```
root@goorm:/workspace/컨테이너명# python --version
```

Plain Text

```
root@goorm:/workspace/컨테이너명# mkdir web
```

Plain Text

```
root@goorm:/workspace/컨테이너명# cd web
```

Plain Text

```
root@goorm:/workspace/컨테이너명/web# sudo apt-get update
```

Plain Text

```
root@goorm:/workspace/컨테이너명/web# sudo apt-get install nginx
```

Plain Text

```
root@goorm:/workspace/컨테이너명/web# vi /etc/nginx/sites-available/default
```

Plain Text

vi 에디터를 이용하여 `root /var/www/html;` 을 `root /workspace/컨테이너이름/web;` 으로 고치세요.
(i를 누르면 편집이 됩니다. 편집을 다 하신 후에는 `ESC` , `:` , `wq!` 를 순서대로 입력하세요.)

```
root@goorm:/workspace/컨테이너명/web# sudo service nginx start
```

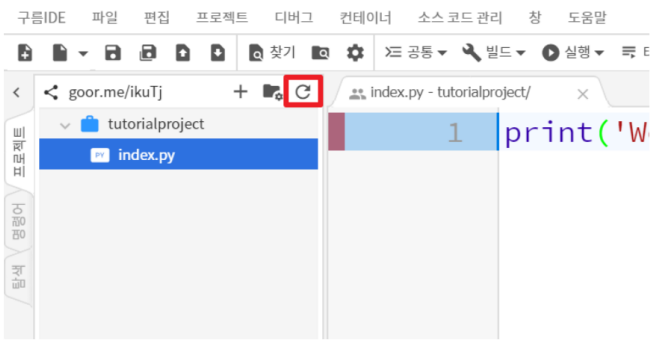
Plain Text

멈추고 싶을 때에는 sudo service nginx stop를 입력합니다.

```
root@goorm:/workspace/컨테이너명/web# sudo service nginx stop
```

Plain Text

작업하신 파일이 안보이실 때에는 새로고침 버튼을 클릭해주세요.



구름IDE에서 HTML, CSS, Javascript을 순차적으로 실행해보도록 하겠습니다.

기본 HTML

```HTML
<html>
<head>
</head>
<body>
  <h1>test</h1>
</body>
</html>
```

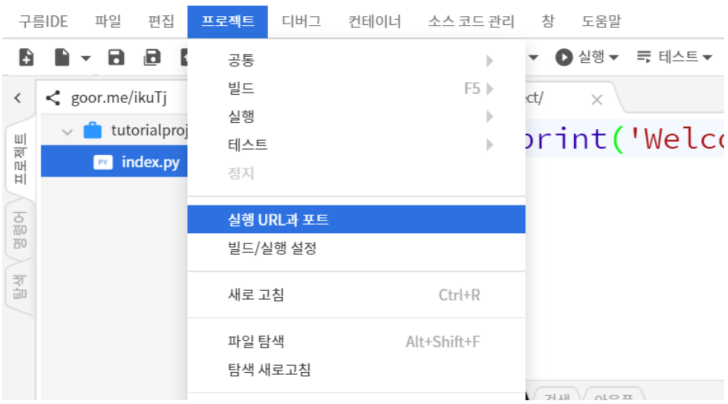
CSS 포함

```HTML
<html>
<head>
  <style>
    h1{color:blue;}
  </style>
</head>
<body>
  <h1>test</h1>
</body>
</html>
```

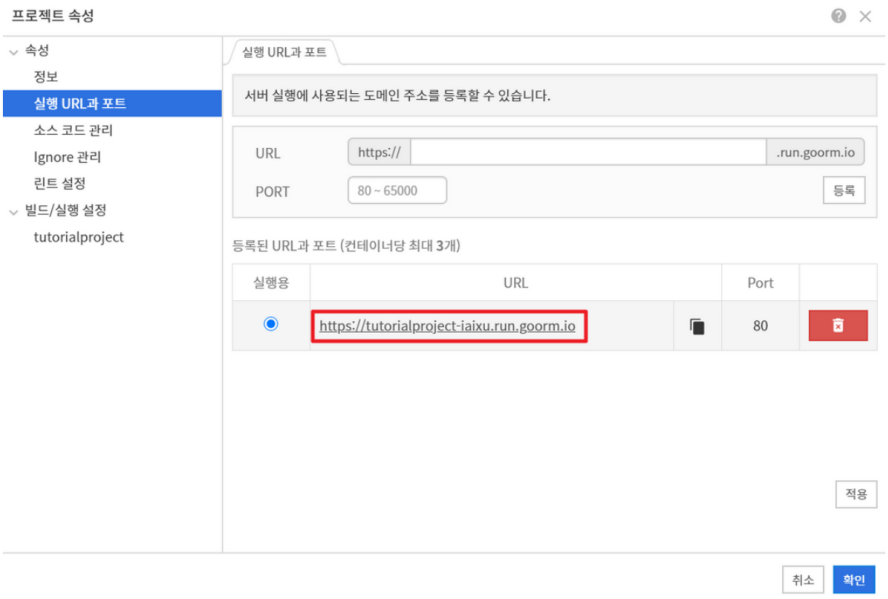
JavaScript 포함

```HTML
<html>
<head>
  <style>
    h1{color:blue;}
  </style>
</head>
<body>
  <h1 id="hojun">test</h1>
  <script>
    document.getElementById("hojun").innerHTML = "Hello World";
  </script>
</body>
</html>
```

자, 이렇게 저장해주시고, 프로젝트에 `실행 URL과 포트`를 눌러주세요.



이제 URL을 클릭하시고 실행이 되는지 확인해주세요!



DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day 1

HTML & CSS

1. HTML의 기초
2. HTML5
3. CSS
4. Code
5. CheatSheet

1. HTML의 기초

HTML과 CSS

HTML과 CSS는 무엇일까요? 그리고 무슨 차이가 있을까요?

HTML은 Hyper Text Markup Language의 약자로, **Hyper Text**(하이퍼텍스트)는 링크로 연결된 텍스트 페이지, **Markup Language**(마크업 언어)는 태그 등을 이용해서 문서를 작성하도록 도와주는 언어를 의미합니다. 확장자명은 .html 입니다.

CSS는 Cascading Stlye Sheets의 약자로, 웹 페이지의 전반적인 스타일을 작성해 저장해두는 시트를 의미합니다. 확장자명은 .css 입니다.

홈페이지 시연 http://studiomeal.com/

2. XHTML과 HTML5

XHTML이란 웹페이지를 제작하기 위해 사용되는 HTML4(지금 사용하고 있는 HTML5의 이전 버전)을 XML에 맞도록 재정의한 언어라고 할 수 있습니다.

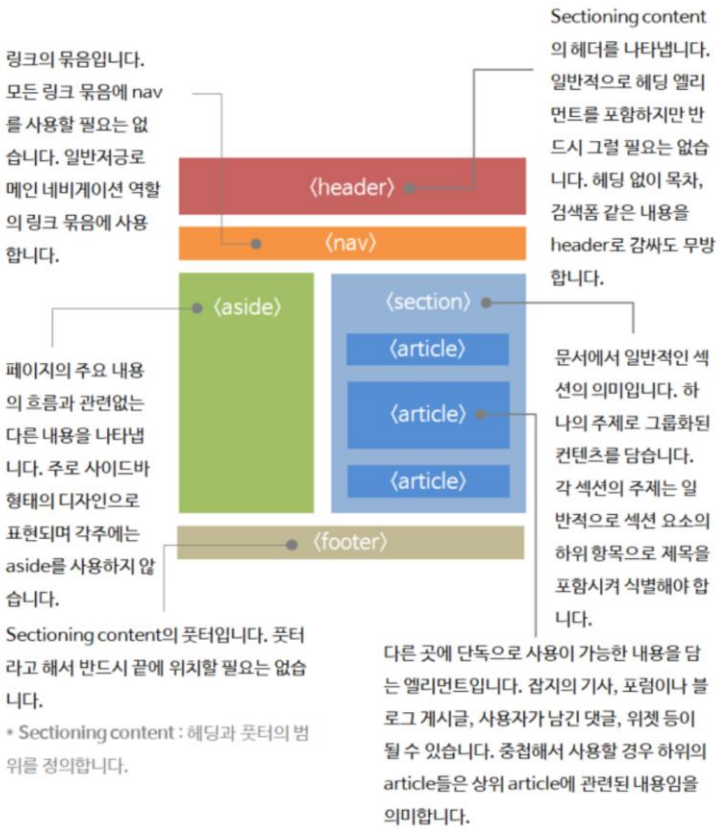
HTML5보다 조금 더 구조화된 형식과 엄격한 문법을 가지고 있는데요,

구체적으로 어떤 차이가 있는지 아래 표를 살펴봅시다.

	XHTML1.0	HTML5
DOCTYPE (독타입, DTD; Document Type Declaration)	〈!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"〉	〈!DOCTYPE html〉
character set (캐릭터 셋, charset)	〈meta http-equiv="content-type" content="text/html; charset=UTF-8"〉	〈meta charset="UTF-8"〉
type 표시 여부	생략 및 단축 불가 〈link type="text/css" rel="stylesheet"〉	생략 및 단축 가능 〈link rel="stylesheet"〉
태그 닫기 여부	단독으로 쓰이는 태그는 /〉으로 닫아야함 〈br /〉 〈hr /〉	닫아줄 필요 없음 〈br〉 〈hr〉

- **DOCTYPE** : 웹페이지의 맨 처음에 선언되어 어떤 종류의 html인지 웹 브라우저에게 알려줍니다.
- **meta charset** : 해당 페이지의 인코딩 방식을 나타냅니다.

HTML5 태그



`<h1> … <h6>`

Sectioning content의 제목을 나타냅니다. 서브타이틀로 사용하면 안됩니다.

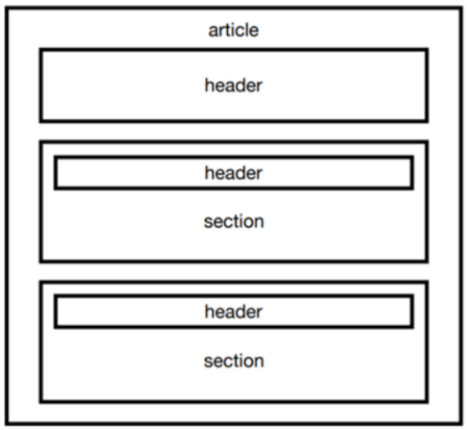
- 틀린 예

```HTML
<header>
<h1>집에서 맛있는 피자 만들기</h1>
<h2>주위에서 쉽게 구입할 수 있는
재료를 중심으로</h2>
</header>
```

- 좋은 예

```HTML
<header>
<h1>집에서 맛있는 피자 만들기</h1>
<p>주위에서 쉽게 구입할 수 있는
재료를 중심으로</p>
</header>
```

앞서 배운 태그를 사용해서 레이아웃을 만들어 볼까요? 왼쪽 화면의 레이아웃을 코드로 작성하면 오른쪽과 같습니다.




```html
<article>
  <header>
    내용
  </header>
  <section>
    <header></header>
    내용
  </section>
  <section>
    <header></header>
    내용
  </section>
</article>
```

HTML

3. CSS

HTML를 예쁘게, 보기 좋게 꾸며주는 CSS를 적용하는 방법은 크게 4가지가 있습니다.

외부 CSS 파일 로드하기

```html
<head>
  <link rel="stylesheet" href="css/foo.css">
</head>
```

HTML

HTML안에 CSS 작성하기

```html
<head>
  <style>
    body {font-size: 14px;}
  </style>
</head>
```

HTML

인라인 스타일

```html
<body style="font-size: 14px;">
```

HTML

CSS 파일 안에 CSS 포함하기

```
@import "foo.css";
```

HTML

에릭 마이어의 reset CSS https://meyerweb.com/eric/tools/css/reset/

Normalize.css https://necolas.github.io/normalize.css/

각 브라우저들에 자체적으로 적용되어 있는 기본 스타일(여백 등)을 초기화 시켜서 원하는 디자인을 적용하기 쉽게 만들어 줍니다. 위의 두가지 방법을 일반적으로 많이 사용하며, 프로젝트에 맞게 직접 만들어 쓰기도 합니다.

```
/* 선택자 */
body {
  background-color: red;
      /* 속성 */  /* 값 */
}
```

CSS

▲ body의 background-color를 red로 설정

```
/* 선택자 */
header a {
  display: inline-block;
 /* 속성 */   /* 값 */
}
```

CSS

▲ header에 포함되어 있는 a의 display를 inline-block으로 설정

아래의 내용 외에도 많은 선택자들이 있지만, 공부를 처음 시작할 때 우선 알면 좋은 것들 위주로 추려보았습니다.

- `*` : 문서 내의 모든 엘리먼트를 선택.
- `body` , `header` , `h1` , `section` 등 해당 이름을 가진 html 엘리먼트를 선택.
- `#foo` id 선택자 : `<div id="foo">` 의 형태로, id는 중복이 불가능한 유일한 값.
- `.foo` class 선택자 : `<div class="foo">` 의 형태로, 중복이 가능. class로 자주 쓰는 스타일을 정의해두고 재사용.
- `input[type="text"]` : input 엘리먼트 중 type 속성의 값이 text인 엘리먼트를 선택.

`.foo > p` : class="foo"인 엘리먼트의 직계 자식 엘리먼트 중 p를 선택. 문서 내의 모든 엘리먼트를 선택.

.foo > p 설명 예시 code

```html
<section class="foo">
  <header>
    <h1>집에서 맛있는 피자 만들기</h1>
    <p>주위에서 쉽게 구할 수 있는 재료를 중심으로</p>
  </header>
  <p>AAAAA</p> <-.foo>p
</section>
```

HTML

.foot:first-child/.foo:last-child/.foo-nth-child(n) 설명 예시 code

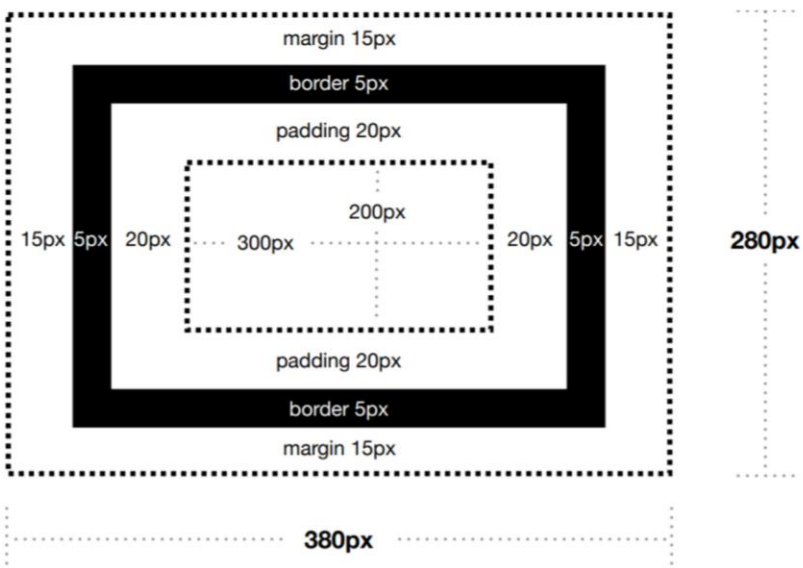
```html
<ul>
  <li class="foo">1</li>    <!-- .foo:first-child -->
  <li class="foo">2</li>
  <li class="foo">3</li>    <!-- .foo:nth-child(3) -->
  <li class="foo">4</li>
  <li class="foo">5</li>    <!-- .foo:last-child -->
</ul>
```

HTML

.foo:first-child : class="foo"인 엘리먼트 중 첫번째 자식인 엘리먼트를 선택

.foo:last-child : class="foo"인 엘리먼트 중 마지막 자식인 엘리먼트를 선택

.foo:nth-child(n) class="foo"인 엘리먼트 중 n번째 자식인 엘리먼트를 선택

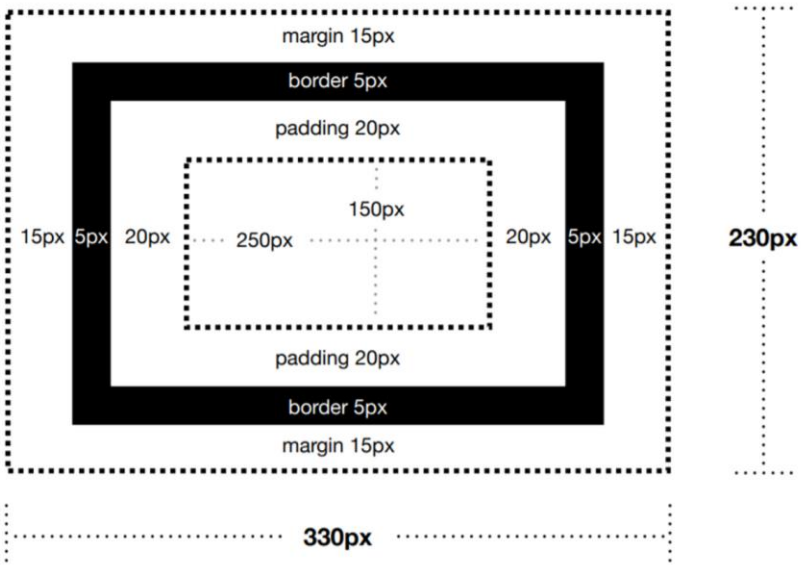


box-sizing: content-box (기본값)

width: 300px; height: 200px; padding: 20px; border: 5px; soild black; margin: 15px;

```css
.box {
  box-sizing: content-box;
  width: 300px;
  height: 200px;
  padding: 20px;
  border: 5px solid black;
  margin: 15px;
}
```

CSS



box-sizing: border-box (border까지 width에 포함)

width: 300px; height: 200px; padding: 20px; border: 5px; soild black; margin: 15px;

```css
.box {
  box-sizing: border-box
  /* border까지 width에 포함 */
  width: 300px;
  height: 200px;
  padding: 20px;
  border: 5px solid black;
  margin: 15px;
}
```

CSS

CSS에 대해 좀 더 상세한 내용을 알고 싶다면, 아래 링크를 참고하세요. 역시 무료책이며, 중급자로 도약하기 위한 상새한 내용이 담겨있습니다. 리디북스에 무료책으로도 공개가 되어있지만, 아래 Notion Page에서 보시는 것을 추천해드립니다.

🥷 ↗ Web Animation 1부 css Animation

🧱 ↗ Web Animation 2부 js Animation

4. Code

수업에서 사용했던 다양한 코드들을 명시해 두었습니다. 해당 코드는 설명이 달려있지 않으니 PDF로 인쇄하시는 분들은 **코딩노트를 활용**해 설명을 기입해주세요.

test.html

```html
<html>
    <head>
    </head>
    <body>
        <h1>Atom</h1>
        <p>github에서 만든 무료 Editer</p>
        <h1>sublime</h1>
        <p>많은 점유율을 차지하고 있는 부분무료 Editer</p>
        <h1>visual studio code</h1>
        <p>microsoft에서 만든 무료 editer, visual studio와 다른 에디터 입니다!</p>
    </body>
</html>
```

HTML

emmet_사용법.html

```
<!DOCTYPE html>
<html>
    <head>

    <title>emmet실습</title>
    </head>
    <body>
        <!--메모장도 사용해볼것-->
        h1
        h1+h1+p
        h1*3
        h1{hello world}*10
        h1#hojun
        h1.hoju
        (div>table>(tr>(td*2))*3)+(footer>p)
        div#one.c1.c2.c3
        ul>li.item$*5
        a{Click}
        lorem
        <!--
            Ctrl + \ : 토글 보이기
            Ctrl + O : 파일 열기
            Ctrl + P : 프로젝트 검색
            Ctrl + F : 열려 있는 파일 내에서 검색
            Ctrl + Shift + F : 열려 있는 전체 프로젝트 내에서 검색
            Ctrl + , : Settings
            Ctrl + N : 새로운 파일
            Ctrl + S : 파일 저장
            Ctrl + Shift + S  :다른 이름으로 저장
            Shift + del : 라인 지우기
            Ctrl + 클릭 : 여러줄 입력
        -->

    </body>
</html>
```

HTML

001.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
    </head>
    <body>
        <h1>글자 태그</h1><br>
        <p>H<sub>2</sub>0</p>
        <p>x<sup>2</sup>=4</p>
        <p>
```

```
        Lorem <br> ipsum
        <hr> dolor sit amet
        <a href="leehojun/a.html">click </a>
        <strong>hello</strong>
        adipisicing <b>elit</b>.
        Quae <em>quisquam </em>
        aperiam, <i> autem </i>, excepturi
        <mark> corrupti </mark> architecto facilis saepe
        </p>
    </body>
</html>
```

HTML

002.html

```
<!DOCTYPE html>
<html>
<head>
    <title> Document </title>
</head>
<body>
    <h1>Front-End</h1>
    <ol type="A">
        <li>HTML</li>
        <li>CSS</li>
        <li>Javacript</li>
        <li>Jquery</li>
        <li>Bootsrap</li>
    </ol>
    <h1>Back-End</h1>
    <ul>
        <li>Python</li>
        <li>Django</li>
    </ul>
    <dl>
        <dt>HTML</dt>
        <dd>마크업 언어입니다.</dd>
    </dl>
    <div>hello</div>
</body>
</html>
```

HTML

003.html

```
<!DOCTYPE html>
<html>
    <head><title>document</title></head>
    <body>
        <img src="img/a.jpg" alt="이미지 없음.">
        <iframe width="1280" height="720" src="https://www.youtube.com/embed/-iuX3r8PSzU"
frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-pi
cture" allowfullscreen>
        </iframe>
    </body>
</html>
```

HTML

004.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
    </head>
    <body>
        <input type="text"><br>
        <input type="password"><br>
        <input type="date"><br>
        <input type="time"><br>
        <input type="range"><br>
        <input type="color"><br>
        <input type="radio"><br>
        <input type="checkbox"><br>
        <input type="file"><br>
        <textarea name="name", rows="8", cols="80"> </textarea>
    </body>
</html>
```

HTML

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

005.html

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <style>
            table, th ,tr, td{
                border: 1px black solid;
                border-collapse: collapse;
            }
        </style>
    </head>
    <body>
        <table>
           <tr>
               <td>구분</td>
               <td>이름</td>
               <td>판매량</td>
           </tr>
           <tr>
               <td>1</td>
               <td>해리포터</td>
               <td>100</td>
           </tr>
           <tr>
               <td>2</td>
               <td>헝거게임</td>
               <td>200</td>
           </tr>
           <tr>
               <td>3</td>
               <td>반지의제왕</td>
               <td>300</td>
           </tr>
           <tr>
               <td colspan="2">총 판매량</td>
               <td>600</td>
           </tr>
       </table>
    </body>
</html>
```

HTML

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

006.html

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <link rel="stylesheet" href="a.css">
        <style>
        h1{
            color: blue;
        }

        </style>
    </head>
    <body>
    <h1 >hello</h1>
    <h2 style="color: red;">hello</h2>

    </body>
</html>
```

HTML

007.html

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <link rel="stylesheet" href="a.css">
        <style>
        #one{
            color: red;
        }

        #two{
            color: blue;
        }

        #three{
            color: green;
        }
        </style>
    </head>
    <body>
    <a href="#one">one으로 가라</h1>
    <a href="#two">two으로 가라</a>
    <a href="#three">three으로 가라</a>
    <h1 id="one">hello</h1>
```

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

```
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <h1 id="two">hello</h1>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <p>hello</p>
    <h1 id="three">hello</h1>

    </body>
</html>
```

HTML

008.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <link rel="stylesheet" href="a.css">
        <style>
        .one{
            color: red;
        }

        .two{
            font-size: 20px;
        }

        .three{
            color: green;
        }
        </style>
    </head>
```

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

```
    <body>
    <h1 class="one two">hello</h1>
    <h1 class="two">hello</h1>
    <h1 class="three">hello</h1>

    </body>
</html>
```

HTML

a.css

```
h1{color: red;}
```

HTML

009.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <style>
            p{
                font-size: 5em;
            }
        </style>
    </head>
    <body>
    <h1>고정 크기 단위</h1>
    <p> px, pt, in, cm ,mm</p>
    <h1>가변 크기 단위</h1>
    <p> em, % rem</p>
    <h1>hello</h1>
    <h1>hello</h1>
    </body>
</html>
```

HTML

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

010.html

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <style>
            h1{
               border: 1px solid black;
               padding: 2px 10px 20px 40px;
               margin:20px
            }
        </style>
    </head>
    <body>
    <h1>hello world</h1>
    </body>
</html>
```

HTML

011.html

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <style>
            .box{
                background-color: blue;
                width: 200px;
                height: 200px;
            }
            .one{
                opacity: 0.5;
            }
            .two{
                opacity: 0.3;
            }
        </style>
    </head>
    <body>
    <div class="box"></div>
    <div class="box one"></div>
    <div class="box two"></div>
    </body>
</html>
```

HTML

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

012.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <link href="https://fonts.googleapis.com/css?family=Unlock&display=swap" rel="sty
lesheet">
        <style>
        h1{font-family: 'Unlock', cursive};
        </style>
    </head>

    <body>
        <h1>helloworld</h1>
    </body>
</html>
```

HTML

013.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Document</title>
        <style>
        h1{
            color: #0000ff;
        };
        </style>
    </head>

    <body>
        <h1>helloworld</h1>
    </body>
```

HTML

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

5. CheatSheet

+ Add a view

태그 이름	설명	구분	대표 속성
<html> </html>	HTML문서를 만들때 사용	기본 태그	
<head> </head>	제목이나 표시되지 않는 정보를 설정	기본 태그	
<body> </body>	문서의 보이는 부분을 설정	기본 태그	bgcolor text link vlink alink
<title> </title>	제목 표시줄에 들어가는 내용을 설정	기본 태그	
<pre> </pre>	텍스트를 있는 그대로 출력	글자태그	
...	제목 텍스트 설정. 숫자가 작아질수록 크기가 커짐.	글자태그	
	볼드 텍스트를 만드는데 사용	글자태그	
	이탤릭체 텍스트를 만드는데 사용	글자태그	
<tt> </tt>	타자기 스타일의 텍스트를 만드는데 사용	글자태그	
<code> </code>	모노스페이스인 소스코드를 정의하는데 사용	글자태그	
<cite> </cite>	인용문을 표시하는데 사용	글자태그	
<adress> </adress>	주소를 표시하는데 사용	글자태그	
OPEN	텍스트를 이탤릭체로 강조하는데 사용	글자태그	
	볼드 텍스트를 만드는데 사용	글자태그	
 	폰트의 크기 색상 등을 설정하는데 사용	글자태그	size color face
<a>	링크를 연결할 때 사용	링크태그	href name target
	새 단락을 만들 때 사용	컨텐츠 그룹 태그	
 	줄바꿈을 넣을 때 사용	컨텐츠 그룹 태그	
<blockquote> </blockquote>	들여쓰기를 통해서 인용문을 만들 때 사용	컨텐츠 그룹 태그	
	css로 블록 컨텐츠를 형식화 하는데 사용	컨텐츠 그룹 태그	
	css로 인라인 컨텐츠를 형식화 하는데 사용	컨텐츠 그룹 태그	
	순서가 없는 리스트를 만듭니다.	컨텐츠 그룹 태그	
	순서가 있는 리스트를 만듭니다.	컨텐츠 그룹 태그	start type
	각 리스트의 항목들을 포함합니다.	컨텐츠 그룹 태그	
<dl></dl>	정의 목록들을 만듭니다.	컨텐츠 그룹 태그	
<dt>	정의되는 용어의 제목을 만듭니다.	컨텐츠 그룹 태그	
<dd>	정의되는 용어의 설명을 만듭니다.	컨텐츠 그룹 태그	
<hr>	수평선을 삽입합니다.	미디어태그	size width noshade
<img>	이미지를 표시할 때 사용	미디어태그	src width height
<form></form>	폼을 정의하는데 사용	폼 태그	src align border height width alt
<select>	다양한 메뉴 형태를 만드는데 사용	폼 태그	multiple name size
<option>	각 매뉴 항목들을 배치하는데 사용	폼 태그	
<textarea>	텍스트 박스 영역을 만드는데 사용	폼 태그	name cols rows
<input>	다양한 입력양식을 만드는데 사용	폼 태그	type name value checked
<table></table>	표를 만드는데 사용	테이블 태그	border cellspacing cellpadding width
<tr></tr>	표의 행들을 배치하는데 사용	테이블 태그	align valign
<td></td>	표의 각 셀들을 배치하는데 사용	테이블 태그	align valign rowspan colspan nowrap
<th></th>	테이블 헤더를 배치하는데 사용	테이블 태그	

+ New

COUNT 38

Day2

JavaScript

1. Javascript의 기초
2. 변수
3. Javascript의 연산
4. 조건문
5. 함수
6. DOM(Document Object Model)
7. HTML FORM
8. 이벤트
9. 정규표현식
10 Javascript CheatSheet

1. Javascript의 기초

1.1 Javascript란?

- HTML을 프로그래밍으로 제어한다
- 웹브라우저가 해석해서 실행할수 있는 유일한 프로그래밍 언어
- 요즘에는 게임프로그래밍, 서버프로그래밍 등 다양한 분야에서 쓰여지고 있음.

자바스크립트는 사용자의 이벤트를 받고 내장 객체들을 이용한 CSS나 태그의 스타일 관련 속성, 날짜, 텍스트 등을 조작 할 수 있습니다.

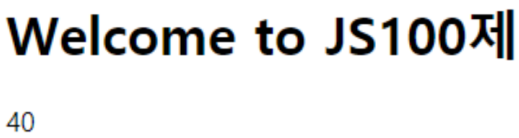
```css
.box{
  height: 300px;
  width: 300px;
  background-color: blue;
  transition: all 2s;
}
```

CSS

```html
<!-- javascript 소개 -->
<h1>자바스크립트로는 어떤 것을 할 수 있나요?<h1>
<button type="button"
onclick="document.getElementById('demoOne').innerHTML = Date()">
Click me (One)</button>

<button type="button"
onclick="document.getElementById('demoTwo').innerHTML = 'hello world'">
Click me (Two)</button>

<button type="button"
onclick="document.getElementById('demoThree').style.backgroundColor = 'red'">
Click me (Three)</button>

<p id="demoOne"></p>
<p id="demoTwo"></p>
<p class="box" id="demoThree"></p>
```

HTML

```HTML
<h1 id='title'>Hello World</h1>
<script>
  // https://ko.javascript.info/ 소개
  // https://www.w3schools.com/ 소개

  // 자바스크립트로 출력하는 방법
  // let age = 10; //지금은 다루지 않습니다.
  // const pi = 3.141592; // 지금은 다루지 않습니다.
  var name = '사용자';
  var age = 20;

  document.getElementById('title').innerHTML = 'Welcome to JS100제';
  document.write(20 + 20);
  window.alert(name);
  console.log(age);
  // 한 줄 주석
  /* 주석
  입니다. */
</script>
```



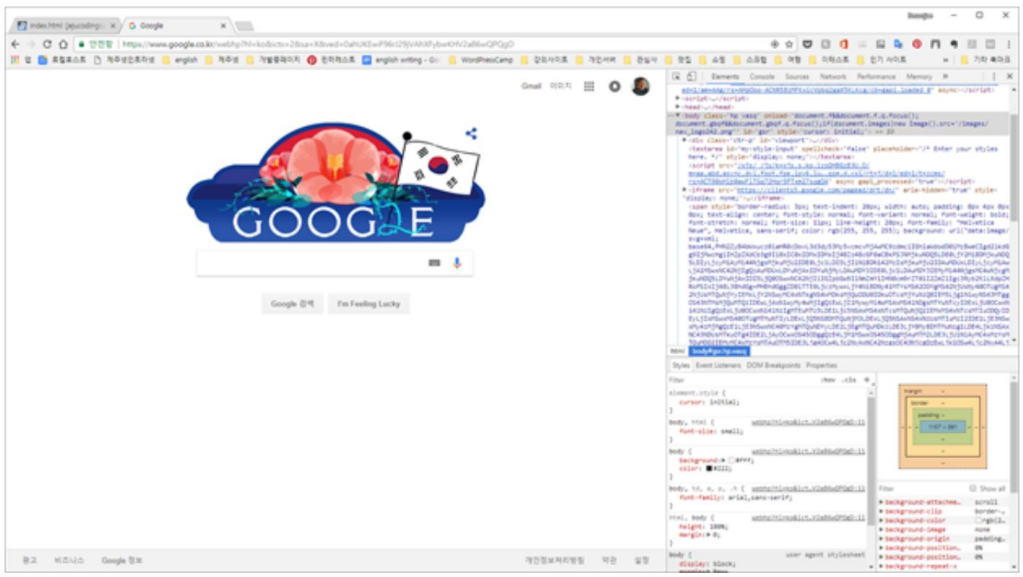
40

1.2 실습방법

- goorm IDE를 사용해 소스코드를 작성하고 실행합니다.
- 브라우저는 Chrome을 사용합니다.

1.3 환경설정

개발자 도구를 열어봅시다. 크롬에서 단축키 F12 또는 `Ctrl + Shift + i`를 누르면 개발자 도구가 열려요. 맥북에서는 `Cmd + Opt+ i`



1.4 HTML에서 Javascript 로드하기

INLINE

HTML내에 javascript를 포함하고 로드합니다.

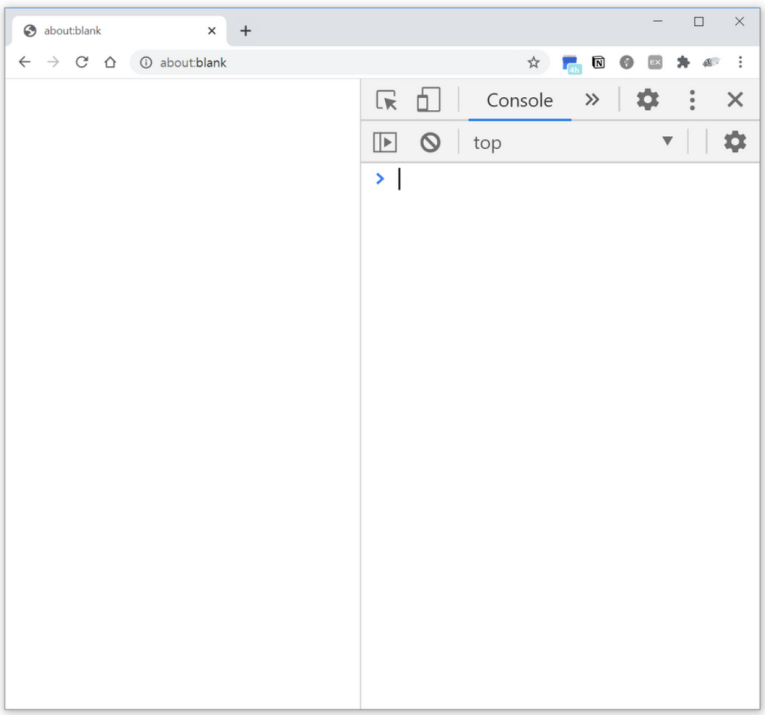
SCRIPT

외부 파일로 저장을 하고 로드합니다.

1.5 Javascript로 출력하기

크롬의 **콘솔창**은 Ctrl + Shift + i 버튼을 누르면 나오는 창입니다. 다음은 크롬에서 `about:blank` 페이지에 접속한 화면입니다.

이 페이지는 비어있는 페이지이며 간단한 javascript 연습을 하기 충분합니다.



2. 변수

2.1 변수란?

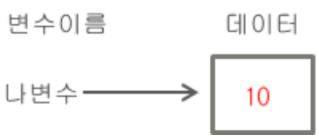
변수는 '**변할 수 있는 수**', '**변할 수 있는 정보**'라는 뜻입니다.

프로그램을 만드는 데 필요한 숫자나 문자와 같은 데이터를 보관할 공간이 필요한데, 물건을 잠시 보관하는 상자 같은 역할을 하는 것이 변수입니다.

변수는 선언하고 할당하고 사용할 수 있습니다. 또한, 변수는 '변할 수 있는 수'이므로 지정된 **값을 계속 바꿀 수 있습니다.**

```html
<!DOCTYPE html>
<html>
<head>
  <title>변수</title>
</head>
<body>
  <script>
  var 나변수 = 10;
  </script>
</body>
</html>
```

HTML



변수명을 정할 때

- 문자와 숫자, 기호 $와 _(언더바)만 사용 가능합니다.
- 첫 글자는 숫자가 될 수 없습니다.
- 대소문자 구별이 안됩니다.
- 예약어가 쓰일 수 없습니다.

```javascript
var a;
var my_name;
```

JavaScript

자바스크립트에서 변수를 선언할 때에는 앞에 var을 써줍니다. 변수의 타입과 var 외 다른 키워드는 뒤에서 더 자세하게 알아보도록 하겠습니다.

```
a = 1;
my_name ="lee";
```

JavaScript

변수를 할당하는 방법은 다음과 같습니다.

```
console.log(a);
console.log(my_name)
```

JavaScript

변수에 저장된 값을 프로그램을 통해서 확인할 수 있습니다. 구글 chrome을 통해서 콘솔창에 찍히는 값을 확인해 보도록 하겠습니다.

2.2 변수의 자료형

```
var 변수하나 = 20;
var 변수둘 = 10;

document.write(변수하나 + 변수둘, '<br>');
document.write(변수하나 * 변수둘, '<br>');
document.write(변수하나 / 변수둘, '<br>');
document.write(변수둘**2, '<br>');  // a**2 = a*a
```

JavaScript

30
200
2
100

우리가 인터넷으로 물건을 주문하면 주문한 물건의 크기에 걸맞는 택배상자에 담아서 오게 됩니다.

이와 같이 변수도 그 크기에 맞는 형식을 가지고 있습니다. 아래 간단하게 몇가지만 요약해 보았어요. 모든 자료형이 다 정리되어 있는 것은 아닙니다.

number	숫자를 담는 상자
string	문자열을 담는 상자
boolean	true, false를 담는 상자
array	여러 자료형을 한꺼번에 담을 수 있는 상자
function	함수를 담는 상자

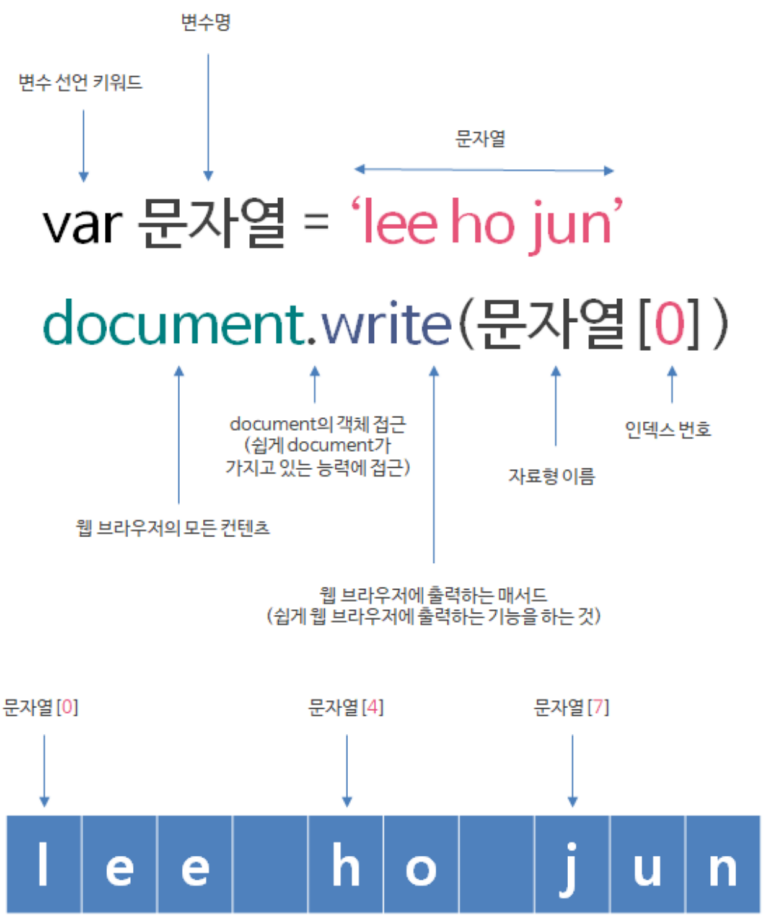
변수의 자료형은 다양한 데이터를 용도에 맞게 쓰기 위해서 입니다. 보통 언어에서는 변수의 자료형과 함께 변수를 선언하지만 자바스크립트는 자료형을 함께 쓸 필요는 없습니다.

```
//ES6
const 이름 = '이호준';
const 소속 = '바울랩 대표입니다.';

let 주소 = '서울';
주소 = '제주';

document.write(이름, '<br>');
document.write(소속, '<br>');
document.write(주소, '<br>');

document.write(소속[0], '<br>');
document.write(소속[1], '<br>');
document.write(소속[2], '<br>');
```

JavaScript

이호준
바울랩 대표입니다.
제주
바
울
랩

여기서 문자열 자료형에 대해 좀 더 알아보고 가도록 하겠습니다. 문자열은 `작은따옴표('예시')`나 `큰따옴표("예시")`로 둘러싼 것을 말합니다.



2.3 자료형에 대해 조금 더 알아봅시다

변수의 자료형에 대하여 좀 더 폭넓게 알아보도록 하겠습니다. 여기서 몇가지 메서드를 사용하는데, 메서드 종류를 다 암기하실 필요는 없습니다. 자연스럽게 많이 사용하시는 것은 암기가 되실거에요!

```javascript
//숫자형 (number)
var num = 10;

document.write(num, '<br>');  // 10
document.write(num/3, '<br>');  // 3.3333..
document.write(parseInt(num/3), '<br>');  // 3

/* 숫자형의 사칙연산 */
document.write("더하기 : ", 2 + 2.5, '<br>');  // 4.5
document.write("빼기 : ", 5 - 7, '<br>');  // -2
document.write("곱하기 : ", 3 * 2, '<br>');  // 6
document.write("나누기 : ", 2/2, '<br>'); // 1

/*특수 숫자 값*/
document.write("무한대 : ", 1/0,'<br>'); // Infinity
document.write( "숫자가 아님" / 2 );
// NaN, 문자열을 숫자로 나누면 오류가 발생합니다.
```

JavaScript

```javascript
//문자열 (string)
var day = 7;

document.write("오늘은 " + day + "일 입니다.", '<br>');
document.write('오늘은 ' + day + '일 입니다.', '<br>');
document.write(`오늘은 ${day}일 입니다.<br>`);
// 오늘은 7일 입니다.
```

JavaScript

```javascript
//논리형 (boolean)
var logic1 = true;
var logic2 = false;
var logic3 = 30 > 20;
var logic4 = 30 < 20;

document.write(logic1, '<br>');  //true
document.write(logic2, '<br>');  //false
document.write(logic3, '<br>');  // true
document.write(logic4, '<br>');  // false
```

JavaScript

```javascript
//null
var a = '';
var b = null;

document.write(a, '<br>');  //
document.write(b, '<br>');  // null

//undefined
var c;
document.write(c, '<br>')  // undefined
```

JavaScript

```javascript
//객체 (object)
var person = {
  name: '이호준',
  age: '비밀',
  married: true
}

person.소속 = '바울랩';

document.write(person.name, person['name'], '<br>');
document.write(person.age, person['age'], '<br>');
document.write(person.소속, person['소속'], '<br>');
```

JavaScript

```javascript
//배열 (array)
var 배열 = ['사과', '수박', '복숭아', '딸기', '바나나'];

document.write(배열[0], '<br>');
document.write(배열[1], '<br>');
document.write(배열[2], '<br>');
document.write(배열[3], '<br>');
document.write(배열[4], '<br>');
```

JavaScript

```javascript
//배열 내장함수
var 배열하나 = ['사과', '수박', '복숭아', '딸기', '바나나'];
var 배열둘 = ['메론', '체리', '한라봉'];

document.write(배열하나, '<br>');
배열하나.pop();
document.write(배열하나, '<br>');
배열하나.push('감귤');
document.write(배열하나, '<br>');

document.write(배열하나.toString(), '<br>');
document.write(배열하나.join('*'), '<br>');
document.write(배열하나.shift(), '<br>');
document.write(배열하나.slice(0, 2), '<br>');
document.write(배열하나.concat(배열둘), '<br>');
document.write(배열하나.reverse(), '<br>');
document.write(배열하나.sort(), '<br>');
```

JavaScript

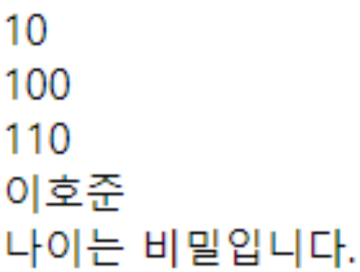
3. Javascript의 연산

3.1 Javascript의 연산

자바스크립트는 HTML, CSS와는 다르게 다양한 산술, 대입 등의 연산자를 통해 숫자, 문자 등을 출력할 수 있습니다.

```javascript
// 연산
var x, y, z;
x = 5;
y = 9;
y++; // 10
x--; // 4
x--; // 3
x = x + 2; // 5
x += 5; // 10
y *= 10; // 100
z = x + y; //110

document.write(x, '<br>'); // 10
document.write(y, '<br>'); // 100
document.write(z, '<br>'); // 110

var 이름_성 = '이';
var 이름 = '호준';
var 나이 = '비밀';

document.write(이름_성+이름, '<br>');
document.write('나이는 '+나이+'입니다.', '<br>');

//ES6
document.write(`${이름_성}${이름}, <br> 나이는 ${나이}입니다.`);
```

JavaScript



x, y, z로 선언된 변수에 초기 값으로 x는 5, y는 9를 대입하였습니다.

x는 -- 후위 감소 산술 연산 2회 , += 덧셈 할당 연산 2회를 통해 변수의 값이 5가 되었습니다.
y는 ++ 후위 증가 산술 연산 1회, *= 곱셈 할당 연산을 통해 변수의 값이 100이 되었습니다.
z는 x(10)과 y(100)를 더한 값으로 110이며, x, y, z는 write()함수를 통해 출력할 수 있습니다.

자바스크립트에서는 피연산자가 하나라도 문자열이면 주로 결합을 수행하게 됩니다.

이름_성, 이름, 나이로 선언된 변수에 초기 값으로 문자열 '이'와 '호준', '비밀'을 대입하였습니다.
write()함수를 통해 결합된 결과값 문자열 '이호준'과 '나이는 비밀입니다.'를 출력합니다.

- 피연산자의 type coercion 은 연산자에 따라 달라집니다. 상세 내용은 https://medium.freecodecamp.org/js-type-coercion-explained-27ba3d9a2839를 참고해주세요.
  예) '1' & 0

산술, 할당 연산자	
변수++	변수 값에 1 더하기(후위)
++변수	변수 값에 1 더하기(전위)
변수--	변수 값에 1 빼기(후위)
--변수	변수 값에 1 빼기(전위)
+=	변수 값에 더한 결과값을 대입
-=	변수 값에 뺀 결과값을 대입
*=	변수 값에 곱한 결과값을 대입
/=	변수 값에 나눈 결과값을 대입
%=	변수 값에 나머지를 구한 결과값을 대입

비교연산

객체의 크고 작고를 비교하거나 맞고 틀림을 비교하는 연산을 비교연산이라고 합니다.

- **비교연산 CODE**

```javascript
var x, y;
x = 20;
y = 10;
document.write(x > y, '<br>');          // true
document.write(x < y, '<br>');          // false
document.write(x <= y, '<br>');         // false
document.write(x >= y, '<br>');         // true

document.write(x == 20, '<br>');        // true
document.write(x === '20', '<br>');     // false
document.write(x != y, '<br>');         // true
document.write(x != y*2, '<br>');       // false
```

JavaScript

true
false
false
true
true
false
true
false

관계연산의 결과는 true, false로 나온다는 것을 기억해 주세요.

관계비교연산자	
>	왼쪽 값이 더 클 경우 true
>=	왼쪽 값이 크거나 같을 경우 true
<=	왼쪽 값이 작거나 같을 경우 true
<	왼쪽 값이 작을 경우 true
==	동일한 값(10=='10'은 true)일 경우 true
===	동일한 값과 동일한 유형일 경우 true
!=	같지 않은 true

논리연산

논리연산 CODE

```javascript
document.write(true || true, '<br>');    //true
document.write(true || false, '<br>');   //true
document.write(false || true, '<br>');   //true
document.write(false || false, '<br>');  //false

document.write(true && true, '<br>');    //true
document.write(true && false, '<br>');   //false
document.write(false && true, '<br>');   //false
document.write(false && false, '<br>');  //false

document.write(!true, '<br>');           //false
document.write(!false, '<br>');          //true

var x = 5;
var y = 20;

document.write(x < 25 && y < 25, '<br>');     //true
document.write(x === 5 || y === 5, '<br>');   //true
document.write(!(x < y), '<br>');             //false
```

JavaScript

typeof 연산자

typeof는 변수의 데이터 타입을 반환하는 연산자입니다.

006.html

```
// 타입 (https://ko.javascript.info/types 참고)

document.write(typeof(5), '<br>');             //number
document.write(typeof(5.5), '<br>');           //number
document.write(typeof('js100제'), '<br>');     //string
document.write(typeof('1'), '<br>');           //string
document.write(typeof(x), '<br>');             //undefined
document.write(typeof(undefined), '<br>');     //undefined
document.write(typeof([1, 2, 3, 4]), '<br>');  //object
document.write(typeof({'one':'하나', 'two':'둘'}), '<br>'); //object
document.write(typeof(js), '<br>');            //function

function js(){
  return 0;
}

document.write("안녕?" / 2, '<br>');            //NaN
document.write(typeof("안녕?" / 2), '<br>');    //number
//number에는 일반적인 숫자 외 Infinity, -Infinity, NaN 포함
document.write(typeof(true), '<br>');          //boolean(true, false)

let test = null;
document.write(typeof(test), '<br>');          //object
//null은 'object'로 표시되지만 특수 값을 가지는 객체가 아닙니다.
//해당 오류는 호환성 유지를 위해 수정하지 않고 남겨둔 상황입니다.
```

JavaScript

006_1.html

```javascript
// 타입의 형 변환 (https://ko.javascript.info/type-conversions 참고)

let one = '1'
document.write(one + one, '<br>');
document.write(Number(one) + Number(one), '<br>');

let two = 2
document.write(two + two, '<br>');
document.write(String(two) + String(two), '<br>');

document.write(one + two, '<br>');     //자동 스트링 변환 : 12

let three = one + two;
document.write(typeof(three), '<br>'); //string

let four = 'leehojun' + two;           //leehojun2
document.write(four);

//boolean 형변환
document.write(Boolean(1), '<br>');      // true
document.write(Boolean(0), '<br>');      // false
document.write(Boolean('0'), '<br>');    // true (문자열 있음)

document.write(Boolean("hello"), '<br>'); // true
document.write(Boolean(""), '<br>');      // false (빈문자열)
```

JavaScript

4. 조건문

4.1 조건문

조건문은 조건이 참(`true`)인지 거짓(`false`)인지에 따라 코드를 수행할지 말지 판단합니다.

- if문 CODE

```javascript
//if

var test=5;
if(test < 10){
  //codes
}
```

JavaScript

- if-else문 CODE

```javascript
//if

var test=5;
if(test < 10){
  //codes
} else {
  //codes
}
```

JavaScript

- if문 CODE

```javascript
if(true){
  //조건이 참일 경우 실행
} else {
  //조건이 거짓일 경우 실행
}

var score = 85;
if (score >= 90) {
  document.write('용돈을 10만원 받았습니다.');
} else if(score >= 80){
  document.write('용돈을 1만원 받았습니다.');
} else if(score >= 70){
  document.write('용돈을 5천원 받았습니다.')
} else {
  document.write('용돈을 천원받았습니다.')
}
```

JavaScript

- Switch문 CODE

```
var day;
switch(new Date().getDay()){
  case 0:
    day = 'Sun';
    break;
  case 1:
    day = 'Mon';
    break;
  case 2:
    day = 'Tue';
    break;
  case 3:
    day = 'Wed';
    break;
  case 4:
    day = 'Thu';
    break;
  case 5:
    day = 'Fri';
    break;
  case 6:
    day = 'Sat';
}
document.write(day);
```

JavaScript

4.2 반복문

다음에는 반복문에 대해서 알아보도록 하겠습니다. 먼저 다음과 같은 배열이 존재한다고 가정해 봅시다.

- 배열 CODE

```
var cars = ["BMW", "Volvo", "Saab", "Ford", "Flat", "Audi"];

var text = "";
```

JavaScript

여기서 만약 배열 cars에 담긴 내용을 전부 text라는 변수에 넣고 싶다면 어떻게 해야 할까요? 지금까지 배운 내용으로는 다음처럼 길고 반복적인 작업을 통해 text에 배열의 내용을 넣을 수 있습니다.

- 기존 방식 CODE

```
text += cars[0] + "<br>";
text += cars[1] + "<br>";
text += cars[2] + "<br>";
text += cars[3] + "<br>";
text += cars[4] + "<br>";
text += cars[5] + "<br>";
```

JavaScript

- for문 CODE

```
var i;
for(i = 0; i < cars.length; i++) {
  text += cars[i] + "<br>";
}
```

JavaScript

하지만 반복문을 사용하면 오른쪽에 나온 것 처럼 간단하게 배열의 내용을 text에 넣을 수 있습니다.

```
//반복문
for(var i=0; i<10; i++){
  document.write(i, '<br>');
}

var j = 0;
var sum = 0;
while(j <= 10){
  sum += j;
  j++;
}
document.write(sum, '<br>');
document.write('<br>');

//구구단 예제
for(var x=1; x<10; x++){
  for(var y=1; y<10; y++){
    document.write(x+'*'+y+'='+x*y, '<br>');
  }
}
```

JavaScript

```javascript
//평균 구하기
var value1 = [100, 200, 50, 400, 900];
var value2 = [60, 40, 80, 30, 90];
function average(value){
  var sum = 0;
  for(var i=0; i<value.length; i++){
    sum += value[i];
  }
  return sum/value.length;
}

document.write('평균1 : ', average(value1)); //330
document.write('<br>');
document.write('평균2 : ', average(value2)); //60
```

JavaScript

```javascript
//최댓값 구하기
var value = [100, 200, 50, 400, 900];
function maximum(value){
  var max = 0;
  for(var i=0; i<value.length; i++){
    if(max < value[i]){
      max = value[i];
    }
  }
  return max;
}
document.write(maximum(value)); //900
document.write('<br>');

//Math.max.apply() 사용해서 최댓값 구하기
var max2 = Math.max.apply(null, value);
document.write(max2); //900
```

JavaScript

5. 함수

5.1 함수

함수는 입력, 출력, 기능을 하나로 묶어 재사용 할 수 있도록 만드는 것입니다. 자바스크립트는 실행 코드들이 들어있는 **독립 블록 단위**의 **객체**인 함수를 사용할 수 있습니다. 여기서 객체란 데이터와 그 데이터를 포함한 모든 동작에 대해 말합니다.

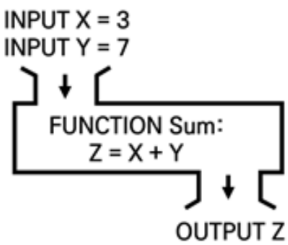
```HTML
<!-- 함수 -->
<p id="result"></p>
<script>
function myFunction(x,y){  //함수의 정의
  z = x + y  //함수의 기능
  return z;  //함수의 결과값
}
// 함수의 호출
document.getElementById("result").innerHTML = myFunction(4,3);
</script>
```

7

함수는 기본적으로

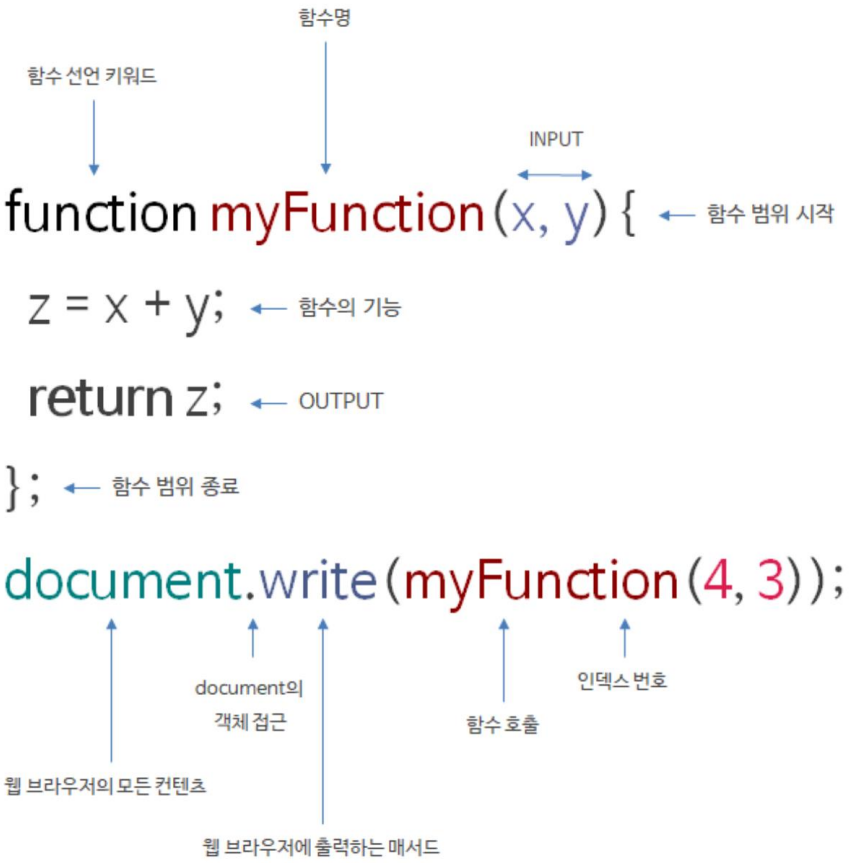
function( ) { };

형태의 구조를 가집니다.



소괄호()내에는 매개변수와 중괄호 {}내에는 실행코드와 return(결과) 값을 가지고 있습니다. 함수를 호출 하기 위해서는 선언된 함수의 매개변수 값을 인수로 전달합니다.

myFunction으로 선언된 함수에 인수로 4, 3을 전달하면 함수내 실행코드의 결과값인 7일을 출력 할 수 있습니다.




```html
<button onclick="test()">클릭 One!!</button>
<button onclick="document.write('hello world Two')">클릭 Two!!</button>
```

HTML

```javascript
// 함수
// 참고 : function은 python에 def과 같습니다.
// 읽어볼만한 문헌 : https://ko.javascript.info/function-basics

//함수선언
function sum(x, y){
  return x + y;
}
document.write(sum(20, 40));//함수호출

function test(){
  document.write('hello world One');
}
```

JavaScript

```javascript
//지역변수와 전역변수
//함수선언(또는 함수정의)
var z = 100; //z는 전역 변수(global variable)
function sum(x){ // x는 매개변수(parameter)
  let y = 50; // y는 지역변수(local variable)
  z = z + y;
  return x + y; // return이 없을 경우 undefined
}
// 아래 20이라는 값은 전달인자(argument)
document.write(sum(20), '<br>'); //함수호출
// Template literals 사용
//   document.write(`x는 ${x}`, '<br>'); // 지역변수
//   document.write(`y는 ${y}`, '<br>'); // 지역변수
document.write(`z는 ${z}`, '<br>'); // 전역변수
```

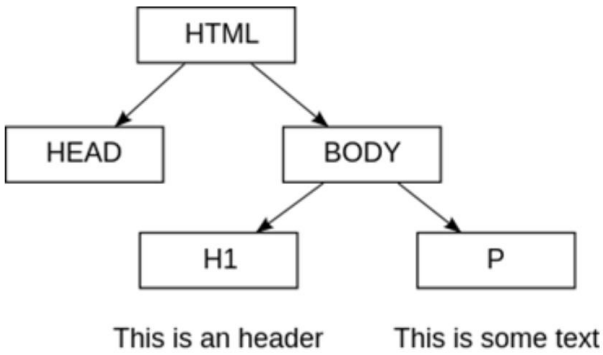
JavaScript

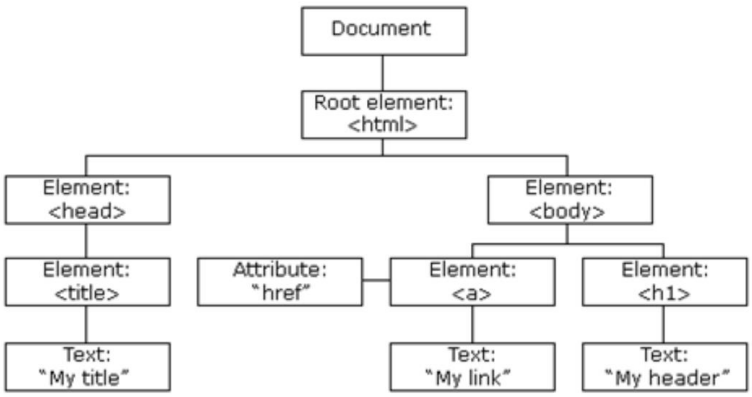
```javascript
// 화살표 함수(arrow function)
// 읽어볼만한 문헌 : https://ko.javascript.info/arrow-functions-basics

function sum(x, y){
  return x + y;
}

let sum_arrow = (a, b) => a + b;
```

JavaScript

6. DOM (Document Object Model)

6.1 DOM

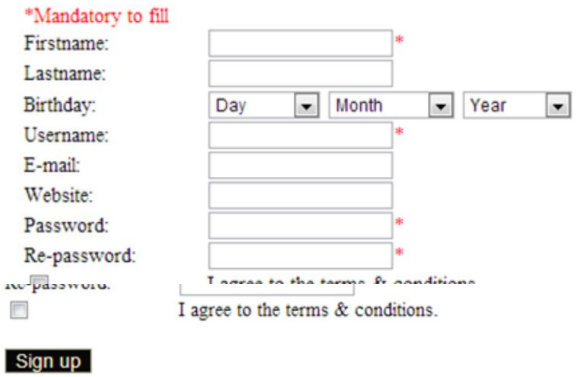




웹페이지가 로딩될 때 브라우저는 다음과 같은 객체들의 트리를 생성합니다. 이를 DOM이라 부릅니다. 자바스크립트는 이 DOM을 참조해 html요소를 제어할 수 있습니다.

7. HTML FORM

7.1 HTML FORM



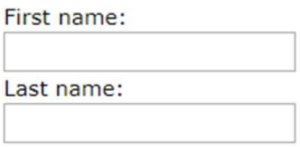
모두 인터넷을 하면서 다음과 같은 입력 폼을 본 적이 있을 것입니다. 이러한 입력 폼은 HTML의 <form>코드로 작성됩니다. 다음 코드를 atom에서 작성해봅시다.

- form 예시 CODE

```html
<form>
  First name : <br>
  <input type="text" name="firstname"><br>
  Last name : <br>
  <input type="text" name="lastname">
</form>
```

HTML

그런 다음 작성한 html 파일을 브라우저 상에서 확인해 보면 다음과 같은 화면이 보일 것입니다.



도전하기!

자바스크립트로 form에 입력된 내용을 가져와보세요!

8. 이벤트

- **클릭 이벤트 CODE**

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<body>
  <button id="click_btn" type="button">클릭해주세요!</button>

  <script>
  var click_btn = document.getElementById('click_btn');

  click_btn.addEventListener('click', click_event);

  //클릭 이벤트
  function click_event(){
    alert('클릭 이벤트 발생!');
  }

  /* 이벤트 종류
    onchange : HTML의 요소 값을 변경할 경우 사용하는 이벤트
    onclick : 사용자가 마우스를 눌렀다 뗀 경우 발생하는 이벤트
    onblur : 요소가 포커스를 잃었을 경우 발생하는 이벤트
    onmouseover : 마우스를 요소 위로 움직일 때 사용하는 이벤트
    onmouseout : 마우스를 요소 밖으로 움직일 때 사용하는 이벤트
    onkeydown : 키보드의 키를 누른 경우  발생하는 이벤트
    onload : 브라우저에서 페이지의 로딩이 끝났을 경우 발생하는 이벤트 */
  </script>
</body>
</html>
```

HTML

이벤트 종류

- onchange : HTML의 요소 값을 변경할 경우 사용하는 이벤트
- onclick : 사용자가 마우스를 눌렀다 뗀 경우 발생하는 이벤트
- onblur : 요소가 포커스를 잃었을 경우 발생하는 이벤트
- onmouseover : 마우스를 요소 위로 움직일 때 사용하는 이벤트
- onmouseout : 마우스를 요소 밖으로 움직일 때 사용하는 이벤트
- onkeydown : 키보드의 키를 누른 경우 발생하는 이벤트
- onkeydown : 키보드의 키를 누른 경우 발생하는 이벤트
- onload : 브라우저에서 페이지의 로딩이 끝났을 경우 발생하는 이벤트

- blur 이벤트 CODE

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<body>
  <form action="index.html" method="post">
    <label id="text_id">아이디</label>
    <input id="user_id" type="text"  />
    <label id="text_pw">비밀번호</label>
    <input id="user_pw" type="password"  />

    <div id="msg"> </div>
  </form>

  <script>
  var user_pw = document.getElementById('user_pw');

  user_pw.addEventListener('blur', blur_event);

  //비밀번호가 6글자 미만일 경우 메시지 출력
  function blur_event(){
    var msg = document.getElementById('msg');
    if(user_pw.value.length < 6){
      msg.textContent = '비밀번호는 6글자 이상이어야합니다.';
    } else {
      msg.textContent = '';
    }
  }
  </script>
</body>
</html>
```

HTML

- 마우스 이벤트 CODE - mouseover, mouseout

```html
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<style>
  div{
    border : solid 1px #dbdbdb;
    width: 150px;
    height: 150px;
  }
  #box {
  }
</style>
<body>
  <div id="box">
  </div>

  <script>
    var box = document.getElementById('box');

    box.addEventListener('mouseover', over);
    box.addEventListener('mouseout', out);

    function over(){
      box.style.backgroundColor = 'blue';
    }

    function out(){
      box.style.backgroundColor = 'white';
    }
  </script>
</body>
</html>
```

HTML

- **스크롤 이벤트**

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<style>
  div{
    border : solid 1px #dbdbdb;
    width: 150px;
    height: 150px;
  }
  #one {
    background-color: blue;
  }
  #one.on {
    position: fixed;
  }
  #two {
    background-color: red;
  }
</style>
<body>
  <div id="one"></div>
  <div id="two"></div>
  <div id="two"></div>
  <div id="two"></div>
  <div id="two"></div>
  <div id="two"></div>
  <div id="two"></div>

  <script>
  //요소를 선택하는 또 다른 방법 : querySelector
  var one = document.querySelector('#one');

  //스크롤 시 파란색 박스만 고정
  function scroll() {
    //pageYOffset : 스크롤바의 offset 반환
    //console.log(pageYOffset);

    if(pageYOffset >= 10){
      one.classList.add('on');
    } else {
      one.classList.remove('on');
    }
  }

  window.addEventListener('scroll', scroll);
  </script>
</body>
</html>
```

HTML

9. 정규표현식

> Computer Science는 어렵다. 그 중 에서도 정규표현식이 가장 어렵다.

라는 말이 있을 정도로 정규표현식은 어렵습니다. 이것을 왜 배워야 하는지, 어떻게 활용하는지 한 번 알아봅시다.

```
<!DOCTYPE html>
<html>
<head>
  <title>정규표현식</title>
</head>
<body>
  <script>
    let s = '0101000202010203048121 23';
    document.write(s.replace(/[^(1-9)]/gi,""));
    // 특정문자 제거 : s.replace(/\-/g,'');
    // 앞뒤 공백 제거 : s.replace(/^\s+|\s+$/g,'');
    // 문자열 내의 공백 제거 : s.replace(/\s/g,'');
    // 개행 제거 : s.replace(/\n/g,'');

    /*
    - `^` : 문자열의 시작
    - `$` : 문자열의 종료. 옵션에 따라 문장의 끝 또는 문서의 끝에 매치된다.
    - `.` : 임의의 한 문자
    - `[]`: 문자 클래스. 문자 클래스 안에 들어가 있는 문자는 그 바깥에서 하나의 문자로 취급된다.
        - `^` : 문자 클래스 내에서 ^는 not
        - `-` : ex) a-z는 a에서 z까지의 문자
    - `|` : or를 나타냄
    - `?` : 앞 문자가 없거나 하나 있음
    - `+` : 앞 문자가 하나 이상임
    - `*` : 앞 문자가 0개 이상임
    - `{n,m}` : 앞 문자가 `n`개 이상 `m`개 이하. `{0,1}` 은 `?`와 같은 의미다.
    - `{n,}` : 앞 문자가 `n`개 이상. 위의 형태에서 `m`이 생략된 형태이다. `{0,}` 이면 `*`와 같고 `{1,}` 이면 `+`와 같은 의미이다.
    - `{n}` : 앞 문자가 정확히 `n`개. `{n,n}` 과 같은 의미이다.
    - `()` : 하나의 패턴구분자 안에 서브 패턴을 지정해서 사용할 경우 괄호로 묶어주는 방식을 사용한다.
```

```
    - `\s` : 공백문자
    - `\b` : 문자와 공백 사이를 의미한다.
    - `\d` : 숫자 [0-9]와 같다.
    - `\t` : 탭문자
    - `\w` : 단어 영문자+숫자+_(밑줄) [0-9a-zA-Z_]문자 이스케이프는 대문자로 적으면 반대를 의미
한다.
    [a-z] : a ~ z 사이의 문자를 찾음
    [1-9] : 1 ~ 9 사이의 문자를 찾음
    [abc] : a, b, c중 하나를 찾음
    [^abc] : a, b, c를 제외한 문자를 찾음
    .z : 아무 문자 하나를 . 기호로 찾으며 z로 끝남을 의미
    a+ : a가 1개 이상을 의미함
    a* : a가 0개 또는 그 이상을 의미함
    s : 공백 문자를 찾음(스페이스, 탭 등), 대문자의 경우 아닌 문자를 찾음
    d : 숫자를 찾음, 대문자의 경우 아닌 문자를 찾음
    w : 알파벳 영문과 숫자와 언더바 _ 기호를 찾음, 대문자의 경우 아닌 문자를 찾음
    t : 탭 공간을 찾음
    g : 검색범위를 전역으로 확장
    i : 대소문자를 구분하지 않음
    gi : 검색 범위를 전역으로 확대하면서 대소문자를 구분하지 않음
    m : 여러줄을 동시에 매칭함
    */
  </script>
</body>
</html>
```

HTML

10. javascript CheatSheet

+ Add a view

cheat	설명	구분
var a = 1	숫자형 (number)	데이터 타입
var b = "cat"	문자열 (sting)	데이터 타입
var c = true	논리형 (boolean)	데이터 타입
var dog = { name : "Spot" , kind : "dog" }	객체 (object)	데이터 타입
function sum(a, b) { return a + b; }	함수 (function)	데이터 타입
var numbers = [1,2,3]	배열 (array)	데이터 타입
null	변수 값이 없음	데이터 타입
undefined	정의되지 않음	데이터 타입
symbol	인스턴스가 고유하고 불변인 데이터 형	데이터 타입
typeof	변수의 데이터 타입을 반환	typeof 연산자
var	변수를 선언. 추가로 동시에 값을 초기화	변수
let	블록 범위(scope) 지역 변수를 선언, 추가로 동시에 값을 초기화	변수
const	블록 범위 읽기 전용 상수를 선언	변수
변수++	변수 값에 1 더하기(후위)	산술 할당 연산자
++변수	변수 값에 1 더하기(전위)	산술 할당 연산자
변수--	변수 값에 1 빼기(후위)	산술 할당 연산자
--변수	변수 값에 1 빼기(전위)	산술 할당 연산자
+=	변수 값에 더한 결과값을 대입	산술 할당 연산자
-=	변수 값에 뺀 결과값을 대입	산술 할당 연산자
*=	변수 값에 곱한 결과값을 대입	산술 할당 연산자
/=	변수 값에 나눈 결과값을 대입	산술 할당 연산자
%=	변수 값에 나머지를 구한 결과 값을 대입	산술 할당 연산자
>	왼쪽 값이 더 클 경우 true	비교 연산자
>=	왼쪽 값이 크거나 같을 경우 true	비교 연산자
<=	왼쪽 값이 작거나 같을 경우 true	비교 연산자
<	왼쪽 값이 작을 경우 true	비교 연산자
==	동일한 값일 경우 true	비교 연산자
===	동일한 값과 동일한 유형일 경우 true	비교 연산자
!=	같지 않을 경우 true	비교 연산자
&&	왼쪽 값과 오른 쪽 값이 모두 true일 경우 true	논리 연산자
||	둘 중 하나라도 true일 경우 true	논리 연산자
!	true이면 false, false이면 true	논리 연산자
if (조건문) {내용}	조건문이 true일 경우 내용 실행	조건문
else if (조건문) {내용}	if 조건문이 false이고 조건문이 true일 경우 내용 실행	조건문
else {내용}	if 조건문, else if 조건문도 모두 false일 경우 내용 실행	조건문
switch (변수) { case 값 : 내용 break; ... default : 내용}	변수가 case의 값과 일치하면 해당하는 case의 내용을 실행. 만약 변수 값이 case들의 값과 일치하지 않는다면 default의 내용 실행	조건문
for(초기문; 조건문; 증감문) {내용}	조건문이 false로 판별될 때까지 반복	반복문
while (조건문) {내용}	조건문이 false로 판별될 때까지 반복	반복문
do {내용} while (조건문)	내용이 한번은 실행이 된 후 조건문이 false로 판별될 때까지 내용 반복	반복문

getElementById()	id의 값으로 특정한 값을 가진 요소 선택	선택자
getElementsByClassName()	class의 값으로 특정한 값을 가진 요소 선택	선택자
getElementsByName()	name의 값으로 특정한 값을 가진 요소 선택	선택자
getElementsByTagName()	태그 이름의 값으로 특정한 값을 가진 요소 선택	선택자
querySelector()	css 선택자에 해당하는 첫번째 요소 선택	선택자
querySelectorAll()	css 선택자에 해당하는 모든 요소 선택	선택자
toString()	array, boolean, function, number 등 모든 객체를 문자열로 바꾸어 반환	객체 메소드
length	문자열의 길이 반환	문자열 메소드
charAt(index)	지정된 위치에서 문자 반환	문자열 메소드
indexOf(string)	지정된 문자의 위치를 왼쪽부터 찾아서 반환	문자열 메소드
lastIndexOf(string)	지정된 문자의 위치를 오른쪽부터 찾아서 반환	문자열 메소드
substring(index1, index2)	지정된 위치에 있는 문자열 반환	문자열 메소드
slice(start_index, end_index)	문자열의 일부를 추출	문자열 메소드
substr(start_index, length)	문자열을 지정한 위치부터 length개수 만큼의 문자열 추출	문자열 메소드
toLowerCase()	소문자로 변환	문자열 메소드
toUpperCase()	대문자로 변환	문자열 메소드
concat(string)	두 문자열을 합침	문자열 메소드
split([분리자])	문자열을 분리	문자열 메소드
charCodeAt([index])	지정된 index에 해당하는 유니코드 값 반환	문자열 메소드
join()	배열 원소 전부를 하나의 문자열로 합침	배열 메소드
pop()	배열 뒷부분의 값 삭제	배열 메소드
push()	배열 뒷부분에 값 삽입	배열 메소드
shift()	배열 앞부분에 값 삭제	배열 메소드
unshift()	배열 앞부분에 값 삽입	배열 메소드
splice(index, 제거할 요소 개수, 배열에 추가될 요소)	배열의 특정위치에 요소를 추가하거나 삭제	배열 메소드
slice(startIndex, endIndex)	배열의 startIndex부터 endIndex 전 까지의 요소를 새로운 배열 객체로 반환	배열 메소드
concat()	다수의 배열을 합침	배열 메소드
sort()	배열의 원소를 알파벳순으로 정렬	배열 메소드
reverse()	배열의 원소 순서를 거꾸로 바꿈	배열 메소드
var d = new Date()	현재 날짜와 시간을 반환	Date 메소드
getFullYear()	연도를 4비트의 숫자(YYYY)로 반환	Date 메소드
getMonth()	월을 정수로 반환	Date 메소드
getDate()	날짜를 정수로 반환	Date 메소드
getDay()	요일을 정수로 반환	Date 메소드
getHours()	시를 정수로 반환	Date 메소드
getMinutes()	분을 정수로 반환	Date 메소드
getSeconds()	초를 정수로 반환	Date 메소드
setTimeout(함수, 시간)	일정 시간 후 함수 실행	타이머 함수
setInterval(함수, 시간)	일정 시간 간격으로 함수 반복 실행	타이머 함수
clearTimeout(id)	실행되고있는 timeout 중지	타이머 함수
clearInterval(id)	실행되고있는 interval 중지	타이머 함수
load	웹 페이지의 로드가 완료되었을 때	윈도우 이벤트
unload	웹 페이지가 언로드 될 때	윈도우 이벤트
error	브라우저가 자바스크립트 오류를 만났거나 요청한 자원이 없는 경우	윈도우 이벤트
resize	브라우저의 창 크기를 조정했을 때	윈도우 이벤트
scroll	사용자가 페이지를 위아래로 스크롤 할 때	윈도우 이벤트

click	마우스 버튼 클릭했다 땔 때	마우스 이벤트
dbclick	마우스 버튼을 두 번 연속 더블 클릭 할 때	마우스 이벤트
mousedown	마우스 버튼을 누르고 있을 때	마우스 이벤트
mouseup	눌렀던 마우스 버튼을 땔 때	마우스 이벤트
mousemove	마우스를 움직였을 때	마우스 이벤트
mouseout	요소 위로 마우스를 움직였을 때	마우스 이벤트
mouseover	요소 바깥으로 마우스를 움직였을 때	마우스 이벤트
keydown	키를 누르는 순간	키보드 이벤트
keyup	키를 눌렀다 떼는 순간	키보드 이벤트
keypress	키를 눌러 문자가 입력되었을 때	키보드 이벤트
focus	요소에 포커스가 갔을 때	포커스 이벤트
blur	요소가 포커스에서 벗어났을 때	포커스 이벤트
submit	전송 버튼을 눌렀을 때 또는 텍스트 필드에서 엔터키를 눌렀을 때	폼 이벤트
reset	폼을 초기화하기 위함	폼 이벤트
change	폼 필드에서 변경이 일어났을 때 (ex. 라디오 버튼 클릭)	폼 이벤트
select	option 태그에서 option이 선택되었을 때	폼 이벤트

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day2

jQuery

1. 라이브러리
2. jQuery의 필터와 실행방법
3. 이벤트
4. jQuery CheatSheet

8. 라이브러리

8.1 라이브러리

라이브러리란 자주 사용하는 코드를 재사용 할 수 있는 형태로 가공하여 프로그래밍 효율을 높여주는 코드들을 말합니다.

자바스크립트의 가장 유명한 라이브러리는 jQuery입니다.

8.2 jQuery

jQuery는 빠르고, 작고, feature-rich한 자바스크립트 라이브러리입니다.

엘리먼트를 선택하는 강력한 방법과 선택된 엘리먼트를 효율적으로 제어할 수 있는 다양한 수단을 제공합니다.

다음은 jQuery에 대해 참고할 수 있는 사이트들 입니다.

http://jquery.com/

http://www.jqueryrain.com/

물론 요즘 프로젝트에서는 jQuery를 걷어내는 작업을 하는 경우도 있지만, 여전히 강력한 영향력을 끼치고 있는 라이브러리임에는 틀림이 없습니다.

8.3 jQuery 로딩하기

먼저 구글에서 가장 최신의 jQuery cdn을 찾습니다.

- jQuery cdn Code

```HTML
<head>
<script src="https://code.jquery.com/jquery-2.2.4.min.js"></script>
</head>
```

cdn에서 복사해 온 코드를 본래 가지고 있던 html 파일에 붙여 넣습니다.

8.4 jQuery 사용하기

jQuery는 다음과 같이 사용할 수 있습니다.

- jQuery test Code

```JavaScript
$("p").hide();
$("#test").show();
$(".test").click(function(){
  alert("클릭!");
});
```

9. jQuery의 필터와 실행방법

9.1 jQquery 필터

기본 필터

- :eq(index)
- :even
- :odd
- :first
- :last
- :gt(index)
- :lt(index)
- :not(filter)

속성 필터

- :attributeContains - input[name*='man']
- :attributeEquals - input[name='newsletter']

차일드 필터

- :first-child, :last-child
- :nth-child(2)
- :nth-child(even), :nth-child(odd)
- :nth-child(3n)

컨텐츠 필터

- :contains(text)
- :empty
- :has(selector)

9.2 실행방법

- 속성값 변경
- DOM 탐색
- DOM 편집
- CSS
- 이벤트
- 효과
- AJAX
- 유틸리티

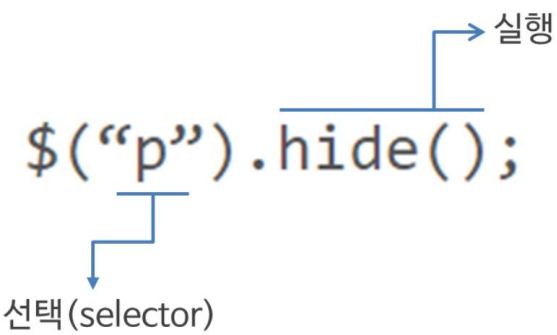
jQuery에서 속성값을 불러오는 방법은 다음과 같습니다.

- 속성값 CODE

```
$("name").val(); #속성값 읽어오기
("name").val();  #속성값 쓰기
```

JavaScript

jQuery는 $(“”)의 부분에서 코드가 적용 될 부분을 작성합니다.
그 다음에 .hide(); 와 같이 실행했을 때 적용되는 코드를 넣습니다.



선택 방법은 기본 선택과 계층 선택이 있습니다.
기본 선택에는 element, .class, #d, all(*)이 있으며,
계층선택에는 child(>)이 있습니다.

이 파트에서는 jQuery에서 DOM을 탐색하는 방법에 대해 알아보도록 하겠습니다.

DOM의 탐색은 일반적으로 선택자를 통해서 이루어집니다.

- 탐색 CODE1

```html
<div class="wrapper">
  <h1 class="title">TITLE</h1>
  <ul class="items">
    <li class="items">item 1</li>
    <li class="items">item 2</li>
  </ul>
</div>
```

HTML

- 탐색 CODE2

```javascript
$("span").parent();
$("div").children();
$("div").first();
$("div").last();
```

JavaScript

빨간 네모 박스가 쳐진 곳이 선택자에 해당하는 부분입니다.

`$(“span”).parent();` 는 부모 선택자입니다. 이것은 부모 인자로 잡히는 모든 상위 인자를 반환합니다. `$(“div”).children();` 는 자식선택자로, 선택된 요소의 자식에 해당하는 것을 전부 반환합니다. `$("div").first();` 는 선택한 요소 중 가장 첫번째를 반환합니다. 이 예시에서는 div 중 가장 처음 나오는 div를 반환합니다. 반대로 `$("div").last();` 는 여러 개의 div 중 가장 마지막에 나오는 div를 반환합니다.

다음으로는 DOM의 편집에 대해 알아보도록 하겠습니다. 먼저 제이쿼리의 문자를 편집하는 메소드에 대해 알아보도록 하겠습니다.

- 편집 CODE1

```javascript
$("test1").text("Hello World!");
$("test2").html("<b>Hello World!</b>");
$("test3").val("<b>Hello World!</b>");
```

JavaScript

위 세 코드는 텍스트를 받아오거나 변경할 수 있게 합니다.

세 코드는 비슷해 보이지만 차이점이 존재합니다.

메소드	특징
.text()	선택한 요소의 텍스트 내용을 받아온다.
.html()	선택한 요소의 html 코드를 설정하거나 받아온다
.val()	form 요소의 값을 설정하거나 받아온다

- 편집 CODE2

```
$("p").append("Some appended text.");
$("p").prepend("Some prepended text.");
$("#div1").remove();
```

JavaScript

`append()` 는 객체를 선택한 객체(p)내의 가장 마지막 요소로 붙입니다. `prepend()` 는 객체를 선택한 객체 내의 가장 첫번째 요소로 붙입니다. `remove()` 는 선택한 객체를 삭제시킵니다.

DOM 편집은 html 요소 뿐만 아니라 css도 변경할 수 있습니다. 아래의 코드를 함께 공부해 보도록 하겠습니다.

- CSS 편집 CODE

```
$("div").addClass("important");
$("h1,h2,p").removeClass("blue");
$("h1,h2,p").toggleClass("blue");
$("p").css("background-color");
$("p").css("background-color","yellow");
```

JavaScript

먼저 addClass() 함수는 클래스를 추가하는 것이 가능합니다. 위 코드에서는 div에 important라는 클래스가 추가되게 됩니다.

예를 들어 위와 같은 코드에 $("div").addClass("world")라는 코드를 실행하면 다음과 같이 클래스가 변화하게 됩니다.

.addClass() 적용 전

```
<div class="hello">hello world!</div>
```

HTML

.addClass() 적용 후

```
<div class="hello world">hello world!</div>
```

HTML

메소드	특징
.removeClass()	선택한 클래스를 삭제한다
.toggleClass()	특정한 클래스의 추가 또는 제거를 한번에 한다
.css("속성", "값")	css의 속성을 선택하고, 값을 변경할 수 있다

이제부터는, jQuery의 effect에 대해 알아보도록 하겠습니다. effct를 사용하면 마치 파워포인트의 애니메이션과 같이 요소에 다양한 효과를 줄 수 있습니다.

- effect CODE

```
$("p").hide();
$("p").show();
$("p"),toggle();
$("#div1").fadeIn();
$("#div1").fadeOut();
$("#div1").fadeToggle();
$("#div3").fadeIn(3000);
```

JavaScript

메소드	설명
.hide()	선택한 요소를 사라지게 한다
.show()	선택한 요소를 나타나게 한다
.toggle()	선택한 요소의 현재 표시상태에 따라 다르게 나타난다. 만약 선택한 요소가 사라진 상태라면 .show()를 실행하고 아닐 경우 반대를 실행한다.
,fadeIn()	선택한 요소의 CSS opacity 값을 높여가며 나타난다
.fadeOut()	선택한 요소의 CSS opacity 값을 낮춰가며 사라진다
.fadeToggle()	선택한 요소에 fadeIn()과 fadeout()을 번갈아가며 적용한다
.fadeIn(3000)	시간을 ms단위 또는 slow, fast등으로 넣어 효과의 시간을 조절할 수 있다.

10. 이벤트

10.1 이벤트

이번에는 이벤트에 대해 알아보도록 하겠습니다. 웹 페이지에서 사용자는 버튼을 클릭하거나, 마우스를 스크롤 하거나, 필드의 내용을 바꾸는 등의 행동(action)을 합니다. 웹 페이지는 이러한 사용자의 행동에 대해 상호작용을 하여 이벤트를 발생시킵니다.

- 클릭 이벤트 CODE - hide

020.html

```html
<p id="text"></p>
<div id="divOne" class="box"></div>
<p class="textTwo"></p>
<h1></h1>
```

HTML

```css
.box {
    width: 100px;
    height: 100px;
    background-color: red;
    display : none;
  }
```

CSS

```javascript
//JQuery - 요소 선택과 hide
  $('#text').text('Hello');
  $('#divOne').fadeIn(3000);
  $('.textTwo').text('안녕하세요');
  $('h1').text('WOW');

  // div 클릭했을 때 안 보이게 하기
  $(document).ready(function(){
    $("div").click(function(){
      $(this).hide();
    });
  });
```

JavaScript

현재 작성중인 html 코드에 다음을 적용해 보도록 하겠습니다. 이 코드에서는 <p>태그에서 .click()이라는 이벤트가 발생하면 <p> 태그에 해당하는 내용을 숨기도록 작성되었습니다. 이렇게 특정 요소의 이벤트를 제어하는 함수를 이벤트 핸들러(event handler)라 합니다.

- 클릭 이벤트 CODE - animate

021.html

```css
#box {
    width: 100px;
    height: 100px;
    opacity: 0.3;
    background-color: red;
  }
```

CSS

```html
<button type="button" id="btn_ani">클릭해주세요!</button>
<div id="box"></div>
```

HTML

```javascript
//JQuery - animate
//버튼 클릭했을 때 변화주기
$('#btn_ani').click(function() {
  $('#box').animate({
    width: '300px',
    height: '300px',
    opacity: 1,
  }, 'slow');
});
```

JavaScript

- 호버 이벤트 CODE

022.html

```css
#box {
    width: 100px;
    height: 100px;
    border: solid 1px #dbdbdb;
  }
```

CSS

```html
<div id="box"></div>
```

HTML

```javascript
//hover - 색 변경하기
$("#box").hover(function(){
  $(this).css("background-color", "yellow");
}, function(){
  $(this).css("background-color", "blue");
});
```

JavaScript

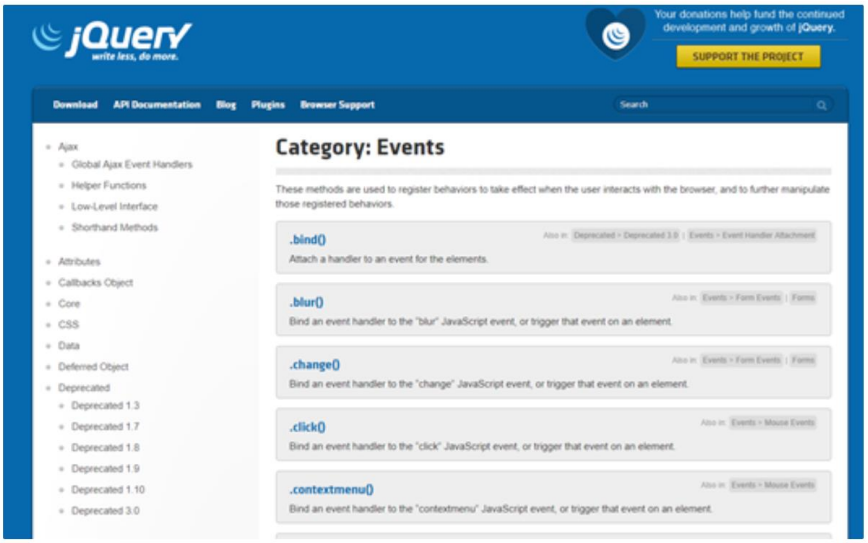
- 마우스 이벤트 CODE

```
$("#p1").mouseenter(function(){
  alert("You entered p1!");
});
$("#p1").mouseleave(function(){
  alert("Bye! You now leaved p1!");
});
```

JavaScript

이번에는 마우스 이벤트를 직접 실습해보면서 위 이벤트 핸들러가 어떤 동작을 발생시키는지 확인해 보도록 합시다. 지금 당장 모든 이벤트 핸들러에 대해서 학습하는 것은 권장하지 않습니다.

다른 이벤트에 대해서 알고 싶다면 다음 사이트(https://api.jquery/com/category/events/)로 들어가 원하는 이벤트에 대해 찾아보면 좋을 것입니다.



혹은 구글에서 원하는 이벤트명과 jQuery를 함께 검색하면 원하는 결과를 찾을 수 있을 것입니다.

10.2 jQuery 플러그인

- jQuery UI
- jQuery validation
- jQuery lightbox
- Bootstrap
- etc

4. jQuery CheatSheet

+ Add a view

Tag Name	설명	구분
$("*")	모든 요소 선택	셀렉터
$("p.demo")	intro 클래스를 가진 <p> 요소 선택	셀렉터
$("p:first")	첫 번째 <p> 요소 선택	셀렉터
$("p span")	p의 하위 태그 span 선택	셀렉터
$("p > span")	p의 바로 한 단계 아래의 하위 태그 span 선택	셀렉터
$("p + span")	p 바로 뒤에 있는 span 선택	셀렉터
$("p ~ span")	p 뒤에 있는 모든 span 선택	셀렉터
$("ul li:first")	첫번째 ul의 첫번째 li 선택	셀렉터
$("ul li:first-child")	모든 ul의 첫번째 li 선택	셀렉터
$("ul li:nth-child(3)")	세번째 li 선택	셀렉터
$("[href]")	href 속성을 가진 모든 요소 선택	셀렉터
$("a[target='_blank']")	target = _blank 속성을 가진 모든 a 속성을	셀렉터
$(":input") ($(":checkbox"))	해당 폼을 선택 (모든 폼 사용 가능)	셀렉터
$(":button")	버튼 태그와 <input type='button'> 인 경우 선택	셀렉터
$("tr:even")	짝수 번째의 tr만 선택	셀렉터
$("tr:odd")	홀수 번쨰의 tr만 선택	셀렉터
$("span:parent")	span을 자식으로 가지는 요소 선택	셀렉터
$("span:contains('demo')")	특정 텍스트를 포함하는 span 선택	셀렉터
$(".demo").click(function(){ $(this).hide(200); });	이벤트는 이와 같은 형태로 사용	이벤트
Mouse 이벤트 종류	scroll, click, dblclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, load, resize, scroll, unload, error	이벤트
Keyboard 이벤트 종류	keydown, keypress, keyup	이벤트
Form 이벤트 종류	submit, change, focus, blur	이벤트
DOM Element 이벤트 종류	blur, focus, focusin, focusout, change, select, submit	이벤트
Browser 이벤트 종류	load, resize, scroll, unload, error	이벤트
$("#demo").hide();	display : none 으로 설정	이펙트
$("#demo").show(200);	애니메이션과 함께 숨겨진 요소를 보여줍니다.	이펙트
$("#demo").toggle();	.hide() 메소드와 .show()메소드를 번갈아가며 적용합니다.	이펙트
$("#demo").fadeIn();	요소를 천천히 사라지게 합니다.	이펙트

$("#demo").fadeOut(300);	요소를 천천히 나타나게 합니다.	이펙트
$("#demo").fadeToggle("slow");	.fadeIn() 메소드와 .faedOut() 메소드를 번갈아가며 적용합니다.	이펙트
$("#demo").fadeTo("slow", 0.25);	최종 opacity 속성값을 직접 설정하여 페이드 효과를 줍니다.	이펙트
$("#demo").slideDown();	서서히 내려가면서 나타나게 합니다	이펙트
$("#demo").slideUp("slow");	서서히 올라가며 나타나게 합니다.	이펙트
$("#demo").slideToggle();	.slideUp() 메소드와 .slideDown() 메소드를 번갈아가며 적용합니다.	이펙트
$("a#mylink").trigger("click");	해당 이벤트와 연결된 핸들러들 실행	일반
$(".demo").each(function() { document.write($(this).text() + "\n"); });	각각의 요소들에게 콜백 함수 실행	일반
$("#demo").text();	텍스트 콘텐츠를 반환	DOM구조 변경
$("#demo").html();	html을 포함한 콘텐츠를 반환	DOM구조 변경
$("#demo").val();	필드 값를 반환	DOM구조 변경
$("#demo").html('Hey yo');	html 콘텐츠를 설정	DOM구조 변경
$("#link").attr("href");	속성 값을 획득	DOM구조 변경
$("#link").attr("href",'https://htmlg.com');	속성 값을 설정	DOM구조 변경
$("#link").attr({ "href" : "https://htmlg.com", // setting multiple attributes "title" : "HTML Editor" });	여러 속성값들을 설정	DOM구조 변경
$(".demo").prepend("Yo!");	선택된 요소의 자식요소 앞에 값을 더함	DOM구조 변경
$(".demo").append("Hey!");	선택된 요소의 자식요소 뒤에 값을 더함	DOM구조 변경
$(".demo").before("Cheers");	선택된 요소의 이전 부분에 값을 더함	DOM구조 변경
$(".demo").after("Peace");	선택된 요소의 아후 부분에 값을 더함	DOM구조 변경
$("#demo").remove();	선택된 요소를 제거합니다.	DOM구조 변경
$("#demo").empty();	자식들을 제거합니다.	DOM구조 변경
$("div").remove(".cl1, .cl2");	나열된클래스가 있는 div를 제거합니다.	DOM구조 변경
$("#demo").addClass("big red");	클래스를 추가합니다.	DOM구조 변경
$("h1, p").removeClass("red");	클래스를 제거합니다.	DOM구조 변경
$("#demo").toggleClass("big");	.addClass()메소드와 .removeClass()메소드를 번갈아가면서 실행합니다.	DOM구조 변경
$("#demo").css("background-color");	css 값을 반환합니다	DOM구조 변경
$("#demo").css("color", "blue");	css 값을 설정합니다.	DOM구조 변경
$("#demo").css({"color": "blue", "font-size": "20px"});	여러 css 값을 설정합니다.	DOM구조 변경

$("#demo").parent();	바로 상위의 부모 요소 선택	트래버싱
$("span").parent().hide();	부모 색상 변경	트래버싱
$("#demo").parents();	모든 상위 요소 선택	트래버싱
$("#demo").parentsUntil("#demo2");	demo2 까지의 상위 요소 선택	트래버싱
$("#demo").children();	바로 아래의 자식 요소 선택	트래버싱
$("#demo").children(".first");	해당 클래스를 가진 자식 요소 선택	트래버싱
$("#demo").find("span");	모든 자식 span 요소 선택	트래버싱
$("#demo").find("*");	모든 자손 요소 선택	트래버싱
$("#demo").siblings("span");	형재 자매인 span 요소 선택	트래버싱
$("#demo").next();	바로 다음에 위치한 형제 요소 선택	트래버싱
$("p").nextAll();	선택한 요소 다음에 위치한 형제 요소를 모두 선택	트래버싱
$("#demo").nextUntil("#demo2");	선택한 요소 다음에서부터 demo2까지의 요소를 모두 선택	트래버싱
$("#demo").prev();	앞의 형제 요소를 선택합니다.	트래버싱
$("p").prevAll();	앞에 있는 모든 요소를 선택합니다.	트래버싱
$("#demo").prevUntil("#demo2");	앞에서부터 demo2까지의 모든 요소를 선택한다.	트래버싱
$("span strong").first();	첫번째 span에 있는 첫번째 strong 요소 선택	트래버싱
$("span strong").last();	마지막 span에 있는 마지막 strong 요소 선택	트래버싱
$("div").eq(9);	n번째 div 요소 선택	트래버싱
$("div").filter(".big");	big 클래스를 가진 모든 div 요소 선택	트래버싱
$("div").not(".big");	big 클래스를 가지지 않은 모든 div 요소 선택	트래버싱

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day2

NotionDB

NotionDB

about.html

```
<!DOCTYPE html>
<html>
<head>
<title>제주카페찾기</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
crossorigin="anonymous">
<link rel="stylesheet" href="./css/main.css">
</head>

<body class="page-about">
<div class="jejucafe-container">
    <!-- global-header 시작 -->
    <header class="global-header">
        <div class="grid">
            <button class="global-menu-btn">
                <i class="fas fa-bars"></i>
                <span class="btn-label">메뉴</span>
            </button>
            <a class="logo" href="./index.html">
                <img src="./images/logo.png" alt="제주카페찾기">
            </a>
            <div class="global-menu">
                <nav class="global-menu-nav">
                    <ul>
                        <li class="menu-item-about">
                            <a href="about.html">서비스 소개</a>
                        </li>
                        <li class="menu-item-list">
                            <a href="list.html">카페들</a>
                        </li>
                    </ul>
                    <a href="write.html" class="add-cafe-btn">
                        <i class="fas fa-plus"></i>
                        <span class="btn-label">카페등록</span>
                    </a>
                </nav>
                <form class="search-cafe-form text-input-and-btn">
                    <div class="input-wrapper">
                        <input type="search" placeholder="찾고 싶은 카페 이름">
                    </div>
```

```
                    <button class="icon-btn search-cafe-form-btn">
                        <i class="fas fa-search"></i>
                        <span class="btn-label">카페찾기</span>
                    </button>
                </form>
            </div>
        </div>
    </header>
    <!-- global-header 끝 -->

    <!-- content-area 시작 -->
    <div class="content-area">
        <!-- 소개 페이지 내용 시작 -->
        <article class="page-content">
            <div class="grid">
                <header>
                    <h1>제주도 카페, 여기 다 있어요!</h1>
                </header>
                <section class="content-section">
                    <header>
                        <h2>섹션 제목</h2>
                    </header>
                    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
                    Dolorem cupiditate molestias vel veritatis odio repellat quod velit,
                    at delectus dolor tenetur, laudantium atque, inventore.
                    Quia obcaecati accusantium consequuntur dignissimos fugit.</p>
                    <p>제주코딩베이스캠프에서 제작하는 예제 웹사이트입니다.</p>
                </section>
            </div>
        </article>
        <!-- 소개 페이지 내용 끝 -->
    </div>
    <!-- content-area 끝 -->

    <!-- global-footer 시작 -->
    <footer class="global-footer">
        <div class="grid">
            <p class="copyright">
                2018 &copy; 제주카페찾기
            </p>
        </div>
    </footer>
    <!-- global-footer 끝 -->
</div>

<script src="./js/index.js"></script>
</body>
</html>
```

HTML

index.html

```
<!DOCTYPE html>
<html>
<head>
<title>제주카페찾기</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
crossorigin="anonymous">
<script src="https://ajax.googleapis.com/ajax/libs/webfont/1.6.16/webfont.js"></script>
<script type="text/javascript">
  WebFont.load({
    // For google fonts
    google: {
      families: ['Noto Sans KR', 'Roboto']
    }
  });
</script>
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/Swiper/4.4.6/css/swiper.min.css">
<script src="https://cdnjs.cloudflare.com/ajax/libs/Swiper/4.4.6/js/swiper.min.js">
</script>

<link rel="stylesheet" href="./css/main.css">
</head>

<body class="page-home">
<div class="jejucafe-container">
    <!-- global-header 시작 -->
    <header class="global-header">
        <div class="grid">
            <button class="global-menu-btn">
                <i class="fas fa-bars"></i>
                <span class="btn-label">메뉴</span>
            </button>
            <a class="logo" href="./index.html">
                <img src="./images/logo.png" alt="제주카페찾기">
            </a>
            <div class="global-menu">
                <nav class="global-menu-nav">
                    </ul>
                    <a href="write.html" class="add-cafe-btn">
                        <i class="fas fa-plus"></i>
                        <span class="btn-label">카페등록</span>
                    </a>
                </nav>
                <form class="search-cafe-form text-input-and-btn">
                    <div class="input-wrapper">
                        <input type="search" placeholder="찾고 싶은 카페 이름">
                    </div>
                    <button class="icon-btn search-cafe-form-btn">
```

```
                    <button class="icon-btn search-cafe-form-btn">
                        <i class="fas fa-search"></i>
                        <span class="btn-label">카페찾기</span>
                    </button>
                </form>
            </div>
        </div>
    </header>
    <!-- global-header 끝 -->

    <!-- content-area 시작 -->
    <div class="content-area">
        <!-- jejumap-container 시작 -->
        <section class="jejumap-container">
            <div class="grid">
                <header class="jejumap-header">
                    <h1 class="main-copy">지도에서 원하는 지역을 선택하세요!</h1>
                    <p>선택하지 않으면 전체 지역에서 찾습니다</p>
                </header>
                <div class="jejumap">
                    <div class="jejumap-items">
                        <div class="jejumap-item" id="jejumap-item-hangyeong">
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-1" data-city="notJcity">
                            <div class="mapshape"></div>
                            <label class="jejumap-item-title te-elem" for="check-area-1">
                            한경면</label>
                        </div>
                        <div class="jejumap-item" id="jejumap-item-hanlim">
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-2" data-city="notJcity">
                            <div class="mapshape"></div>
                            <label class="jejumap-item-title te-elem" for="check-area-2">
                            한림읍</label>
                        </div>
                        <div class="jejumap-item" id="jejumap-item-aewol">
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-3" data-city="notJcity">
                            <div class="mapshape"></div>
                            <label class="jejumap-item-title te-elem" for="check-area-3">
                            애월읍</label>
                        </div>
                        <div class="jejumap-item" id="jejumap-item-jeju">
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-0" data-city="Jcity">
                            <div class="mapshape"></div>
                            <label class="jejumap-item-title te-elem" for="check-area-0">
                            제주시</label>
                        </div>
                        <div class="jejumap-item" id="jejumap-item-jocheon">
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-4" data-city="notJcity">
                            <div class="mapshape"></div>
```

```
                <label class="jejumap-item-title te-elem" for="check-area-4">
                조천읍</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-gujwa">
                <input type="checkbox" class="jejumap-check te-elem"
                id="check-area-5" data-city="notJcity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-5">
                구좌읍</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-daejeong">
                <input type="checkbox" class="jejumap-check te-elem"
                id="check-area-9" data-city="notScity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-9">
                대정읍</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-andeok">
                <input type="checkbox" class="jejumap-check te-elem"
                id="check-area-10" data-city="notScity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-10">
                안덕면</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-seogwipo">
                <input type="checkbox" class="jejumap-check te-elem"
                id="check-area-8" data-city="Scity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-8">
                서귀포시</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-namwon">
                <input type="checkbox" class="jejumap-check te-elem"
            id="check-area-11" data-city="notScity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-11">
                남원읍</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-pyoseon">
                <input type="checkbox" class="jejumap-check te-elem"
                  id="check-area-12" data-city="notScity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-12">
                표선면</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-seongsan">
                <input type="checkbox" class="jejumap-check te-elem"
                id="check-area-13" data-city="notScity">
                <div class="mapshape"></div>
                <label class="jejumap-item-title te-elem" for="check-area-13">
                성산읍</label>
            </div>
            <div class="jejumap-item" id="jejumap-item-udo">
```

```
                            <input type="checkbox" class="jejumap-check te-elem"
                            id="check-area-6" data-city="notJcity">
                            <div class="mapshape"></div>
                            <label class="jejumap-item-title te-elem" for="check-area-6">
                            우도면</label>
                        </div>
                    </div>
                </div>
                <a href="list.html" class="btn big-btn search-map-btn">카페찾기!</a>
            </div>
        </section>
        <!-- jejumap-container 끝 -->
    </div>
    <!-- content-area 끝 -->

    <!-- global-footer 시작 -->
    <footer class="global-footer">
        <div class="grid">
            <p class="copyright">
                2018 &copy; 제주카페찾기
            </p>
        </div>
    </footer>
    <!-- global-footer 끝 -->
</div>

<script src="./js/index.js"></script>
</body>
</html>
```

list.html

Copy to clipboard

```
<!DOCTYPE html>
<html>
<head>
<title>제주카페찾기</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
crossorigin="anonymous">
<link rel="stylesheet" href="./css/main.css">
</head>

<body class="page-list">
<div class="jejucafe-container">
    <!-- global-header 시작 -->
    <header class="global-header">
        <div class="grid">
            <button class="global-menu-btn">
                <i class="fas fa-bars"></i>
                <span class="btn-label">메뉴</span>
            </button>
            <a class="logo" href="./index.html">
                <img src="./images/logo.png" alt="제주카페찾기">
            </a>
            <div class="global-menu">
                <nav class="global-menu-nav">
                    <ul>
                        <li class="menu-item-about">
                            <a href="about.html">서비스 소개</a>
                        </li>
                        <li class="menu-item-list">
                            <a href="list.html">카페들</a>
                        </li>
                    </ul>
                    <a href="write.html" class="add-cafe-btn">
                        <i class="fas fa-plus"></i>
                        <span class="btn-label">카페등록</span>
                    </a>
                </nav>
                <form class="search-cafe-form text-input-and-btn">
                    <div class="input-wrapper">
                        <input type="search" placeholder="찾고 싶은 카페 이름">
                    </div>
                    <button class="icon-btn search-cafe-form-btn">
                        <i class="fas fa-search"></i>
                        <span class="btn-label">카페찾기</span>
                    </button>
                </form>
            </div>
        </div>
```

```
        </div>
    </div>
</header>
<!-- global-header 끝 -->

<!-- content-area 시작 -->
<div class="content-area">
    <!-- cafe-list 시작 -->
    <div class="cafe-list">
        <div class="grid">
            <!-- 카페 1개 시작 -->
            <section class="cafe-item">
                <figure class="cafe-item-photo">
                    <img src="./images/temp/cafe_0.jpg" alt="">
                </figure>
                <div class="cafe-item-info">
                    <h2>제주코딩카페</h2>
                    <p>개발자와 디자이너들을 위한 카페에요.</p>
                    <ul class="cafe-item-contact">
                        <li>
                            <i class="fas fa-map-marker-alt"></i>
                            제주시 조천읍 함덕00길 000
                        </li>
                        <li>
                            <i class="fas fa-phone"></i>
                            064-000-0000
                        </li>
                        <li>
                            <i class="fab fa-instagram"></i>
                            @jejucodingcafe
                        </li>
                    </ul>
                </div>
            </section>
            <!-- 카페 1개 끝 -->

            <!-- 카페 1개 시작 -->
            <section class="cafe-item">
                <figure class="cafe-item-photo">
                    <img src="./images/temp/cafe_1.jpg" alt="">
                </figure>
                <div class="cafe-item-info">
                    <h2>CAFE 마트료시카</h2>
                    <p>러시아 인형 장식이 인상적인 러시아풍 카페에요.</p>
                    <ul class="cafe-item-contact">
                        <li>
                            <i class="fas fa-map-marker-alt"></i>
                            제주시 조천읍 함덕00길 000
                        </li>
                        <li>
                            <i class="fas fa-phone"></i>
                            064-000-0000
                        </li>
                        <li>
```

```
                            <i class="fab fa-instagram"></i>
                            @jejucodingcafe
                        </li>
                    </ul>
                </div>
            </section>
            <!-- 카페 1개 끝 -->

            <!-- 카페 1개 시작 -->
            <section class="cafe-item">
                <figure class="cafe-item-photo">
                    <img src="./images/temp/cafe_2.jpg" alt="">
                </figure>
                <div class="cafe-item-info">
                    <h2>크리스마스 카페</h2>
                    <p>크리스마스에만 문을 여는 카페입니다.</p>
                    <ul class="cafe-item-contact">
                        <li>
                            <i class="fas fa-map-marker-alt"></i>
                            제주시 조천읍 함덕00길 000
                        </li>
                        <li>
                            <i class="fas fa-phone"></i>
                            064-000-0000
                        </li>
                        <li>
                            <i class="fab fa-instagram"></i>
                            @jejucodingcafe
                        </li>
                    </ul>
                </div>
            </section>
            <!-- 카페 1개 끝 -->
        </div>
    </div>
    <!-- cafe-list 끝 -->
</div>
<!-- content-area 끝 -->

<!-- global-footer 시작 -->
<footer class="global-footer">
    <div class="grid">
        <p class="copyright">
            2018 &copy; 제주카페찾기
        </p>
    </div>
</footer>
<!-- global-footer 끝 -->
</div>

<script src="./js/index.js"></script>
</body>
</html>
```

write.html

Copy to clipboard •••

```
<!DOCTYPE html>
<html>
<head>
<title>제주카페찾기</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
crossorigin="anonymous">
<link rel="stylesheet" href="./css/main.css">
</head>

<body class="page-write">
<div class="jejucafe-container">
    <!-- global-header 시작 -->
    <header class="global-header">
        <div class="grid">
            <button class="global-menu-btn">
                <i class="fas fa-bars"></i>
                <span class="btn-label">메뉴</span>
            </button>
            <a class="logo" href="./index.html">
                <img src="./images/logo.png" alt="제주카페찾기">
            </a>
            <div class="global-menu">
                <nav class="global-menu-nav">
                    <ul>
                        <li class="menu-item-about">
                            <a href="about.html">서비스 소개</a>
                        </li>
                        <li class="menu-item-list">
                            <a href="list.html">카페들</a>
                        </li>
                    </ul>
                    <a href="write.html" class="add-cafe-btn">
                        <i class="fas fa-plus"></i>
                        <span class="btn-label">카페등록</span>
                    </a>
                </nav>
                <form class="search-cafe-form text-input-and-btn">
                    <div class="input-wrapper">
                        <input type="search" placeholder="찾고 싶은 카페 이름">
                    </div>
                    <button class="icon-btn search-cafe-form-btn">
                        <i class="fas fa-search"></i>
                        <span class="btn-label">카페찾기</span>
                    </button>
                </form>
            </div>
        </div>
    </header>
```

```
<!-- global-header 끝 -->

<!-- content-area 시작 -->
<div class="content-area">
    <!-- 카페등록 페이지 내용 시작 -->
    <article class="page-content">
        <div class="grid">
            <header>
                <h1>카페등록</h1>
            </header>
            <form>
                <div class="form-row">
                    <div class="form-row-content">
                        <span class="form-row-label">카페위치</span>
                        <div class="form-row-input">
                            <label>
                                <input type="radio" name="cafeArea" value="1">
                                <span>구좌읍</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="2">
                                <span>남원읍</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="3">
                                <span>대정읍</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="4">
                                <span>서귀포시내</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="5">
                                <span>성산읍</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="6">
                                <span>안덕면</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="7">
                                <span>애월읍</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="8">
                                <span>우도면</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="9">
                                <span>제주시내</span>
                            </label>
                            <label>
                                <input type="radio" name="cafeArea" value="10">
```

```
                        <span>조천읍</span>
                    </label>
                    <label>
                        <input type="radio" name="cafeArea" value="11">
                        <span>표선면</span>
                    </label>
                    <label>
                        <input type="radio" name="cafeArea" value="12">
                        <span>한경면</span>
                    </label>
                    <label>
                        <input type="radio" name="cafeArea" value="13">
                        <span>한림읍</span>
                    </label>
                </div>
            </div>
        </div>
        <div class="form-row">
            <label class="form-row-content">
                <span class="form-row-label">카페이름</span>
                <div class="form-row-input">
                    <input type="text" name="cafeName">
                </div>
            </label>
        </div>
        <div class="form-row">
            <label class="form-row-content">
                <span class="form-row-label">주소</span>
                <div class="form-row-input">
                    <input type="text" name="cafeAddress">
                </div>
            </label>
        </div>
        <div class="form-row">
            <label class="form-row-content">
                <span class="form-row-label">전화번호</span>
                <div class="form-row-input">
                    <input type="text" name="cafePhoneNum">
                </div>
            </label>
        </div>
        <div class="form-row">
            <label class="form-row-content">
                <span class="form-row-label">카페소개</span>
```

```
                    </div>
                    <div class="form-row">
                        <label class="form-row-content">
                            <span class="form-row-label">카페소개</span>
                            <div class="form-row-input">
                                <input type="text" name="cafeIntro">
                            </div>
                        </label>
                    </div>
                    <div class="form-row">
                        <label class="form-row-content">
                            <span class="form-row-label">사진</span>
                            <div class="form-row-input">
                                <input type="file" name="cafePhoto">
                            </div>
                        </label>
                    </div>
                    <div class="form-row form-row-btn-con">
                        <input type="submit" value="등록하기"
                        class="btn big-btn cafe-write-btn">
                    </div>
                </form>
            </div>
        </article>
        <!-- 카페등록 페이지 내용 끝 -->
    </div>
    <!-- content-area 끝 -->

    <!-- global-footer 시작 -->
    <footer class="global-footer">
        <div class="grid">
            <p class="copyright">
                2018 &copy; 제주카페찾기
            </p>
        </div>
    </footer>
    <!-- global-footer 끝 -->
</div>

<script src="./js/index.js"></script>
</body>
</html>
```

HTML

write.html

Copy to clipboard •••

```
<!DOCTYPE html>
<html>
<head>
<title>제주카페찾기</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/"
crossorigin="anonymous">
<link rel="stylesheet" href="./css/main.css">
</head>

<body class="page-write">
<div class="jejucafe-container">
    <!-- global-header 시작 -->
    <header class="global-header">
        <div class="grid">
            <button class="global-menu-btn">
                <i class="fas fa-bars"></i>
                <span class="btn-label">메뉴</span>
            </button>
            <a class="logo" href="./index.html">
                <img src="./images/logo.png" alt="제주카페찾기">
            </a>
            <div class="global-menu">
                <nav class="global-menu-nav">
                    <ul>
                        <li class="menu-item-about">
                            <a href="about.html">서비스 소개</a>
                        </li>
                        <li class="menu-item-list">
                            <a href="list.html">카페들</a>
                        </li>
                    </ul>
                    <a href="write.html" class="add-cafe-btn">
                        <i class="fas fa-plus"></i>
                        <span class="btn-label">카페등록</span>
                    </a>
                </nav>
                <form class="search-cafe-form text-input-and-btn">
                    <div class="input-wrapper">
                        <input type="search" placeholder="찾고 싶은 카페 이름">
                    </div>
                    <button class="icon-btn search-cafe-form-btn">
                        <i class="fas fa-search"></i>
                        <span class="btn-label">카페찾기</span>
                    </button>
                </form>
            </div>
        </div>
    </header>
```

js

index.js

```
(function() {

    var menuBtn = document.querySelector('.global-menu-btn');

    menuBtn.addEventListener('click', function() {
        document.body.classList.toggle('menu-on');
    });

})();
```

JavaScript

CSS

main.css

```
html, body, h1, h2, h3, h4, h5, h6, p, blockquote, code, figure, img, dl,
dt, dd, ol, ul, li, fieldset, legend, caption { margin: 0; padding: 0; border: 0; }
div, span, article, section, header, footer, p, ul, li, figure, a, fieldset,
legend, label, input { box-sizing: border-box; }

html {
    height: 100%;
    color: #333;
    font-family: 'Noto Sans KR', Roboto, sans-serif;
}

img {
    max-width: 100%;
    height: auto;
}

a {
    display: inline-block;
    color: #333;
}

li {
    list-style: none;
}

input[type='text'],
input[type='password'],
input[type='search'],
input[type='email'] {
    height: 36px;
    border: 1px solid #ccc;
    border-radius: 3px;
    font-size: 1rem;
    background: white;
}

.grid {
    padding: 0 20px;
}
```

```
.jejucafe-container {
    display: flex;
    flex-direction: column;
    min-height: 100vh;
    padding-top: 45px;
}

.content-area {
    flex: 1;
    padding: 2rem 0;
}

body.page-home .content-area {
    display: flex;
    align-items: center;
    justify-content: center;
}

.global-header {
    position: fixed;
    top: 0;
    left: 0;
    z-index: 100;
    width: 100%;
    height: 45px;
    background: #1cb1c9;
}

.global-header > .grid {
    height: 100%;
}

.global-header .logo {
    position: absolute;
    top: 50%;
    left: 50%;
    width: 140px;
    font-size: 0;
    transform: translate(-50%, -50%);
}

.global-header input[type='search'] {
    border: 0;
}
```

```
.global-menu-btn {
    display: flex;
    align-items: center;
    justify-content: center;
    position: absolute;
    top: 0;
    left: 0;
    z-index: 110;
    width: 45px;
    height: 45px;
    border: 0;
    font-size: 1.2rem;
    color: white;
    background: transparent;
    cursor: pointer;
}

.btn {
    border-radius: 3px;
    font-size: 1rem;
    text-decoration: none;
    background: white;
    cursor: pointer;
}

.big-btn {
    display: block;
    margin: 2rem auto;
    padding: 0.5em;
    border: 0;
    color: white;
    text-align: center;
    background: #1cb1c9;
}

.icon-btn {
    width: 36px;
    height: 36px;
    font-size: 1.2rem;
    border: 0;
    background: transparent;
    cursor: pointer;
}

.btn-label {
    display: inline-block;
}
```

```
.icon-btn .btn-label,
.global-menu-btn .btn-label {
    overflow: hidden;
    position: absolute;
    width: 1px;
    height: 1px;
    clip: rect(0, 1px, 1px, 1px);
}

.global-menu {
    display: none;
    position: fixed;
    top: 0;
    left: 0;
    z-index: 100;
    width: 100%;
    height: 100%;
    padding: 2em;
    color: white;
    background: rgba(0, 0, 0,0.8);
}

.global-menu-nav {
    margin-top: 5vh;
}

.global-menu a {
    padding: 1em 0;
    color: white;
    text-decoration: none;
}

body.page-about .menu-item-about a,
body.page-list .menu-item-list a {
    font-weight: 900;
    text-shadow: rgba(0, 0, 0, 0.3) 0 0 0.3em;
}

.global-menu .icon-btn {
    color: white;
}

body.menu-on {
    overflow: hidden;
    height: 100%;
}
```

```
body.menu-on .global-menu {
    display: block;
}

.text-input-and-btn {
    display: flex;
}

.text-input-and-btn .input-wrapper {
    flex: 1;
    margin-right: 0.5em;
}

.text-input-and-btn input[type='text'],
.text-input-and-btn input[type='search'],
.text-input-and-btn input[type='email'] {
    width: 100%;
}

.search-cafe-form {
    margin-top: 1rem;
}

.jejumap-header {
    margin-bottom: 2rem;
    text-align: center;
}

.main-copy {
    text-align: center;
    font-size: 1.3rem;
    font-weight: 200;
}

.jejumap-header p {
    font-size: 0.8rem;
}

.jejumap-container {
    margin: 2rem 0;
}

.jejumap {
    position: relative;
    width: 320px;
    height: 160px;
    margin: 0 auto;
}
```

```
body.menu-on .global-menu {
    display: block;
}

.text-input-and-btn {
    display: flex;
}

.text-input-and-btn .input-wrapper {
    flex: 1;
    margin-right: 0.5em;
}

.text-input-and-btn input[type='text'],
.text-input-and-btn input[type='search'],
.text-input-and-btn input[type='email'] {
    width: 100%;
}

.search-cafe-form {
    margin-top: 1rem;
}

.jejumap-header {
    margin-bottom: 2rem;
    text-align: center;
}

.main-copy {
    text-align: center;
    font-size: 1.3rem;
    font-weight: 200;
}

.jejumap-header p {
    font-size: 0.8rem;
}

.jejumap-container {
    margin: 2rem 0;
}

.jejumap {
    position: relative;
    width: 320px;
    height: 160px;
    margin: 0 auto;
}
```

```
.jejumap-item {
    position: absolute;
    transition: all 0.2s;
}

.jejumap-item-title {
    position: absolute;
    z-index: 20;
    min-width: 40px;
    min-height: 40px;
    padding-top: 13px;
    font-size: 12px;
    text-align: center;
    cursor: pointer !important;
}

.jejumap-check {
    position: absolute;
    top: 30%;
    left: 30%;
    opacity: 0;
}

.jejumap-check:checked ~ .jejumap-item-title {
    font-weight: bold;
}

.jejumap-item .mapshape {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-image: url('../images/sprite_map.png');
    background-repeat: no-repeat;
    background-size: 428px 334px;
}

.jejumap-item:nth-child(1) {
    top: 44.6875%;
    left: 0.46875%;
    width: 57px;
    height: 59px;
}
```

```
.jejumap-item:nth-child(1) .mapshape {
    background-position: 0.539% 0.364%;
 }
.jejumap-item:nth-child(1) .jejumap-check:checked + .mapshape {
    background-position: 0.54% 61.202%;
}
.jejumap-item:nth-child(1) .jejumap-item-title { top: 23%; left: 0%; }

.jejumap-item:nth-child(2) {
    top: 24.6875%; left: 8.3%;
    width: 64px; height: 64px;
}
.jejumap-item:nth-child(2) .mapshape { background-position: 17.308% 0.37%; }
.jejumap-item:nth-child(2) .jejumap-check:checked + .mapshape {
    background-position: 17.194% 62.222%;
}
.jejumap-item:nth-child(2) .jejumap-item-title { top: 20%; left: 17%; }

.jejumap-item:nth-child(3) {
    top: 16.875%; left: 20.3%;
    width: 85px; height: 65px;
}
.jejumap-item:nth-child(3) .mapshape { background-position: 38.047% 0.372%; }
.jejumap-item:nth-child(3) .jejumap-check:checked + .mapshape {
     background-position: 38.047% 62.57%;
}
.jejumap-item:nth-child(3) .jejumap-item-title {
    top: 15%; left: 13%; min-height: 48px; padding-top: 16px;
}

.jejumap-item:nth-child(4) {
    top: 5.625%; left: 31%;
    width: 81px; height: 69px;
}
.jejumap-item:nth-child(4) .mapshape { background-position: 63.165% 0.377%; }
.jejumap-item:nth-child(4) .jejumap-check:checked + .mapshape {
    background-position: 63.112% 63.396%;
}
.jejumap-item:nth-child(4) .jejumap-item-title {
    top: 5%; left: 26%; min-width: 58px; min-height: 58px; padding-top: 27px;
}

.jejumap-item:nth-child(5) {
    left: 54.8%; top: 1.875%;
    width: 52px; height: 66px;
}
.jejumap-item:nth-child(5) .mapshape { background-position: 80.718% 0.373%; }
.jejumap-item:nth-child(5) .jejumap-check:checked + .mapshape {
    background-position: 80.5% 62.804%;
}
.jejumap-item:nth-child(5) .jejumap-item-title {
     top: 5%; left: 15%; min-height: 46px; padding-top: 18px;
}
```

```
.jejumap-item:nth-child(7) {
    left: 5%; top: 67%;
    width: 65px; height: 51px;
}
.jejumap-item:nth-child(7) .mapshape { background-position: 0.551% 25.7%; }
.jejumap-item:nth-child(7) .jejumap-check:checked + .mapshape {
    background-position: 0.552% 84.956%; }
.jejumap-item:nth-child(7) .jejumap-item-title { top: 13%; left: 28%; }

.jejumap-item:nth-child(8) {
    left: 19.7%; top: 56%;
    width: 58px; height: 56px;
}
.jejumap-item:nth-child(8) .mapshape { background-position: 19.189% 26.439%; }
.jejumap-item:nth-child(8) .jejumap-check:checked + .mapshape {
    background-position: 19.215% 86.643%;
}
.jejumap-item:nth-child(8) .jejumap-item-title { top: 9%; left: 12%; }

.jejumap-item:nth-child(9) {
    left: 33.8%; top: 50%;
    width: 67px; height: 70px;
}
.jejumap-item:nth-child(9) .mapshape { background-position: 36.842% 27.788%; }
.jejumap-item:nth-child(9) .jejumap-check:checked + .mapshape {
    background-position: 36.806% 90.909%;
}
.jejumap-item:nth-child(9) .jejumap-item-title {
    top: 8%; left: 6%; min-width: 58px; min-height: 58px; padding-top: 22px;
}

.jejumap-item:nth-child(10) {
    left: 49.7%; top: 40.5%;
    width: 70px; height: 81px;
}
.jejumap-item:nth-child(10) .mapshape { background-position: 56.983% 29.051%; }
.jejumap-item:nth-child(10) .mapshape { background-position: 56.983% 29.051%; }
.jejumap-item:nth-child(10) .jejumap-check:checked + .mapshape {
    background-position: 56.923% 95.05%;
}
.jejumap-item:nth-child(10) .jejumap-item-title {
    top: 10%; left: 20%; min-width: 45px; min-height: 60px; padding-top: 25px;
}

.jejumap-item:nth-child(11) {
    left: 60%; top: 27%;
    width: 72px; height: 84px;
}
.jejumap-item:nth-child(11) .mapshape { background-position: 78.09% 29.341%; }
.jejumap-item:nth-child(11) .jejumap-check:checked + .mapshape {
    background-position: 78.059% 96%;
}
.jejumap-item:nth-child(11) .jejumap-item-title { top: 11%; left: 32%; min-height: 55px; }
```

```
.jejumap-item:nth-child(11) .jejumap-item-title { top: 11%; left: 32%; min-height: 55px; }

.jejumap-item:nth-child(12) {
    left: 78.125%; top: 14.5%;
    width: 56px; height: 91px;
}
.jejumap-item:nth-child(12) .mapshape { background-position: 94.899% 30.247%; }
.jejumap-item:nth-child(12) .jejumap-check:checked + .mapshape {
    background-position: 94.8% 99.2%;
}
.jejumap-item:nth-child(12) .jejumap-item-title { top: 23%; left: 16%; min-height: 50px; }

.jejumap-item:nth-child(13) {
    left: 94.6%; top: 13.75%;
    width: 13px; height: 19px;
}
.jejumap-item:nth-child(13) .mapshape { background-position: 99.518% 23.333%; }
.jejumap-item:nth-child(13) .jejumap-item-title {
    top: -70%; left: auto; right: 0; width: 33px; min-height: 33px; padding-top: 0;
    text-align: right;
}
.jejumap-item:nth-child(13) .jejumap-check:checked + .mapshape {
    background-position: 99.518% 76.2%;
}
.jejumap-item:nth-child(13) .jejumap-check {
    top: 0; left: -2px;
}


.global-footer {
    margin-top: 3em;
    padding: 1em 0 2em;
    border-top: 1px solid #ddd;
    color: #999;
    text-align: center;
    font-size: 0.8rem;
}
.cafe-item {
    margin-bottom: 3rem;
}

.cafe-item-info h2 {
    margin: 0.5em 0;
    font-size: 1.5rem;
    font-weight: 200;
}

.cafe-item-contact {
    margin-top: 1em;
    font-size: 0.8rem;
}
```

```
.cafe-item-contact li {
    margin: 0.5em 0;
}

.cafe-item-contact i {
    display: inline-flex;
    align-items: center;
    justify-content: center;
    width: 16px;
    height: 16px;
}

.content-section {
    margin: 2em 0;
}

.page-content h1 {
    margin-bottom: 1em;
    font-size: 2.5rem;
    font-weight: 200;
}

.page-content h2 {
    margin-bottom: 1em;
    font-size: 1.5rem;
}

.page-content p {
    line-height: 1.6;
}

.form-row {
    margin-bottom: 1.5em;
}

.form-row-btn-con {
    text-align: center;
}

.form-row-content {
    display: flex;
    align-items: center;
}

.form-row-label {
    width: 30%;
}

.form-row-input {
    flex: 1;
}
```

```
.jejumap-item:nth-child(11) .jejumap-item-title { top: 11%; left: 32%; min-height: 55px; }

.jejumap-item:nth-child(12) {
    left: 78.125%; top: 14.5%;
    width: 56px; height: 91px;
}
.jejumap-item:nth-child(12) .mapshape { background-position: 94.899% 30.247%; }
.jejumap-item:nth-child(12) .jejumap-check:checked + .mapshape {
    background-position: 94.8% 99.2%;
}
.jejumap-item:nth-child(12) .jejumap-item-title { top: 23%; left: 16%; min-height: 50px; }

.jejumap-item:nth-child(13) {
    left: 94.6%; top: 13.75%;
    width: 13px; height: 19px;
}
.jejumap-item:nth-child(13) .mapshape { background-position: 99.518% 23.333%; }
.jejumap-item:nth-child(13) .jejumap-item-title {
    top: -70%; left: auto; right: 0; width: 33px; min-height: 33px; padding-top: 0;
    text-align: right;
}
.jejumap-item:nth-child(13) .jejumap-check:checked + .mapshape {
    background-position: 99.518% 76.2%;
}
.jejumap-item:nth-child(13) .jejumap-check {
    top: 0; left: -2px;
}
        display: flex;
        align-items: center;
        justify-content: space-between;
    }

    .global-header .logo {
        position: static;
        transform: none;
    }

    .global-menu-btn {
        display: none;
    }

    .global-menu,
    body.menu-on .global-menu {
        display: flex;
        align-items: center;
        position: static;
        width: auto;
        padding: 0;
        background: transparent;
    }
```

```
.global-menu-nav {
    display: flex;
    margin-top: 0;
}

.global-menu-nav ul {
    display: flex;
}

.search-cafe-form {
    margin-top: 0;
    margin-left: 3rem;
}

.cafe-item {
    display: flex;
    align-items: center;
}

.cafe-item-photo {
    flex: 6;
    margin-right: 2rem;
}

.cafe-item-info {
    flex: 4;
}

.jejumap-header {
    font-size: 1.5rem;
}

.jejumap-container {
    margin-top: 3rem;
    margin-bottom: 3rem;
}
```

```
    .big-btn {
        display: inline-flex;
        padding: 0.5em 3em;
        font-size: 1.2rem;
    }

    .jejumap-container {
        text-align: center;
    }

    .main-copy {
        font-size: 2.5rem;
        letter-spacing: -0.05em;
        word-spacing: 0.05em;
    }

    .cafe-item-info h2 {
        margin-top: -1em;
        font-size: 2.5rem;
        letter-spacing: -0.05em;
        word-spacing: 0.05em;
    }

    .cafe-item-photo {
        margin-right: 3rem;
    }
}
```

CSS

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day3

Bootstrap

1. 부트스트랩
2. 부트스트랩 적용
3. 그리드
4. 이미지
5. 테이블
6. 버튼
7. CARD
8. 캐러셀
9. 마무리하며

0. 부트스트랩

부트스트랩은 웹 프레임 웍 입니다. 프레임 웍은 최소한의 작업으로 빠르게 결과물을 만들 수 있도록 구축해 놓은 종합 공구 세트라고 생각하시면 쉽습니다.

부트스트랩은 큰 모니터 화면으로 보던지 휴대폰처럼 작은 화면으로 보던지 상관 없이 사용자의 시점에 맞춰 최적화 되어 화면을 구성합니다. 이렇게 사용자의 시점에 맞춰 구축되는 사이트를 반영형 사이트라고 하는데 부트스트랩은 반응형 사이트를 구축하기에 최적화 되어 있습니다.

부트스트랩은 form, button, table, navigator가 HTML, CSS로 미리 디자인 되어 있습니다. 부트스트랩 3에서 사용하던 대부분의 태그들이 그대로 살아있지만 많은 부분이 바뀌었기 때문에 새로 공부를 하셔야 합니다.

부트스트랩 4가 나온지 벌써 2년이 넘었고, 이제 부트스트랩 공식 버전(21년)은 5로 넘어갔습니다. 그럼에도 시중에는 부트스트랩 3 책이 있고, 구글에서 한국어로 부트스트랩을 검색하면 아직도 3.0 Ver 번역 페이지가 나오기에 Version에 유의하셔서 개발을 하셔야 합니다.

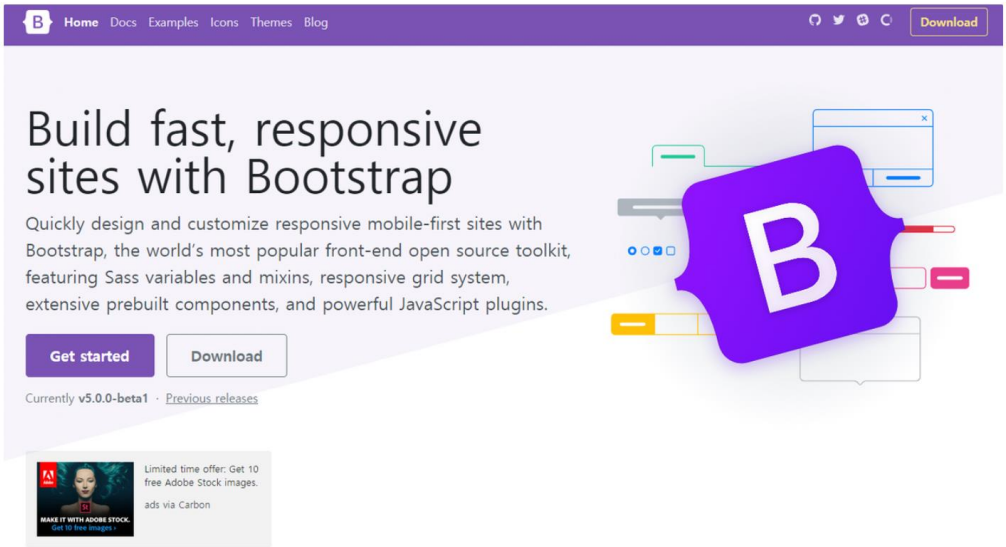
bootstrap4와 bootstrap5의 가장 큰 차이는 JQeury가 없어졌다는 것입니다.

1. 부트스트랩 적용

https://getbootstrap.com/에서 다운로드 받을 수 있습니다.

• 실습은 다운로드를 받지 않고 진행하니 다운로드를 안받으셔도 됩니다

- 영문 홈페이지 : https://getbootstrap.com/(Ver 5.0.0)

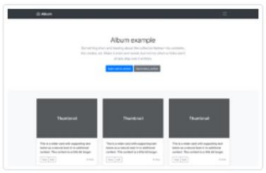


부트스트랩을 이용한다면 다음과 같은 여러 레이아웃을 만들 수 있습니다. 부트스트랩 공식 홈페이지에서 **Examples** 탭에서 다양한 레이아웃을 확인해보세요!

유료 템플릿 사이트는 https://wrapbootstrap.com/ 를 참고해보세요.

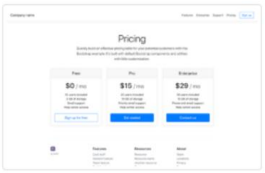
Custom components

Brand new components and templates to help folks quickly get started with Bootstrap and demonstrate best practices for adding onto the framework.



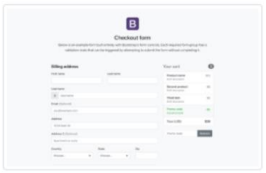
Album

Simple one-page template for photo galleries, portfolios, and more.



Pricing

Example pricing page built with Cards and featuring a custom header and footer.



Checkout

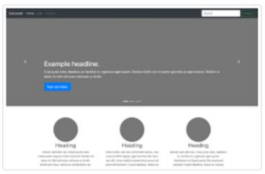
Custom checkout form showing our form components and their validation features.

Product

Lean product-focused marketing page with extensive grid and image work.

Cover

A one-page template for building simple and beautiful home pages.



Carousel

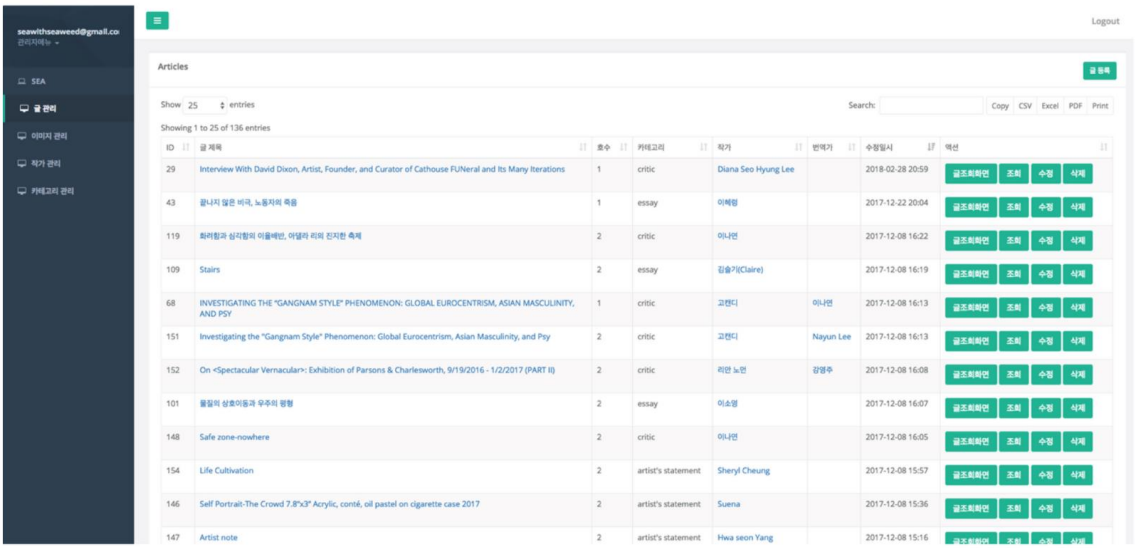
Customize the navbar and carousel, then add some new components.

Blog

Magazine like blog template with header, navigation, featured content.

Dashboard

Basic admin dashboard shell with fixed sidebar and navbar.



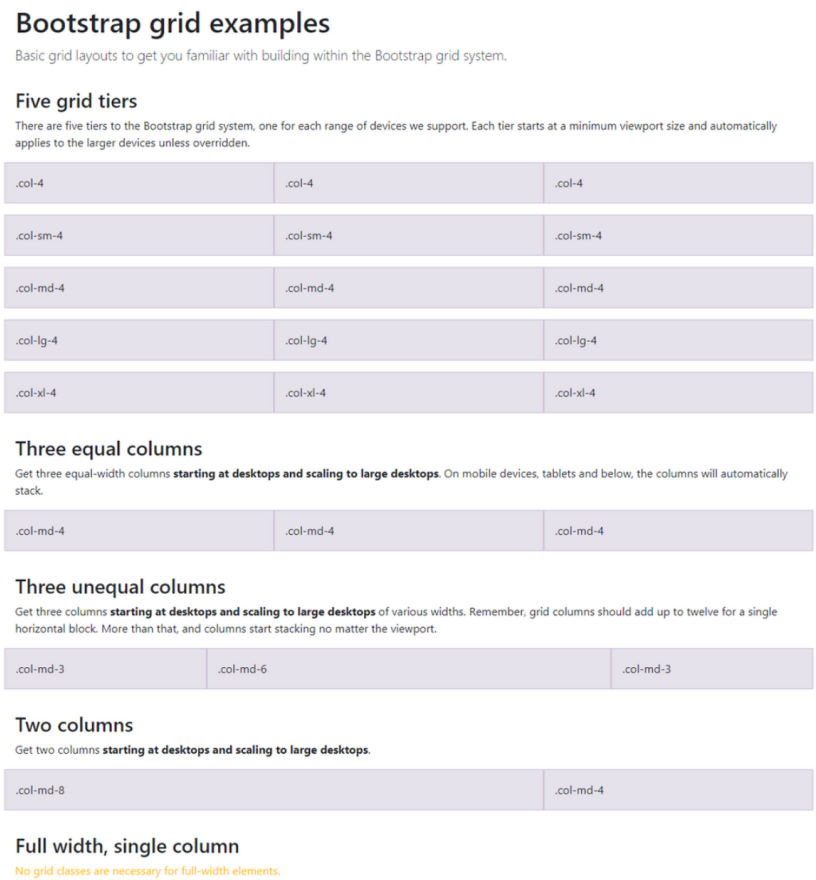
자, 이제 부트스트랩을 적용시켜 보도록 하겠습니다. 다음과 같이 작성하신 다음 `001.html`로 저장해 주세요. 핵심적인 코드만 넣어두었습니다. 그리고 다음 버전에서는 jQuery가 없어지니 이 점도 유념해주세요.

아래 코드 작성 시점은 21년 1월 27일입니다. 이 버전이 너무 오래되었다면 notion에서 commet 부탁드립니다.

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet">

    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.bundle.min.js"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.min.js"></script>
    -->
  </body>
</html>
```

2. 그리드

처음에는 그리드 시스템에 대해 알아볼 것입니다. 아래 공식 홈페이지 글에도 나와 있는 것처럼 부트스트랩은 전체 화면을 12개의 컬럼으로 나눠놓았습니다. 그 나눠놓은 컬럼에 무엇을 배치할 것인지 우리가 정해서 홈페이지를 만들 수 있습니다. 컬럼 12개가 모여 하나의 row가 됩니다.



Bootstrap grid examples

Basic grid layouts to get you familiar with building within the Bootstrap grid system.

Five grid tiers

There are five tiers to the Bootstrap grid system, one for each range of devices we support. Each tier starts at a minimum viewport size and automatically applies to the larger devices unless overridden.

.col-4	.col-4	.col-4
.col-sm-4	.col-sm-4	.col-sm-4
.col-md-4	.col-md-4	.col-md-4
.col-lg-4	.col-lg-4	.col-lg-4
.col-xl-4	.col-xl-4	.col-xl-4

Three equal columns

Get three equal-width columns **starting at desktops and scaling to large desktops**. On mobile devices, tablets and below, the columns will automatically stack.

.col-md-4	.col-md-4	.col-md-4

Three unequal columns

Get three columns **starting at desktops and scaling to large desktops** of various widths. Remember, grid columns should add up to twelve for a single horizontal block. More than that, and columns start stacking no matter the viewport.

.col-md-3	.col-md-6	.col-md-3

Two columns

Get two columns **starting at desktops and scaling to large desktops**.

.col-md-8	.col-md-4

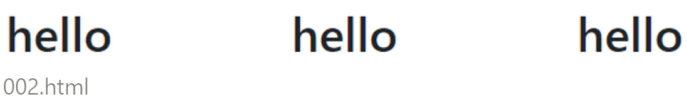
Full width, single column

No grid classes are necessary for full-width elements.

앞서 말씀드린 것처럼 실습 캡쳐는 <body>와 </body> 안에 있는 태그들 중 <script>...</script>를 생략하고 작성하도록 하겠습니다. Code와 같이 작성한다음 `002.html`로 저장을 하고 실행하면 아래와 같이 나옵니다.

빨간색 네모는 이해를 돕기 위해 작성한 것입니다. 1개의 row에 4 컬럼씩 할당했으므로 균등한 넓이로 출력됩니다.

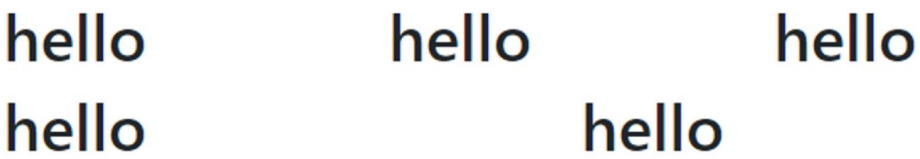
```
<div class="container">
  <div class="row">
    <div class="col-md-4">
      <h1>hello</h1>
    </div>

    <div class="col-md-4">
      <h1>hello</h1>
    </div>

    <div class="col-md-4">
      <h1>hello</h1>
    </div>
  </div>
</div>
```



002.html

이번에는 한 개의 row를 더 주어 봤습니다. 빨간색 네모는 이해를 돕기 위한 것입니다. 위에는 4컬럼씩 주었기 때문에 3등분이 되었고 아래는 6컬럼씩 주었기 때문에 2등분이 되었습니다.

```
<div class="container">
  <div class="row">
    <div class="col-md-4">
      <h1>hello</h1>
    </div>
    <div class="col-md-4">
      <h1>hello</h1>
    </div>
    <div class="col-md-4">
      <h1>hello</h1>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
      <h1>hello</h1>
    </div>
    <div class="col-md-6">
      <h1>hello</h1>
    </div>
  </div>
</div>
```



003.html

아래 공식 홈페이지에서 보듯이 하나의 컬럼 단위를 다시 분할하는 것도 가능합니다. 실습은 해보지 않도록 하겠습니다.

Two columns with two nested columns

Per the documentation, nesting is easy—just put a row of columns within an existing column. This gives you two columns **starting at desktops and scaling to large desktops**, with another two (equal widths) within the larger column.

At mobile device sizes, tablets and down, these columns and their nested columns will stack.

.col-md-8	.col-md-4
.col-md-6 .col-md-6	

Mixed: mobile and desktop

The Bootstrap v4 grid system has five tiers of classes: xs (extra small), sm (small), md (medium), lg (large), and xl (extra large). You can use nearly any combination of these classes to create more dynamic and flexible layouts.

Each tier of classes scales up, meaning if you plan on setting the same widths for xs and sm, you only need to specify xs.

.col-12 .col-md-8	.col-6 .col-md-4
.col-6 .col-md-4 .col-6 .col-md-4 .col-6 .col-md-4	
.col-6 .col-6	

Mixed: mobile, tablet, and desktop

.col-12 .col-sm-6 .col-lg-8	.col-6 .col-lg-4
.col-6 .col-sm-4 .col-6 .col-sm-4 .col-6 .col-sm-4	

그리드 시스템에서 12컬럼을 모두 이용할 때 넓이를 100% 사용하고 싶다면 .container-fluid를 사용합니다.

컬럼 사이에 여백은 .row에 .no-gutters를 적용하면 padding과 margin을 제거할 수 있습니다.

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

공식 홈페이지에 나와있는 그리드 시스템 기준입니다. Nestable은 컬럼을 또다른 컬럼으로 나눌 수 있다는 얘기입니다.

우리는 주로 md를 사용합니다.

3. 이미지

이미지에 반응형과 스타일을 입히는 방법입니다. 아래 실행 이미지는 가로가 2000px인 이미지입니다. 3버전에서는 img-responsive를 사용했었습니다.

```
<div class="container">
  <div class="row">
    <div class="col-2">
      <img src="img/temp2000.png" class="img-fluid" alt="img">
    </div>
    <div class="col-2">
      hello
    </div>
    <div class="col-2">
      hello
    </div>
  </div>
</div>
```

HTML

004.html

이미지 썸네일입니다. 작은이미지 보다는 큰 이미지에 활용됩니다.

```html
<div class="container">
    <div class="row">
      <div class="col-2">
        <img src="img/temp.png" class="img-thumbnail" alt="img">
      </div>
    <div class="col-2">
      hello
    </div>
    <div class="col-2">
      hello
    </div>
  </div>
</div>
```

HTML

hello hello

005.html

화면에는 보이지 않지만 이미지가 컬럼의 가장 오른쪽에 붙어있습니다. 또한 테두리가 미묘하게 깎여 있는 것을 볼 수 있습니다. 추가로 이미지를 둥그렇게 만드는 class는 rounded-circle입니다. 3 이전의 버전에서는 img-circle을 사용했었습니다.

```
<div class="container">
      <div class="row">
        <div class="col-6">
          <img src="img/temp.png" class="rounded float-right" alt="img">
        </div>
        <div class="col-2">
          hello
        </div>
        <div class="col-2">
          hello
        </div>
      </div>
    </div>
```

Python

hello hello

006.html

이제부터는 Code와 Output 위주로 보도록 하겠습니다.

```HTML
<img src="...", class="img_field" alt="Responsive image">
```


```HTML
<img src="..." alt="..." class="img-thubnail">
```


```Python
<img src="cinqueterre.jpg" class="rounded-circle alt="Cinque Terre">
```

Circle:

4. 테이블

다음으로는 부트스트랩에서 기본적으로 제공하고 있는 Table에 대하여 알아보도록 하겠습니다. Thead에는 thead-dark나 thead-light를 주어 thead를 부각시킬 수 있습니다. table-striped를 주면 테이블이 중간중간 강조된 격자무늬로 볼 수 있습니다.

```HTML
<table class="table">
      <thead>
        <tr>
          <th scope="col">구분</th>
          <th scope="col">책이름</th>
          <th scope="col">판매량</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">1</th>
          <td>해리포터</td>
          <td>100권</td>
        </tr>
        <tr>
          <th scope="row">2</th>
          <td>해리포터2</td>
          <td>200권</td>
        </tr>
      </tbody>
    </table>
```

HTML

구분	책이름	판매량
1	해리포터	100권
2	해리포터2	200권

기본 테이블과 마찬가지로 table-striped를 주면 테이블이 중간중간 강조된 격자무늬로 볼 수 있습니다.

```html
<table class="table table-dark">
      <thead>
        <tr>
          <th scope="col">구분</th>
          <th scope="col">책이름</th>
          <th scope="col">판매량</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">1</th>
          <td>해리포터</td>
          <td>100권</td>
        </tr>
        <tr>
          <th scope="row">2</th>
          <td>해리포터2</td>
          <td>200권</td>
        </tr>
      </tbody>
    </table>
```

HTML

구분	책이름	판매량
1	해리포터	100권
2	해리포터2	200권

기본 테이블과 마찬가지로 table-striped를 주면 테이블이 중간중간 강조된 격자무늬로 볼 수 있습니다. table-dark에도 사용할 수 있습니다. table-borderless를 사용하면 테두리를 없앨 수 있습니다.

```html
<table class="table table-bordered">
      <thead>
        <tr>
          <th scope="col">구분</th>
          <th scope="col">책이름</th>
          <th scope="col">판매량</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">1</th>
          <td>해리포터</td>
          <td>100권</td>
        </tr>
        <tr>
          <th scope="row">2</th>
          <td>해리포터2</td>
          <td>200권</td>
        </tr>
      </tbody>
    </table>
```

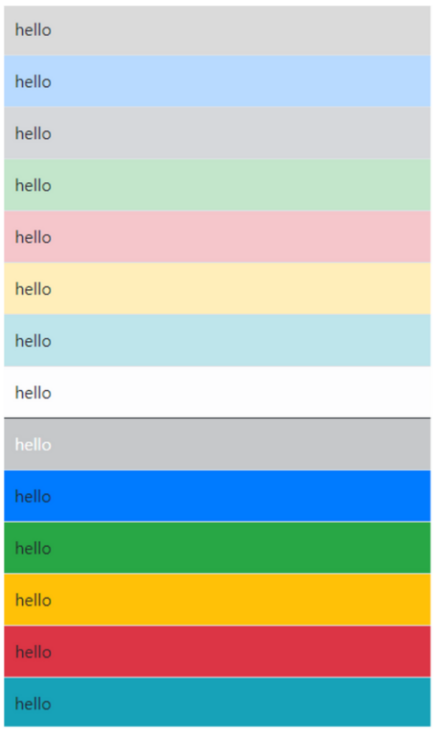
HTML

구분	책이름	판매량
1	해리포터	100권
2	해리포터2	200권

Table을 꾸밀 수 있는 전체 색입니다. 필요에 따라 활용하시면 됩니다.

```
<tbody>
        <tr class="table-active"><td>hello</td></tr>
        <tr class="table-primary"><td>hello</td></tr>
        <tr class="table-secondary"><td>hello</td></tr>
        <tr class="table-success"><td>hello</td></tr>
        <tr class="table-danger"><td>hello</td></tr>
        <tr class="table-warning"><td>hello</td></tr>
        <tr class="table-info"><td>hello</td></tr>
        <tr class="table-light"><td>hello</td></tr>
        <tr class="table-dark"><td>hello</td></tr>
        <tr class="bg-primary"><td>hello</td></tr>
        <tr class="bg-success"><td>hello</td></tr>
        <tr class="bg-warning"><td>hello</td></tr>
        <tr class="bg-danger"><td>hello</td></tr>
        <tr class="bg-info"><td>hello</td></tr>
      </tbody>
    </table>
```

HTML

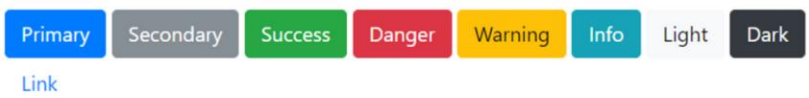


5. 버튼

부트스트랩에 버튼은 다양한 크기와 색을 지원합니다. 아래는 부트스트랩에서 기본적으로 제공하고 있는 버튼입니다.

```html
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>
<button type="button" class="btn btn-link">Link</button>
```

HTML



버튼은 <button> 요소와 함께 사용하도록 설계되었지만 다른 요소에도 사용할 수 있습니다. 아래 다양한 예제를 참고하여 홈페이지를 작성하시면 좋을 것 같습니다. 모두 동일한 모양의 버튼을 만들어 줍니다.

```html
<a class="btn btn-primary" href="#" role="button">Link</a>
<button class="btn btn-primary" type="submit">Button</button>
<input class="btn btn-primary" type="button" value="Input">
<input class="btn btn-primary" type="submit" value="Submit">
<input class="btn btn-primary" type="reset" value="Reset">
```

HTML

Link Button Input Submit Reset

아웃라인만 들어간 버튼을 만들 수도 있습니다. 기본 클래스를 .btn-outline-*로 바꾸면 버튼의 배경 이미지와 색상이 제거된 아웃라인 버튼을 만들 수 있습니다.

```html
<button type="button" class="btn btn-outline-primary">Primary</button>
<button type="button" class="btn btn-outline-secondary">Secondary</button>
<button type="button" class="btn btn-outline-success">Success</button>
<button type="button" class="btn btn-outline-danger">Danger</button>
<button type="button" class="btn btn-outline-warning">Warning</button>
<button type="button" class="btn btn-outline-info">Info</button>
<button type="button" class="btn btn-outline-light">Light</button>
<button type="button" class="btn btn-outline-dark">Dark</button>
```

HTML

Primary Secondary Success Danger Warning Info Light Dark

부트스트랩은 버튼의 사이즈를 빠르게 변경하기 위한 클래스를 제공합니다. .btn-lg (큰 버튼)또는 .btn-sm(작은 버튼)을 추가하면 아래처럼 버튼의 크기가 조정됩니다.

```html
<button type="button" class="btn btn-primary btn-lg">큰 버튼</button>
<button type="button" class="btn btn-secondary btn-lg">큰 버튼</button>

<button type="button" class="btn btn-primary btn-sm">작은 버튼</button>
<button type="button" class="btn btn-secondary btn-sm">작은 버튼</button>
```

HTML

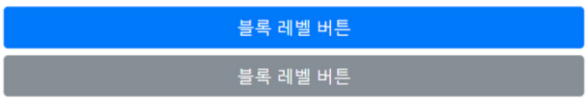
큰 버튼 큰 버튼

작은 버튼 작은 버튼

.btn-block을 추가하면 부모의 전체 넓이만한 버튼을 만들 수도 있습니다.

```
<button type="button" class="btn btn-primary btn-lg btn-block">블록 레벨 버튼</button>
<button type="button" class="btn btn-secondary btn-lg btn-block">블록 레벨 버튼</button>
```

HTML



부트스트랩 버튼에는 활성과 비활성을 제공하고 있습니다. 버튼이 Acrive(활성)상태가 되면 버튼이 눌러져 보입니다.

```
<a href="#" class="btn btn-primary btn-lg active" role="button" aria-pressed="true">Prima
ry link</a>
<a href="#" class="btn btn-secondary btn-lg active" role="button" aria-pressed="true">Lin
k</a>
```

HTML

버튼의 비활성 상태는 기존의 색 그대로를 나타냅니다. disabled 속성을 <button> 요소에 추가하여 버튼을 비활성 상태로 만들 수 있습니다. 활성화된 상태는 .active입니다.

```
<button type="button" class="btn btn-lg btn-primary" disabled>Primary button</button>
<button type="button" class="btn btn-secondary btn-lg" disabled>Button</button>
```

HTML

<a> 요소를 사용하는 비활성 버튼은 조금 다르게 동작합니다.

1. <a>는 disabled 속성을 지원하지 않으므로 비활성화 된 것으로 보이게 하려면 .disabled 클래스를 추가 해야 합니다.
2. anchor 버튼의 모든 pointer-events를 비활성화 합니다. 해당 속성을 지원하는 브라우저에서는 비활성화 된 커서가 전혀 표시되지 않습니다.
3. Disabled buttons(비활성상태 버튼)은 aria-disabled="true" 속성을 포함해야 합니다.
4. 비활성화 된 버튼에는 요소의 상태를 표시하기 위해 aria-disabled="true" 속성을 포함합니다.

```
<a href="#" class="btn btn-primary btn-lg disabled" role="button" aria-disabled="true">Pr
imary link</a>
<a href="#" class="btn btn-secondary btn-lg disabled" role="button" aria-disabled="true">
Link</a>
```

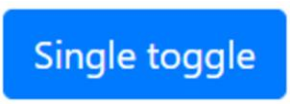
HTML

버튼 플러그인을 사용하면 버튼으로 더 많은 작업을 할 수 있게 됩니다. 토글 상태를 알기 위해서는 `data-toggle="button"` 을 추가하여 버튼 active 상태를 토글합니다.

버튼을 미리 토글할 경우에는 .active 클래스와 aria-pressed="true"를 수동으로 <button>에 추가해야 합니다.

```
<button type="button" class="btn btn-primary" data-toggle="button" aria-pressed="false" a
utocomplete="off">
    Single toggle
</button>
```

HTML



체크박스와 라디오 버튼에 부트스트랩에서 제공하고 있는 플러그인을 사용해보도록 하겠습니다. 부트스트랩의 .button 스타일은 <label>과 같은 요소에 적용되어 checkbo나 radio에 버튼토글을 지정할 수 있습니다. 각각의 스타일에서 토글링할 수 있도록 변경된 버튼이 포함된 `.btn-group` 에 `data-toggle="buttons"` 를 추가할 수 있습니다.

버튼들의 체크상태는 버튼의 클릭 이벤트를 통해서만 업데이트됩니다. 다른 방법을 사용하여<input type="reset"> 또는 input의 checked 속성을 적용하여 입력을 업데이트하는 경우 <label>을 수동으로 토글해야 합니다.

기본 선택된 버튼은 적용하려면 .active 클래스를 input의 <label>에 수동으로 추가해야 합니다.

```html
<div class="btn-group" data-toggle="buttons">
  <label class="btn btn-secondary active">
    <input type="checkbox" checked autocomplete="off"> 체크박스 1 (기본선택)
  </label>
  <label class="btn btn-secondary">
    <input type="checkbox" autocomplete="off"> 체크박스 2
  </label>
  <label class="btn btn-secondary">
    <input type="checkbox" autocomplete="off"> 체크박스 3
  </label>
</div>
```

HTML

체크박스 1 (기본선택) 체크박스 2 체크박스 3

```html
<div class="btn-group" data-toggle="buttons">
  <label class="btn btn-secondary active">
    <input type="radio" name="options" id="option1" autocomplete="off" checked> 라디오 1
 (기본선택)
  </label>
  <label class="btn btn-secondary">
    <input type="radio" name="options" id="option2" autocomplete="off"> 라디오 2
  </label>
  <label class="btn btn-secondary">
    <input type="radio" name="options" id="option3" autocomplete="off"> 라디오 3
  </label>
</div>
```

HTML

Methods(메소드):

```
$().button('toggle')
```

HTML

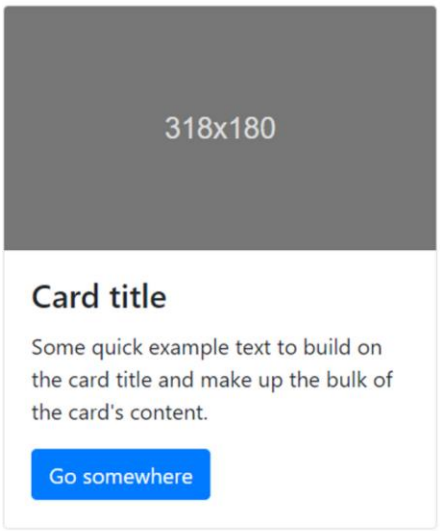
푸시 상태를 토글할 수 있습니다. 버튼에 활성화된 상태를 지정해보세요.

6. CARD

부트스트랩의 Card(카드)는 다양한 컨텐츠를 포함할 수 있는 확장성을 지니고 있습니다. 부트스트랩 버전 3에 익숙하시다면 panel(패널)이나 well(웰), thumbnail(텀브네일)로 대체가 가능합니다. Card(카드)는 Flexbox(플렉스박스)로 제작되어 쉽게 정렬할 수 있으며 다른 부트스트랩 컴포넌트와 잘 섞일 수 있습니다.

```
<div class="card" style="width: 20rem;">
  <img class="card-img-top" src="...." alt="Card image cap">
  <div class="card-body">
    <h4 class="card-title">Card title</h4>
    <p class="card-text">
Some quick example text to build on the card title
and make up the bulk of the card's content.
    </p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

HTML



카드의 body를 작성하기 위해서는 .card-body 클래스를 사용하면 됩니다. 아래 보이는 것이 기본 골격입니다.

```html
<div class="card">
  <div class="card-body">
    This is some text within a card body.
  </div>
</div>
```

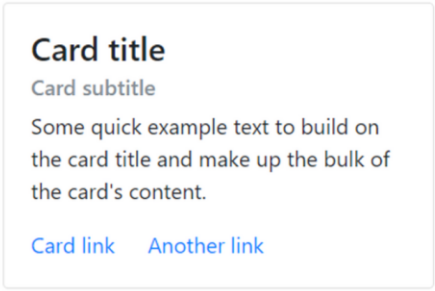
HTML

This is some text within a card body.

다음은 공식 홈페이지에서 제공하고 있는 Card에 타이틀, 텍스트 링크 작성 방법입니다.

Titles, text, and links(타이틀, 텍스트, 링크):

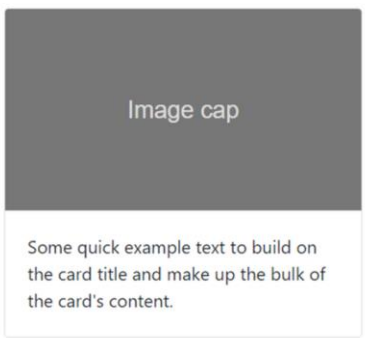
- 타이틀은 `<h*>` 태그에 `.card-title`을 추가
- 서브타이틀은 `<h*>` 태그에 `.card-subtitle`을 추가
- `*.card-body item`에 `.card-title`과 `.card-subtitle`을 배치하면 알맞게 정렬
- 링크는 `<a>` 태그에 `.card-link`를 추가하면 사용할 수 있음



Card안에서 이미지를 사용하는 방법입니다.

- `.card-img-top`은 이미지를 카드의 윗 부분에 놓아줍니다.
- `.card-text`를 이용하여 텍스트는 카드에 추가될 수 있습니다.
- `.card-text`안의 텍스트는 HTML 태그로 꾸밀 수 있습니다.

```HTML
<div class="card" style="width: 20rem;">
  <div class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content
    </p>
  </div>
</div>
```



Card안에도 리스트 그룹을 만들 수 있습니다. Flush list group(플러시 리스트 그룹)을 이용해서 아래와 같이 카드 안에 컨텐츠 리스트를 만들어 보세요.

```html
<div class="card" style="width: 20rem;">
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
</div>
```

HTML

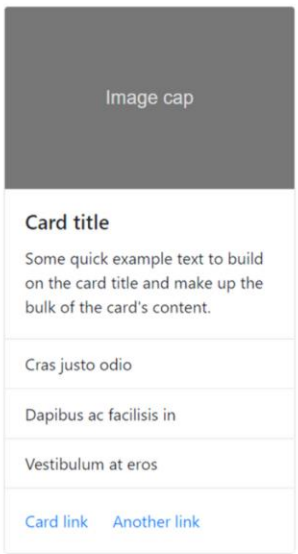
```html
<div class="card" style="width: 18rem;">
  <div class="card-header">
    Featured
  </div>
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
</div>
```

HTML

카드를 보다 더 다체롭게 꾸미기 위해 여러 컨텐츠를 믹스해보세요. 아래 예제는 한번 타이핑을 해보시길 추천해 드립니다.

```html
<div class="card" style="width: 18rem;">
  <img class="card-img-top" src=".../100px180/?text=Image cap" alt="Card image cap">
  <div class="card-body">
  <h5 class="card-title">Card title</h5>
  <p class="card-text">
    Some quick example text to build on the card title
    and make up the bulk of the card's content.
  </p>
  </div>
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
  <div class="card-body">
    <a href="#" class="card-link">Card link</a>
    <a href="#" class="card-link">Another link</a>
  </div>
</div>
```

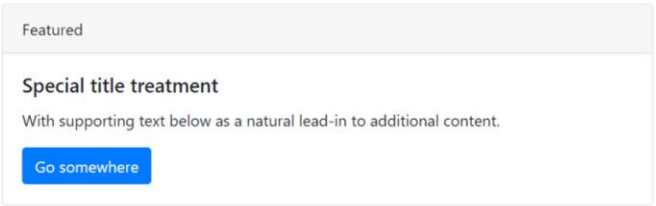
HTML



카드 안에 해더와 푸터를 추가할 수 있습니다.

```html
<div class="card">
  <div class="card-header">
    Featured
  </div>
  <div class="card-body">
    <h5 class="card-title">
      Special title treatment
    </h5>
    <p class="card-text">
      With supporting text below
      as a natural lead-in to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
</div>
```

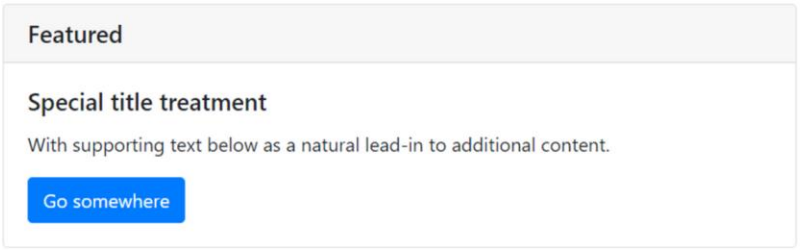
HTML



Card header(카드 헤더)는 <h*> 요소에 **.card-header**를 추가하여 스타일링 할 수 있습니다.

```html
<div class="card">
  <h5 class="card-header">
    Featured
  </h5>
  <div class="card-body">
    <h5 class="card-title">
      Special title treatment
    </h5>
    <p class="card-text">
      With supporting text below
      as a natural lead-in to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
</div>
```

HTML



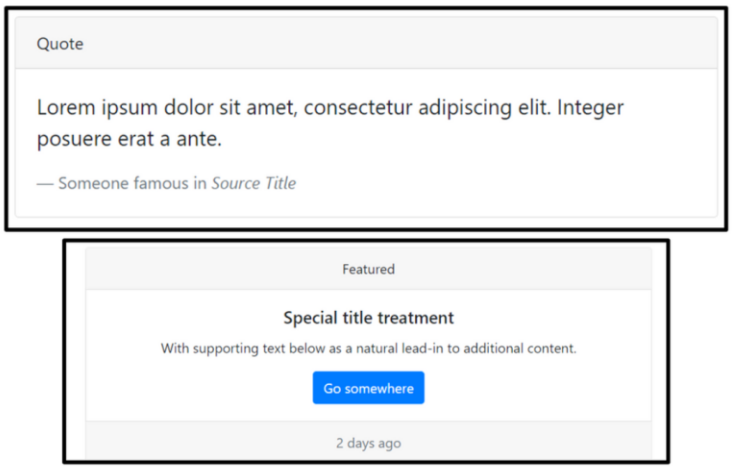
Header and footer(헤더와 풋터)

```
<div class="card">
  <div class="card-header"
    Quote
  </div>
  <div class="card-body">
    <blockquote class="blockquote mb-0">
      <p>
        Lorem ipsum dolor sit amet,
        consectetur adipiscing elit.
        Integer posuere erat a ante.
      </p>
      <footer class="blockquote-footer">
        Someone famous in
      <cite title="Source Title">
        Source Title</cite>
      </footer>
    </blockquote>
  </div>
</div>
```

HTML

```html
<div class="card text-center">
  <div class="card-header">
    Featured
  </div>
  <div class="card-body">
    <h5 class="card-title">
      Special title treatment
    </h5>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
  <div class="card-footer text-muted">
    2 days ago
  </div>
</div>
```

HTML



Sizing(크기 조정)

2. Using utilities(유틸리티 사용하기):

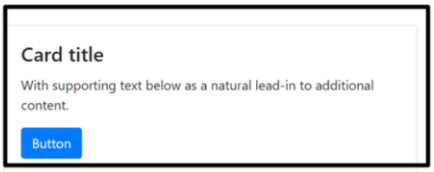
빠르게 카드의 너비를 맞출 수 있는 sizing utilities를 사용해 보세요.

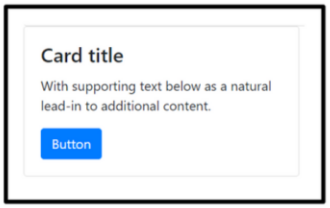
```html
<div class="card w-75">
  <div class="card-body">
    <h4 class="card-title">
      Card title
    </h4>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Button
    </a>
  </div>
</div>
```

HTML

```html
<div class="card w-50">
  <div class="card-body">
    <h4 class="card-title">
      Card title
    </h4>
    <p class="card-text">
      with supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Button
    </a>
  </div>
</div>
```

HTML



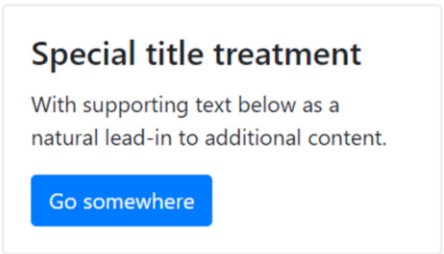


3. Using custon CSS(사용자 정의 CSS 사용하기):

너비를 맞추기 위해 stylesheets(스타일시트)나 inline style(인라인 스타일)을 통해 사용자 정의 CSS를 사용해 보세요.

```html
<div class="card" style="width: 20rem;">
  <div class="card-body">
    <h4 class="card-title">
      Special title treatment
    </h4>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
</div>
```

HTML



Text alignment(텍스트 정렬):

Text align classes를 통해 전체 혹은 일부분에서 텍스트 정렬을 빠르게 바꿀 수 있습니다.

```html
<div class="card" style="width: 20rem;">
  <div class="card-body">
    <h4 class="card-title">
      Special title treatment
    </h4>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
</div>
```

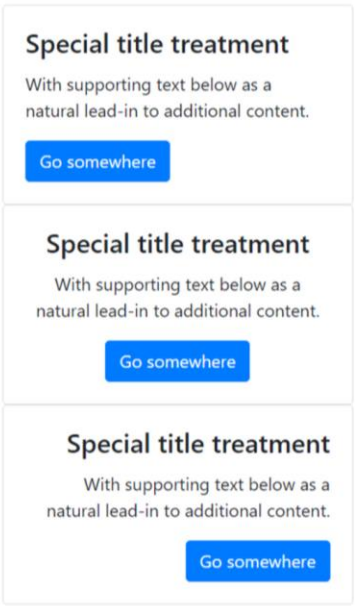
HTML

```html
<div class="card text-center" style="width: 20rem;">
  <div class="card-body">
    <h4 class="card-title">
      Special title treatment
    </h4>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere </a>
  </div>
</div>
```

HTML

```html
<div class="card text-right" style="width: 20rem;">
  <div class="card-body">
    <h4 class="card-title">
      Special title treatment
    </h4>
    <p class="card-text">
      With supporting text below
      as a natural lead-in
      to additional content.
    </p>
    <a href="#" class="btn btn-primary">
      Go somewhere
    </a>
  </div>
</div>
```

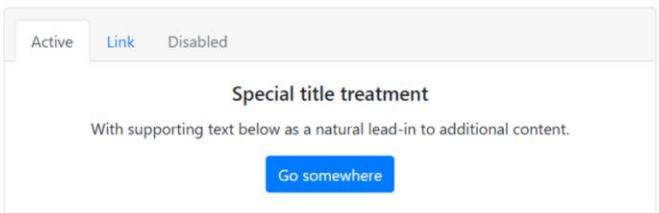
HTML



부트스트랩은 카드 내 네비게이션을 허용하고 있습니다. nav components를 이용해서 카드의 header(헤더)나 block(블록)에 네비게이션을 추가해보세요. (예제 계속)

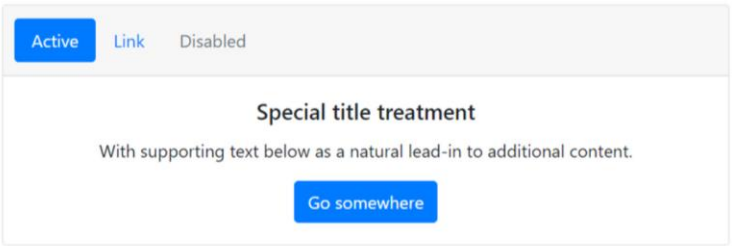
```html
<div class="card text-center">
  <div class="card-header">
    <ul class="nav nav-tabs card-header-tabs">
      <li class="nav-item"> <a class="nav-link active" href="#"> Active </a> </li>
      <li class="nav-item"> <a class="nav-link" href="#">Link </a> </li>
      <li class="nav-item"> <a class="nav-link disabled" href="#"> Disabled </a> </li>
    </ul>
  </div>
  <div class="card-body">
    <h5 class="card-title"> Special title treatment </h5>
    <p class="card-text"> With supporting text below as a natural lead-into additional co
ntent. </p>
    <a href="#" class="btn btn-primary">Go somewhere </a>
  </div>
</div>
```

HTML



```html
<div class="card text-center">
  <div class="card-header">
    <ul class="nav nav-pills card-header-pills">
      <li class="nav-item"> <a class="nav-link active" href="#"> Active </a> </li>
      <li class="nav-item"> <a class="nav-link" href="#">Link </a> </li>
      <li class="nav-item"> 34 <a class="nav-link disabled" href="#">Disabled </a> </li>
    </ul>
  </div>
  <div class="card-body">
    <h5 class="card-title"> Special title treatment </h5>
    <p class="card-text"> With supporting text below as a natural lead-into additional co
ntent. </p>
    <a href="#" class="btn btn-primary"> Go somewhere </a>
  </div>
</div>
```

HTML

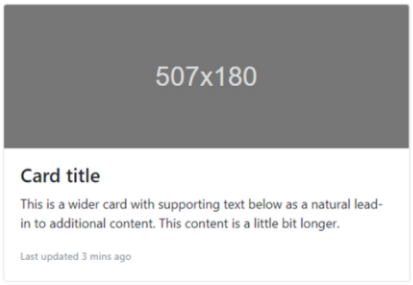


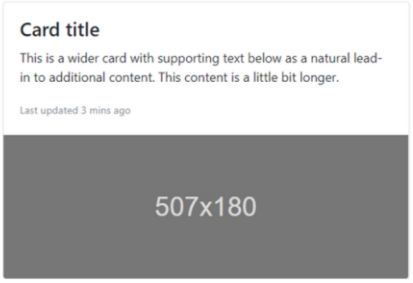
부트스트랩에는 카드에 이미지 작업을 위한 몇가지 옵션이 있습니다. 예를 들어 카드의 양쪽 끝에 `image caps`를 추가하거나 카드 내용으로 이미지를 겹치거나 이미지를 카드에 간단하게 끼워 넣기를 할 수 있습니다.

Header(헤더), footer(풋터)와 마찬가지로 card(카드)는 상단 및 하단에 "image caps"이 포함될 수 있습니다.

```html
<div class="card mb-3">
  <img class="card-img-top" src="..." alt="Card image cap">22
  <div class="card-body">
    <h4 class="card-title"> Card title </h4>
    <p class="card-text">
      This is a wider card with supporting text below as a natural
      lead-in to additional content. This content is a little bit longer.
    </p>
    <p class="card-text">
      <small class="text-muted"> Last updated 3 mins ago</small>
    </p>
  </div>
</div>

<div class="card">
  <div class="card-body">
    <h4 class="card-title"> Card title </h4>
    <p class="card-text">
      This is a wider card with supporting text below as a natural
      lead-in to additional content. This content is a little bit longer.
    </p>
    <p class="card-text">
      <small class="text-muted">
        Last updated 3 mins ago
      </small>
    </p>
  </div>
  <img class="card-img-bottom" src="..." alt="Card image cap">
</div>
```

HTML

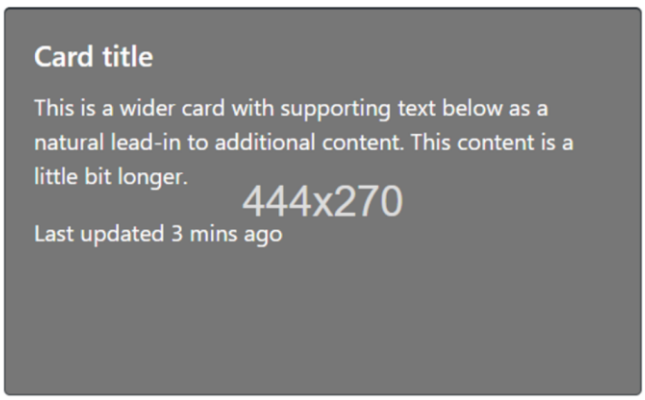




부트스트랩에서는 이미지를 카드 배경으로 바꾸고 카드의 텍스트를 오버레이할 수 있는 방법을 제공하고 있습니다.

```html
<div class="card bg-dark text-white">
  <img class="card-img"
  src=".../100px270/#55595c:#373a3c/text:Card image"
  alt="Card image">
  <div class="card-img-overlay">
    <h5 class="card-title"> Card title </h5>
    <p class="card-text">
      This is a wider card
      with supporting text
      below as a natural
      lead-in to additional content.
      This content is a little bit longer.
    </p>
    <p class="card-text">
      Last updated 3 mins ago
    </p>
  </div>
</div>
```

HTML



카드는 배경, 테두리, 색깔을 바꿀 수 있습니다. 아래 내용을 참고하세요.

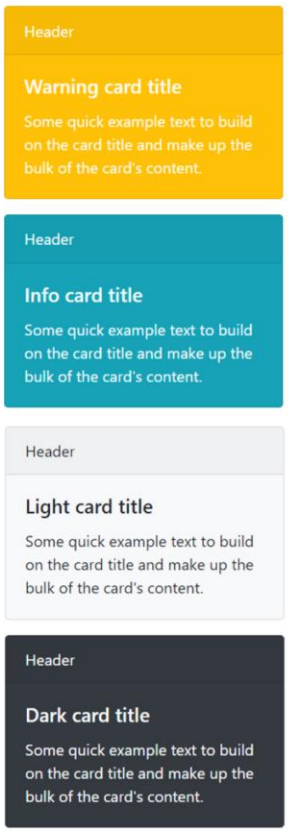
```html
<div class="card text-white bg-primary mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
    <div class="card-body">
      <h5 class="card-title">Primary card title</h5>
      <p class="card-text">
        Some quick example text to build on the card title
        and make up the bulk of the card's content.</p>
    </div>
</div>
<div class="card text-white bg-secondary mb-3" style="max-width: 18rem;">
  <div class="card-header">Header </div>
  <div class="card-body">
    <h5 class="card-title">Secondary card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card text-white bg-success mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Success card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card text-white bg-danger mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Danger card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
```

HTML


```
<div class="card text-white bg-warning mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Warning card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card text-white bg-info mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
```

```
  <div class="card-body">
    <h5 class="card-title">Info card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card bg-light mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Light card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card text-white bg-dark mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Dark card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
```

HTML

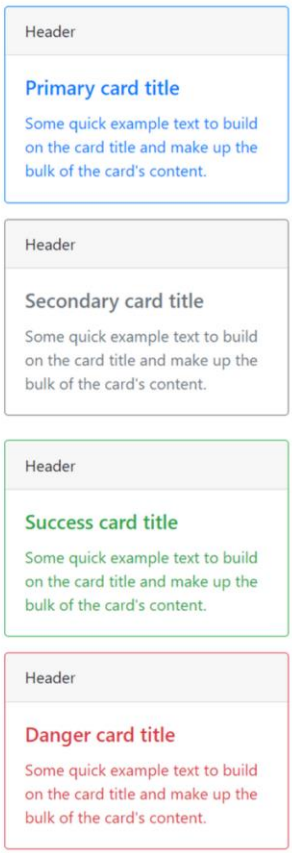


카드에도 테두리를 줄 수 있습니다. border-color를 바꾸기 위해서는 border utilities를 사용하면 됩니다. 부모요소의 카드나 카드 컨텐츠에 .text-{color} 클래스를 추가하면 됩니다.

```
<div class="card border-primary mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-primary">
    <h5 class="card-title">Primary card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
<div class="card border-secondary mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-secondary">
    <h5 class="card-title">Secondary card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
```

```html
<div class="card border-success mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-success">
    <h5 class="card-title">Success card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content. </p>
  </div>
</div>
<div class="card border-danger mb-3" style="max-width: 18rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-danger">
    <h5 class="card-title">Danger card title</h5>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
</div>
```

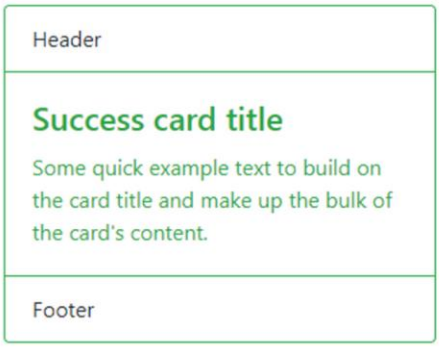
HTML



카드 헤더와 풋터 테두리를 바꾸거나 지우면 좀 더 세련된 카드를 구현할 수 있습니다. `.bg-transparent` 를 이용해서 background-color를 제거할 수도 있습니다.

```html
<div class="card border-success mb-3" style="max-width: 20rem;">
  <div class="card-header bg-transparent border-success">Header</div>
  <div class="card-body text-success">
    <h4 class="card-title">Success card title</h4>
    <p class="card-text">
      Some quick example text to build on the card title
      and make up the bulk of the card's content.
    </p>
  </div>
  <div class="card-footer bg-transparent border-success">Footer</div>
</div>
```

HTML



카드 안에 컨텐츠를 추가하기 위해서 부트스트랩은 카드의 레이아웃작업의 옵션들을 포함하고 있습니다.

그룹의 카드들을 같은 너비와 높이의 붙어있는 columns(열)로 바꿔줄 수 있습니다.

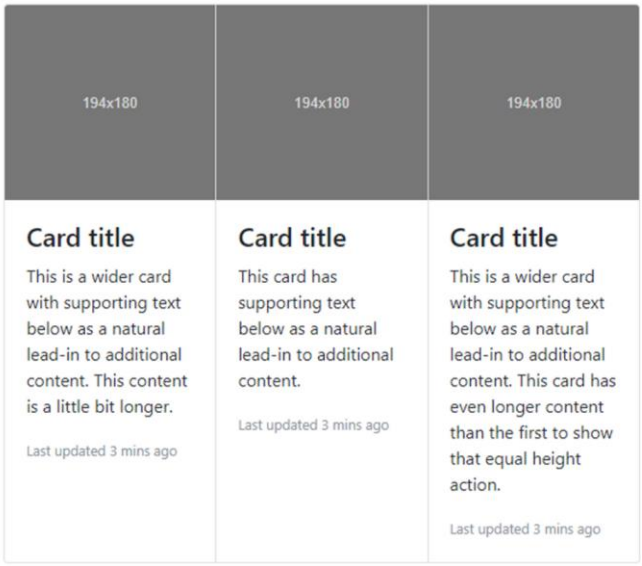
카드 그룹은 동일한 사이즈로 만들기 위해 display: flex;를 사용할 수 있습니다. 풋터를 사용해서 카드 그룹을 사용할 경우, 컨텐츠는 자동적으로 정렬됩니다. 코드는 다음장에 넣어두었습니다.

Card groups(카드 그룹) (예시 계속)

```html
<div class="card-group">
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural
        lead-in to additional content. This content is a little bit longer.
      </p>
      <p class="card-text">
        <small class="text-muted"> Last updated 3 mins ago </small>
      </p>
    </div>
  </div>
```

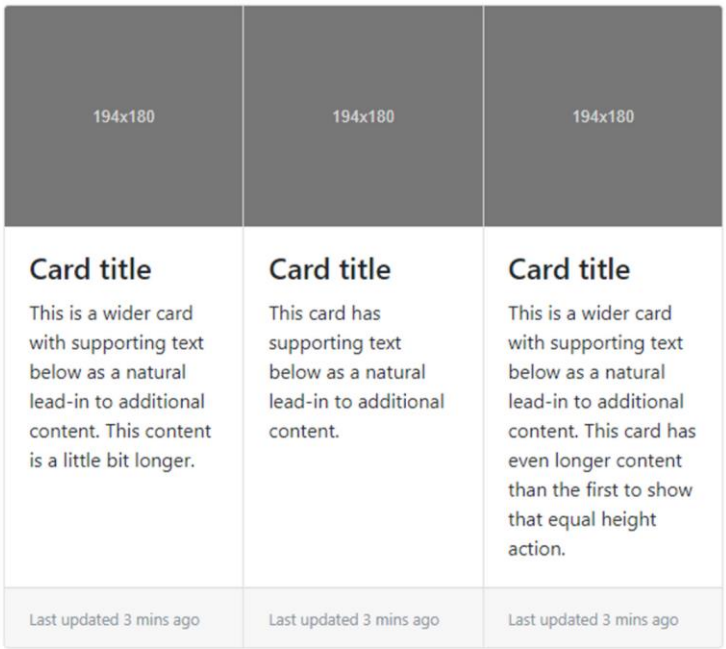
```
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This card has supporting text below as a natural lead-in
        to additional content.
      </p>
      <p class="card-text">
        <small class="text-muted"> Last updated 3 mins ago</small>
      </p>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural lead-in to additiona
l content. This card has even longer content than the first to show that equal height act
ion.
      </p>
      <p class="card-text">
        <small class="text-muted">
          Last updated 3 mins ago
        </small>
      </p>
    </div>
  </div>
</div>
```

HTML



```
<div class="card-group">
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card
        with supporting text
        below as a natural lead-in to
        additional content.
        This content is a little bit longer.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This card has supporting text
        below as a natural lead-in
        to additional content.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural lead-in to additiona
l content. This card has even longer content than the first to show that equal height act
ion.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
</div>
```

HTML



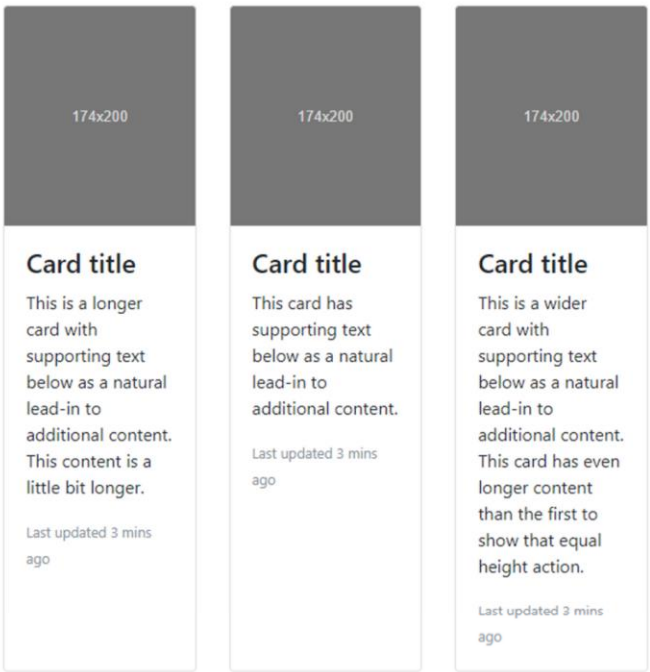
너비와 높이가 동일하지만 붙어있지 않은 것을 카드 데크라고 말합니다. 푸터를 사용해서 카드 그룹을 사용할 경우 컨텐츠는 자동으로 정렬됩니다.

Card decks(카드 데크) (예제 계속)

```
<div class="card-deck">
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a longer card with supporting text
        below as a natural lead-in to additional content.
        This content is a little bit longer.
      </p>
      <p class="card-text">
      <small class="text-muted">Last updated 3 mins ago</small>
      </p>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
      This card has supporting text below as a natural lead-in to additional content.
      </p>
      <p class="card-text">
```

```
        <small class="text-muted">Last updated 3 mins ago</small>
      </p>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text
        below as a natural lead-in to additional content.
        This card has even longer content than the first
        to show that equal height action.
      </p>
      <p class="card-text">
        <small class="text-muted">Last updated 3 mins ago</small>
      </p>
    </div>
  </div>
</div>
```

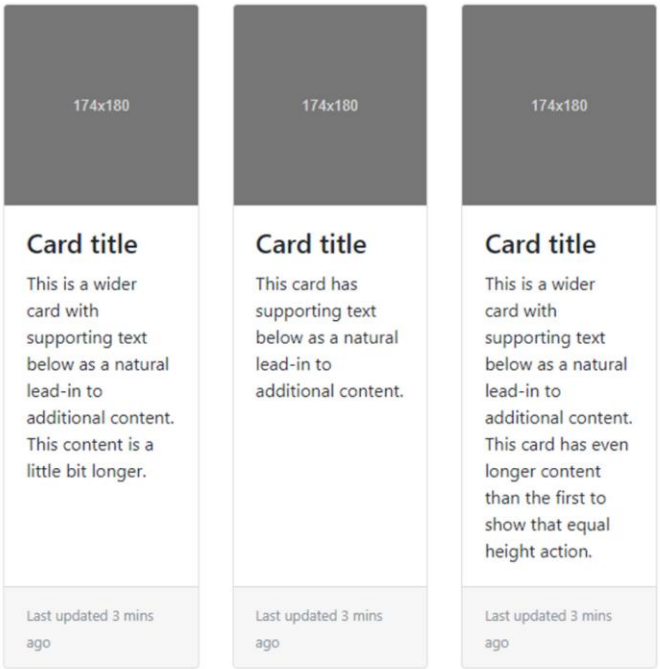
HTML



Card decks(카드 데크)(예제계속)

```
<div class="card-deck">
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural lead-in
        to additional content. This content is a little bit longer.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This card has supporting text
        below as a natural lead-in to additional content.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural lead-in
        to additional content. This card has even longer content than the first
        to show that equal height action.
      </p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
</div>
```

HTML



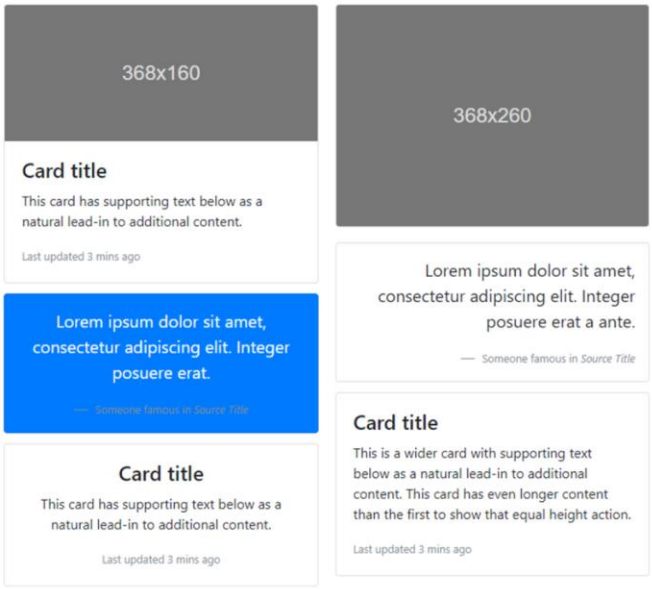
카드는 `.card-columns`에서 CSS를 이용해 감싸주면 벽돌 형으로 정렬된 열을 구성 할 수 있습니다. 카드는 flexbox(플렉스 박스)가 아닌 CSS 열 속성을 사용하여 쉽게 정렬 할 수 있습니다. 카드는 위에서 아래로, 왼쪽에서 오른쪽으로 정렬됩니다.

카드 열이 있는 마일리지는 다를 수 있습니다. 카드가 열을 가로 지르지 못하게 하려면 `display: inline-block`에 `column-break-inside: avoid`로써 설정해야 합니다.

```
<div class="card-columns">
  <div class="card">
    <img class="card-img-top" src="..." alt="Card image cap">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This card has supporting text below as a natural lead-in to additional content.
      </p>
      <p class="card-text"> <small class="text-muted">
        Last updated 3 mins ago</small></p>
    </div>
  </div>
  <div class="card bg-primary text-white text-center p-3">
    <blockquote class="blockquote mb-6">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat.</
p>
      <footer class="blockquote-footer">
        <small> Someone famous in
          <cite title="Source Title">
            Source Title</cite>
```

```
        </small>
      </footer>
    </blockquote>
  </div>
  <div class="card text-center">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This card has supporting text
        below as a natural lead-in
        to additional content.</p>
      <p class="card-text">
        <small class="text-muted">
          Last updated 3 mins ago
        </small>
      </p>
    </div>
  </div>
  <div class="card">
    <img class="card-img" src="..." alt="Card image">
  </div>
  <div class="card p-3 text-right">
    <blockquote class="blockquote mb-8">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a
 ante.</p>
      <footer class="blockquote-footer">
        <small class="text-muted"> Someone famous in
          <cite title="Source Title">Source Title</cite>
        </small>
      </footer>
    </blockquote>
  </div>
  <div class="card">
    <div class="card-body">
      <h4 class="card-title">Card title</h4>
      <p class="card-text">
        This is a wider card with supporting text below as a natural lead-in to additiona
l content. This card has even longer content than the first to show that equal height act
ion.</p>
      <p class="card-text">ht
        <small class="text-muted">
          Last updated 3 mins ago
        </small>
      </p>
    </div>
  </div>
</div>
```

HTML



추가적인 코드를 이용해서 카드를 커스터마이징할 수 있습니다. CSS의 `.card-columns`를 이용해서 커스터 마이징 해보세요.

```
.card-columns{ @include media-breakpoint-only(lg) { column-count: 4; } @include media-breakpoint-only(xl) { column-count: 5; }}
```

HTML

7. 캐러셀

캐러셀은 이미지나 텍스트 슬라이드를 구성하는 슬라이드 쇼 입니다. 마지막 캐러셀 이미지에서는 다시 처음 슬라이드로 돌아오게 됩니다.

부트스트랩 공식 홈페이지에서는 캐러셀을 'CSS 3D 트랜스폼과 약간의 자바스크립트로 제작된 시리즈 컨텐츠를 순환하는 슬라이드 쇼'라고 말하고 있습니다. 주의하셔야 할 부분은 캐러셀은 중첩을 허락하지 않으며 웹 접근성 표준을 준수하지 않는다는 것입니다. 또한 소스에서 JS를 빌드하는 경우 util.js가 필요합니다.

캐러셀은 자동으로 슬라이드 크기를 표준화하지 않습니다. 따라서 컨텐츠의 크기를 적절하게 조정하려면 추가 유틸리티 또는 사용자 정의 스타일을 사용해야 할 수도 있습니다.

.active 클래스 슬라이드당 하나씩 추가되어야 합니다. 그렇지 않으면 캐러셀이 보이지 않습니다. 하나의 페이지에 다양한 캐러셀을 사용할 경우, 선택적인 컨트롤을 위해 .carousel에 각각의 고유한 아이디를 부여해주세요. 컨트롤과 인디케이터 요소들의 경우 반드시 .carousel요소의 아이디를 맞추는 데이터 맞춤 속성을 가져야 합니다.

다음은 슬라이드만 있는 캐러셀입니다.

```html
<div id="carouselExampleSlidesOnly"
class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img class="d-block w-100" src="..." alt="첫번째 슬라이드">
    </div>
    <div class="carousel-item">
      <img class="d-block w-100" src="..." alt="두번째 슬라이드">
    </div>
  </div>
</div>
```

HTML

이전과 다음 컨트롤이 필요한 경우 아래와 같이 사용할 수 있습니다.

```html
<div id="carouselExampleControls"
class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img class="d-block w-100" src="..." alt="첫번째 슬라이드">
    </div>
  </div>
  <a class="carousel-control-prev"
  href="#carouselExampleControls"
  role="button" data-slide-"prev">
    <span class="carousel-control-prev-icon"
    aria-hidden="true"></span>
    <span class="sr-only">이전 </span>
  </a>
  <a class="carousel-control-next
  href="#carouselExampleControls"
  role="button" data-slide="next">
    <span class="carousel-control-next-icon"
    aria-hidden="true"></span>
    <span class="sr-only">다음 </span>
  </a>
</div>
```

HTML

컨트롤과 함께 인디케이터를 캐러셀에 추가할 수 있습니다. 슬라이드 하단에 보이는 것이 인디케이터 입니다.

```HTML
<div id="carouselExample Indicators"
class="carousel slide" data-ride-"carousel">
  <ol class="carousel-indicators">
    <li data-target="#carouselExample Indicators"
    data-slide-to="0" class="active"></li>
    <li data-target="#carouselExample Indicators"
    data-slide-to="1"></li>
    <li data-target="#carouselExample Indicators"
    data-slide-to="2"></li>
  </ol>
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img class="d-block w-100"
      src="..." alt="첫번째 슬라이드">
    </div>
    <div class="carousel-item">
    <img class="d-block w-100" src="..." alt="두번째 슬라이드">
    </div>
  </div>
  <a class="carousel-control-prev"
  href="#carouselExample Indicators"
  role="button" data-slide="prev">
    <span class="carousel-control-prev-icon"
    aria-hidden="true"></span>
    <span class="sr-only">이전</span>
    </a>
  <a class="carousel-control-next"
  href="#carousel Example Indicators"
  role="button" data-slide="next">
      <span class="carousel-control-next-icon"
      aria-hidden="true"></span>
      <span class="sr-only">다음</span>
  </a>
</div>
```

`.carousel-item` 내에 `.carousel-caption` 요소를 사용하여 슬라이드에 캡션을 추가할 수 있습니다. 선택적으로 display utilites를 사용하여 작은 뷰포트에 숨길 수 있습니다. 처음에는 `.d-none`으로 숨기고 `.d-md-block`으로 중각 크기의 디바이스에서 다시 가져옵니다.

```html
<div class="carousel-item">
  <img src="..." alt="...">
  <div class="carousel-caption d-none d-md-block">
    <h3>...</h3>
    <p>...</p>
  </div>
</div>
```

HTML

부트스트랩에 Crossfade(크로스 페이드)는 이미지가 animate(애니메이트)되는 대신 fadein fadeout 되는 fade transition(페이드 트랜지션)을 사용하는 방법입니다. 캐러셀에 .carousel-fade 를 추가하면 됩니다.

```html
<div id="carousel ExampleFade"
class="carousel slide carousel-fade"
data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img class="d-block w-100"
      src=".../800x400?auto=yes&bg=777&fg=555&text=First slide"
      alt="First slide">
    </div>
  </div>
  <a class="carousel-control-prev"
  href="#carouselExampleFade"
  role="button" data-slide-"prev">
    <span class="carousel-control-prev-icon"
    aria-hidden="true"></span>
    <span class="sr-only">Previous</span>
  </a>
  <a class="carousel-control-next"
  href="#carouselExampleFade"
  role="button" data-slide="next">
    <span class="carousel-control-next-icon"
    aria-hidden="true"></span>
    <span class="sr-only">Next</span>
  </a>
</div>
```

HTML

데이터 속성을 사용하여 캐러셀의 위치를 쉽게 제어할 수 있습니다. data-slide는 prev 또는 next 키워드를 사용하여 현재 위치를 기준으로 슬라이드 위치를 변경합니다. 또는 data-slide-to를 사용해서 슬라이드 포인터를 0으로 시작하는 특정 인덱스로 이동하는 슬라이드 포인터 data-slide-to="2"에 원시 슬라이드 인덱스를 전달합니다.

`data-ride="carousel"` 속성은 캐러셀 페이지 로드시 애니메이션으로 표시하는데 사용됩니다. 동일한 캐러셀을 JavaScript 초기화 (중복 및 불필요한)방식과 함께 사용할 수는 없습니다.

Via JavaScript(JavaScript 방식):

`$('.carousel').carousel()` 를 사용해서 수동으로 캐러셀을 호출할 수 있습니다.

8. 마무리하며

부트스트랩에 모든 기능을, 모든 문법을 다 익히고 숙련되야만 부트스트랩을 잘 사용할 수 있을까요? 그렇지 않습니다. 부트스트랩 유무료 템플릿을 다운로드 받는 방법이 있고, 그 방법을 더 추천해 드립니다. 아무리 잘 사용하더라도, 전문가의 손길보다 뛰어나게 홈페이지를 작성할 수 없습니다. 기회비용을 고려하여, 2만원 ~ 3만원에 부트스트랩 템플릿을 다운로드 받는 다는 것은 훌륭한 대안이 될 것이에요.

그래도 처음부터 끝까지 직접 만들고 싶으시다면, 부트스트랩 보다는 직접 한땀한땀 처음부터 끝까지 만들어보시길 권해드립니다.

DATE.

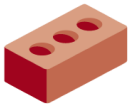
비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day3

웹서비스 개발 기획

1. 왜 홈페이지 기획을 하는가?
2. 무엇을 이용해서 기획을 하는가?
3. 마인드맵(FREEMIND)
4. 화면설계(KAKAO OVEN)
   카카오 오븐(https://ovenapp.io)



웹서비스 개발 기획

1. 왜 홈페이지 기획을 하는가?

2. 무엇을 이용해서 기획을 하는가?

3. 마인드맵(FREEMIND)

4. 화면설계(KAKAO OVEN & Figma)

    4.1 카카오 오븐(https://ovenapp.io)

    4.2 Figma(https://www.figma.com/)

기획을 하기 앞서 먼저 해야 하는 질문은 다음과 같습니다.

- 왜 웹 서비스를 만드는가?
- 핵심 요소는 무엇인가?
- 누가, 어떤 기기를 가지고 이 홈페이지를 방문하는가?
- 어떤 기능을 제공하는가?

1. 왜 홈페이지 기획을 하는가?

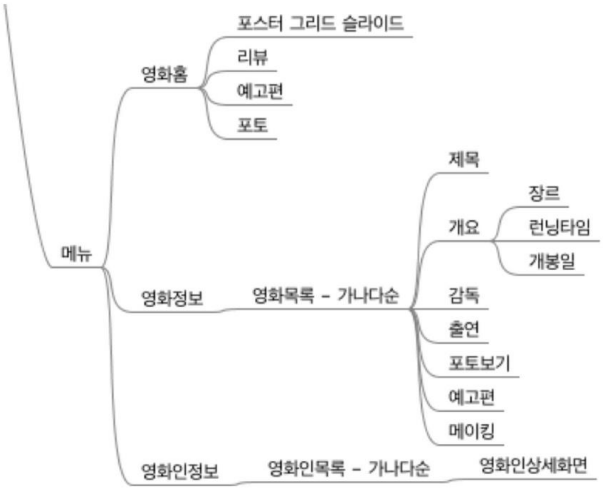
- 시행착오를 줄이고 원하는 결과물을 잘 구상하기 위해서

2. 무엇을 이용해서 기획을 하는가?

- 정보구조설계(I.A.)
- 화면설계(스토리보드)
- 정보구조설계 Information Architecture
    - 웹사이트의 정보(메뉴구조)를 단계별로 정리한 문서
    - 웹사이트 사이트맵 검색결과
      https://bit.ly/2uvjTVG

- 기능목록
  홈페이지를 구성하고 있는 메뉴와 기능 목록

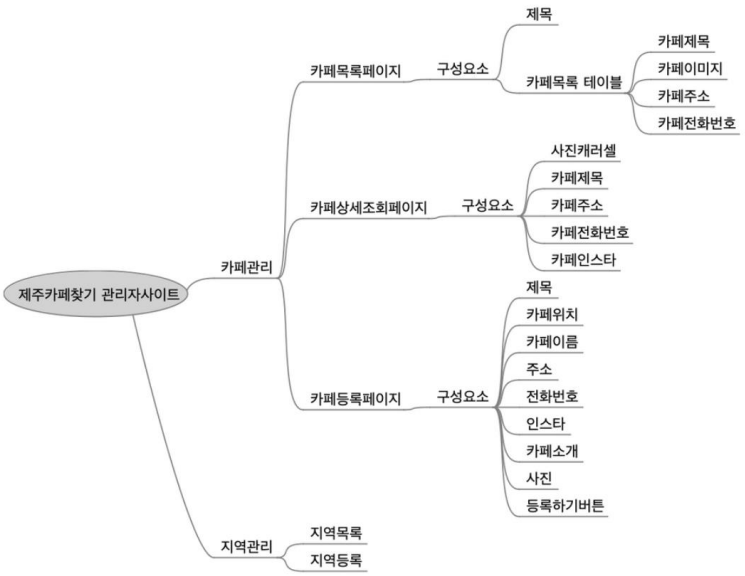
블록체인으로 구현하는 투표 시스템 - jeju BlockChain Vote



영화 추천 사이트 구현

3. 마인드맵(FREEMIND)

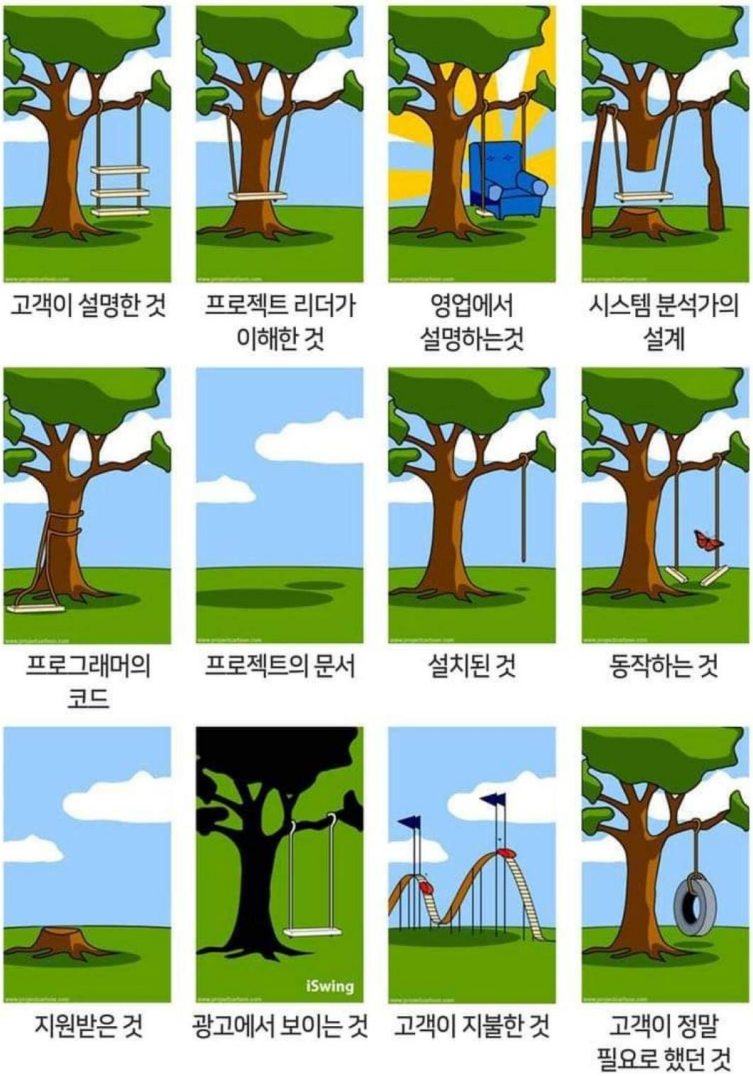
- 설치하기
- Freemind를 사용해서 제주카페관리자사이트 메뉴 만들어보기, Insert키를 이용해 가지를 칠 수 있습니다.



- 단축키 모음 :

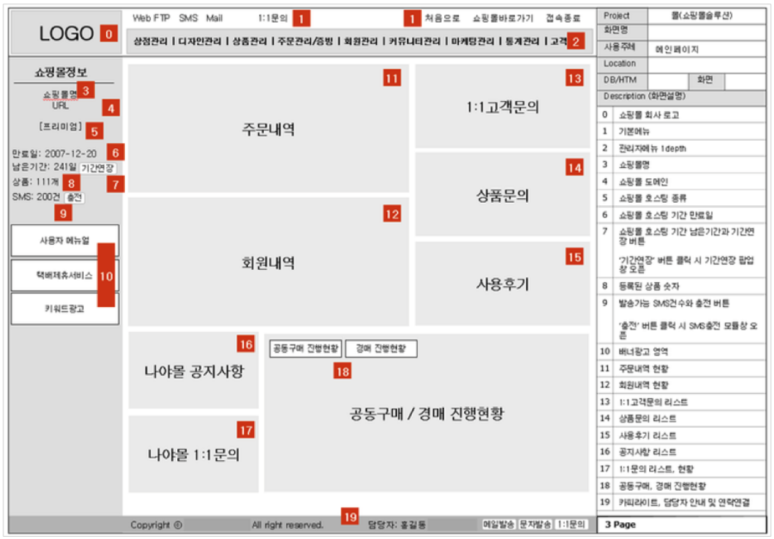
```
1. 새 마인드 맵 :
  Ctrl+N이나 파일>새파일을 선택하면 "새로운 마인드맵"으로 시작할 수 있습니다.

2. 노드 선택 및 수정 :
  노드를 클릭하면 선택하여 수정할 수 있고 F2를 클릭해도 됩니다.

3. 하위 노드 추가 :
  선택 노드에서 Insert 키를 누르면 하위 노드를 추가합니다.

4. 같은 레벨 추가 :
  선택 노드에서 Enter 키를 누르면 같은 레벨로 노드를 추가합니다.

5. 노드간 위치 바꾸기 :
  Ctrl+상하좌우 방향키로 노드 위치를 바꿀 수 있고, 마우스로 끌어다 놓기 해도 됩니다.

6. 맵 저장 :
  Ctrl+S 또는 파일>저장으로 파일을 저장할 수 있습니다.
```

4. 화면설계(KAKAO OVEN & Figma)

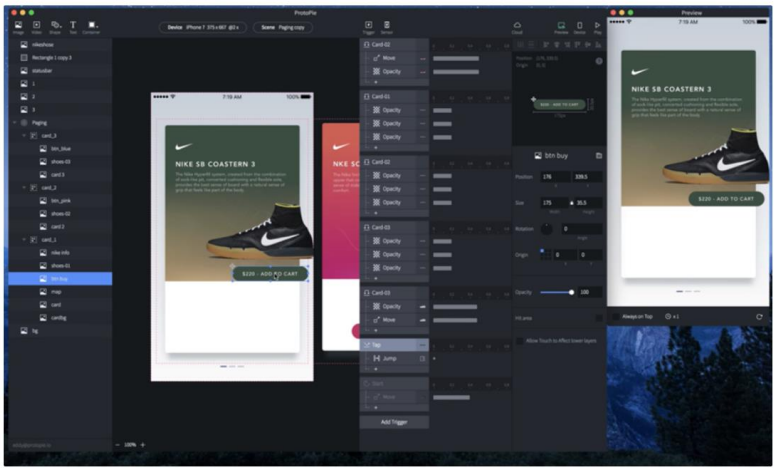


출처 : http://blog.naver.com/PostView.nhn?blogId=yingbbang&logNo=221473198909&from=search&redirect=Log&widgetTypeCall=true&directAccess=false

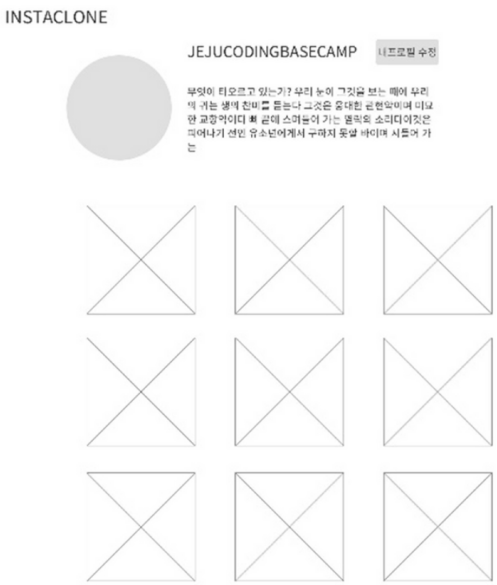
- 보통 프로젝트를 할 때는 혼자서 하는 게 아닌 디자이너, 기획자, 개발자와 같은 다양한 사람이 참여하여 이루어집니다. 이 때 화면설계는 아이디어 발안자가 아닌 다른 사람들이 작업을 이해하고 아이디어를 소통하는데 중요한 역할을 합니다. 화면설계는 프로젝트의 청사진이자 목적지이기도 합니다.
  만약 화면설계를 하지 않은 채 프로젝트를 진행한다면 위와 같은 결과가 벌어지기도 합니다.
- 스토리보드, 와이어프레임, 프로토타입



스토리보드



프로토타입

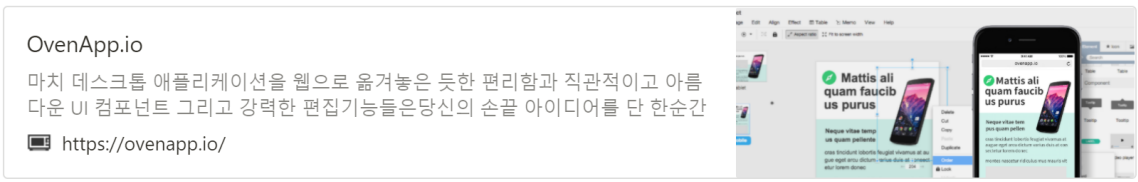


와이어 프레임

- 소통! 소통! 소통! 아무리 강조해도 지나치지 않습니다. 작업이 어느정도 마무리 된 다음에 소통을 하는 것이 아니라 끊임없이 중간과정을 지나치다 싶을 정도로 소통하셔서 의도를 정확히 반영하려는 노력이 필요합니다.

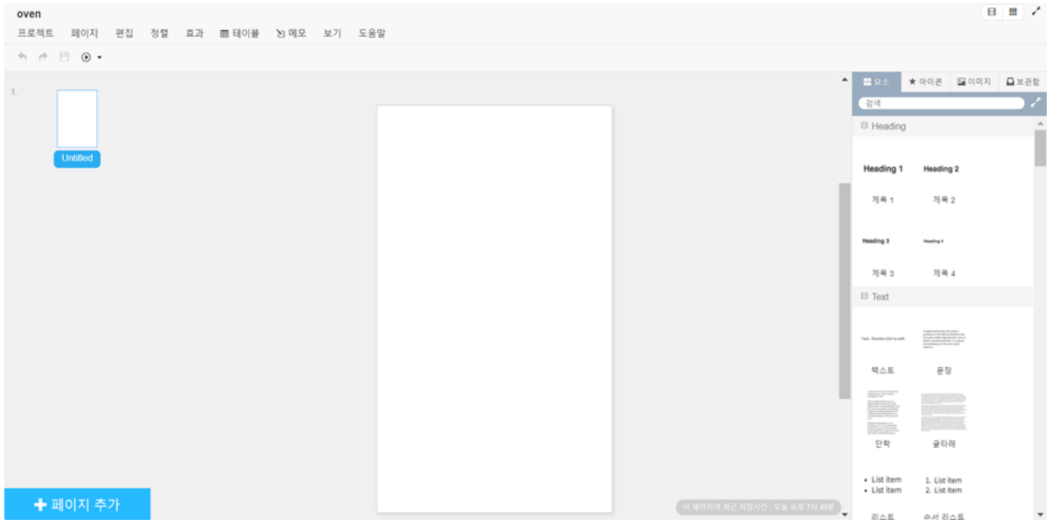
4.1 카카오 오븐(https://ovenapp.io)

- 앞서 한 질문들을 바탕으로 웹사이트를 제작하는데 큰 도움이 되는 도구들을 소개하겠습니다. 저희가 지금 만들고자 하는 웹사이트 뿐만 아니라 앞으로 프로젝트를 기획하는데 큰 도움이 되는 도구들입니다.
- Oven은 프로토타입을 만들기 적절한 툴입니다. input을 최소화 하여 화면 구성을 할 수 있습니다.

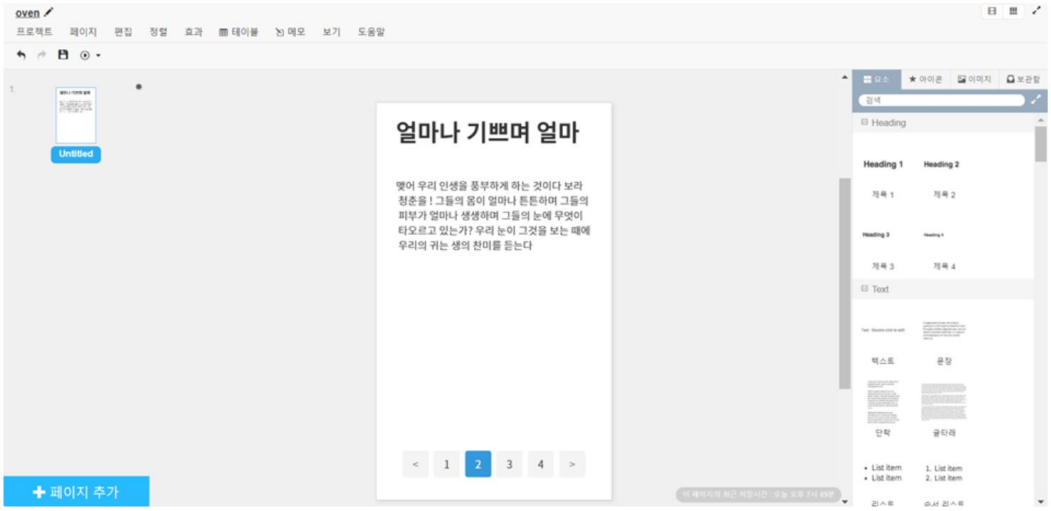


- 홈페이지에서 오븐을 소개하는 문구

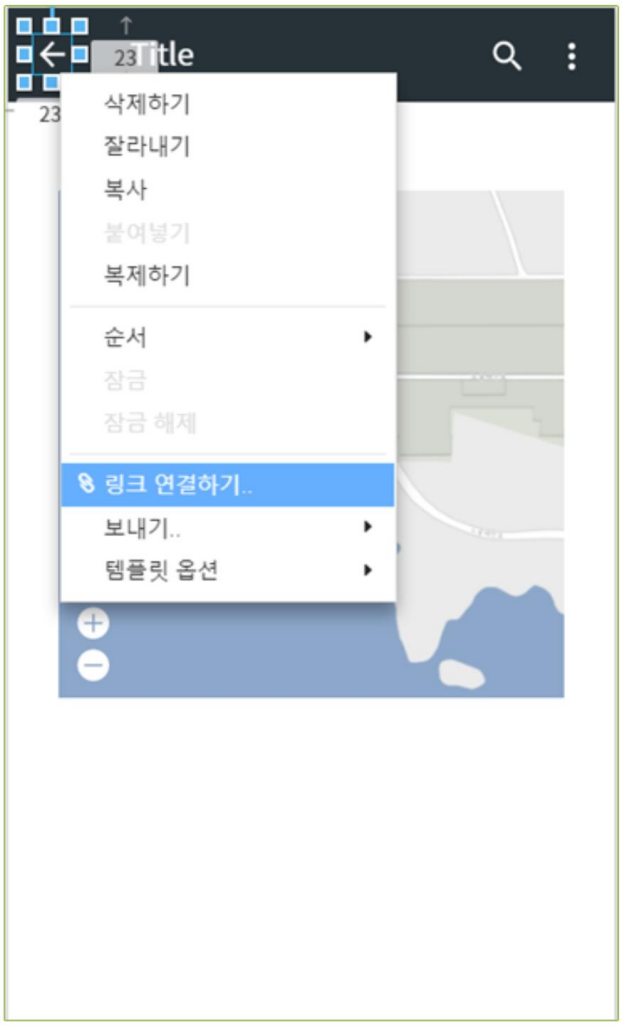
- 예시
  - 제주카페찾기 관리자사이트
    https://ovenapp.io/view/BRvoQsMUr3opY2hzLk70XNqQfVa7S6BS/RUKEb
  - 영화정보사이트
    https://ovenapp.io/view/02t1dEDUdhEKYtNy9fG7pwlWOwAHK7Qn/Ffr3y
- 기본기능
  - 페이지 추가 : 왼쪽 하단에 페이지 추가를 눌러 페이지를 추가해보세요!



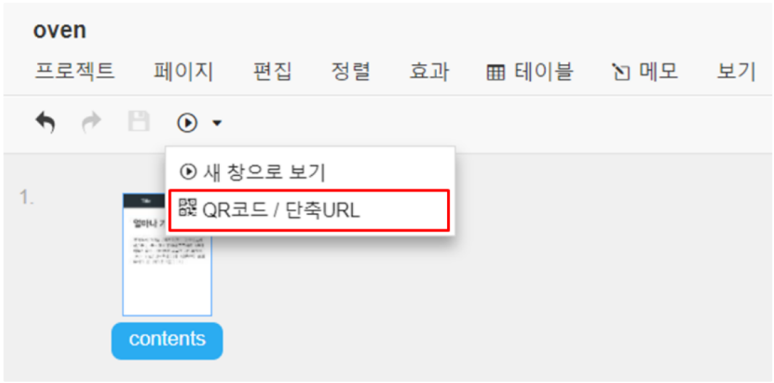
- 텍스트 및 이미지 추가 : 드래그 엔 드롭으로 각 요소를 추가할 수 있습니다.



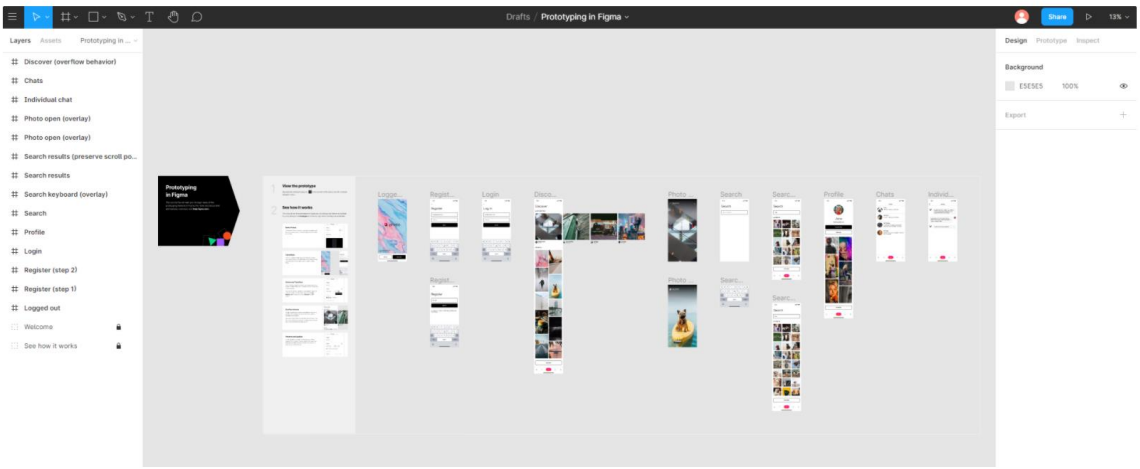
- 링크연결 : 각 버튼에 링크를 연결할 수 있습니다.



- 미리보기
- 공유하기 : 누군가에게 링크로 바로 공유할 수 있습니다.



4.2 Figma(https://www.figma.com/)



- 전문적인 디자인 툴도 있습니다. 공유와 설계가 자유로워 카카오 Oven처럼 쉽게 화면 설계를 할 수 있습니다. 다만 기능이 많다 보니 숙련이 되기까지 시간이 필요합니다.
- 수정을 하게 되면 공유받은 모든 인원의 workspace에 반영됩니다.
- 별도의 파일로 저장할 수도 있지만, 따로 저장 안해도 되는 클라우드 환경에서 작동합니다.
- Window, Mac에서 동일한 환경을 제공하고, 히스토리 관리가 가능하며 능숙해질 경우 팀과 디자이너에게 큰 만족도를 줍니다.

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day3

Git & Github

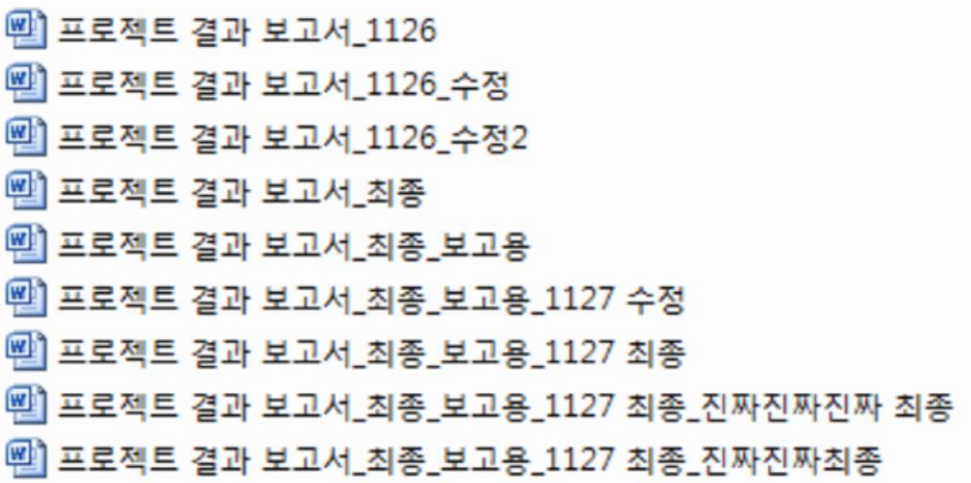
1. 소스코드 버전관리를 사용하지 않았을때 발생했던 문제점
2. 소스코드 버전관리를 통해 문제가 해결되는 사례
3. 소스코드 버전관리란?
4. Git이란?
5. Git 설치하기
6. 저장소 만들기
7. 수정하고 저장소에 저장하기
8. Branch
   8.1 브랜치의 사용예
9. Github와 Bitbucket
10. Atom을 이용한 Git 사용하기
11. Git에 관한 잡다한 이야기들

소스코드 버전관리를 사용하지 않았을때 발생했던 문제점

- 우리는 프로그래밍을 하면서 많은 소스코드 파일을 작성합니다. 작업을 할때 마다 종종 백업이 필요한 상황이 많습니다. 특히 어떤 작업을 마친 뒤 그 폴더를 통으로 백업하려고 합니다.
  공유를 위해서 폴더를 압축하고 다른사람에게 "폴더명_날짜.zip" 같이 파일명을 지어서 전송합니다.
- 하지만 이렇게 백업할 경우 불편한 점은 내가 수정한 내용을 구체적으로 기억하지 못할경우 변경내역을 확인 하기가 어렵습니다.
- 또한 만약 컴퓨터에만 보관을 했는데 하드디스크가 고장난다면 애써 작업한 소스코드를 다 잃어버리게 됩니다.

소스코드버전관리를 통해 문제가 해결되는 사례

- 소스코드버전관리는 이런 문제를 해결하기 위해 만들어졌습니다.



- 프로그래밍을 하고 중간중간 백업을 함으로써 언제든지 해당 시점으로 내 폴더를 복구 할 수 있습니다.
- 또한 외부의 서버에 소스코드를 보관해서 내 컴퓨터의 파일이 삭제 되더라도 다시 다운 받아서 작업을 계속할 수 있습니다.
- 또한 함께 작업하는 사람에게도 편리하게 모든 작업 내역을 공유할 수 있습니다.

소스코드버전관리란?

- 소스코드 버전관리를 이용하면
  - 소스코드의 변경된 내역을 묶어서 저장할 수 있습니다.
  - 원하는 시점으로 복구 할 수 있습니다.
  - 서버에 변경저장기록을 모두 저장 할 수 있습니다.
  - 다른 사람으로 부터 공유받은 올려놓은 소스코드를 서버로 부터 쉽게 내려 받을 수 있습니다.

Git이란?

- Git은 소스코드 및 파일의 변경내역을 저장하는 분산버전관리시스템 입니다.
- 리누스 토발즈에 의해 처음 만들어졌습니다.
- Github, Bitbucket, Gitlab 등의 Git기반의 버전관리호스팅 서비스들이 있습니다.
- 추천서적 : ProGit https://git-scm.com/book/ko/v2

Git 설치하기

- Ubuntu에서 설치하기
  - 명령어로 설치하기

```bash
$ sudo apt-get update
$ sudo apt-get install git
```

Bash

- Mac 설치하기
  - https://git-scm.com/ 접속
  - Download for Mac 버튼 클릭
  - 다운 받은 파일 실행
- Windows 설치하기
  - https://git-scm.com/ 접속
  - Download for Windows 버튼 클릭
  - 다운 받은 파일 실행

- Git 버전 확인

```
$ git --version
git version 2.19.1
```

Bash

- 초기 설정
  - 사용자정보 설정

```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

Bash

저장소 만들기

- 작업할 디렉토리 만들고 이동하기

```
$ mkdir hellogit
$ cd hellogit
```

Bash

- 현재 디렉토리를 Git 저장소로 만들기

```
$ git init
```

Bash

- 저장소에 파일을 추가하고 커밋하기

```
$ touch README
$ git add README
$ git commit -m "initial project version"
```

Bash

수정하고 저장소에 저장하기

- 파일의 상태 확인하기

```bash
$ git status
```

- git이 관리할 대상으로 파일 등록

```bash
$ git add README
```

- 버전 만들기(commit)

```bash
$ git commit -m "저장메세지를 입력해주세요"
```

- 파일 무시하기 - gitignore
  - .gitignore파일에 버전관리에서 제외할 파일을 추가한다.

```
# a comment - 이 줄은 무시한다.
# 확장자가 .a인 파일 무시
*.a
# 윗 줄에서 확장자가 .a인 파일은 무시하게 했지만 lib.a는 무시하지 않는다.
!lib.a
# 루트 디렉토리에 있는 TODO파일은 무시하고 subdir/TODO처럼 하위디렉토리에 있는 파일은 무시하지 않는다.
/TODO
# build/ 디렉토리에 있는 모든 파일은 무시한다.
build/
# `doc/notes.txt`같은 파일은 무시하고 doc/server/arch.txt같은 파일은 무시하지 않는다.
doc/*.txt
# `doc` 디렉토리 아래의 모든 .txt 파일을 무시한다.
doc/**/*.txt
```

- 변경사항 확인하기

```
$ git diff
```

Bash

- **커밋히스토리 조회하기**

```
$ git log
```

Bash

- Git Remote
  - 클론할 프로젝트 찾기
    - https://github.com 접속
    - jquery 검색하기
    - https://github.com/jquery/jquery 로 접속하기
    - Clone or download 버튼 클릭하기
    - 주소 복사하기
  - 새 작업 디렉토리만들고 이동하기

```
$ cd
$ mkdir clonetest
$ cd clonetest
```

Bash

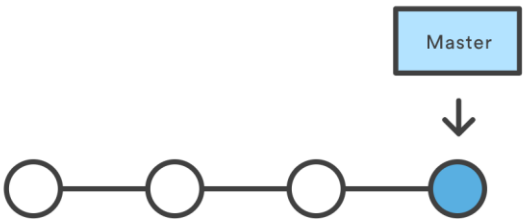
  - 클론하기

```
$ git clone https://github.com/jquery/jquery.git
$ cd jquery
$ git log
```

Bash

Branch

- branch란?
  - 개발을 하다 보면 코드를 여러 개로 복사해야 하는 일이 자주 생긴다. 코드를 통째로 복사하고 나서 원래 코드와는 상관없이 독립적으로 개발을 진행할 수 있는데, 이렇게 독립적으로 개발하는 것이 브랜치다.
  - Git의 브랜치는 특정 커밋을 가르키는 바로가기 같은 것 입니다.
  - 브랜치를 생성하면 현재 커밋을 가르키는 브랜치가 생성됩니다.
  - 특정 브랜치에서 작업중인데 새로운 커밋을 하면 브랜치는 그 새로운 커밋을 가르키게 됩니다.
  - 기본 브랜치는 master 브랜치 입니다. git저장소를 초기화 할때 자동으로 만들어 집니다.
  - Git은 'HEAD'라는 특수한 포인터가 있습니다. 이 포인터는 지금 작업중인 브랜치를 가르킵니다.

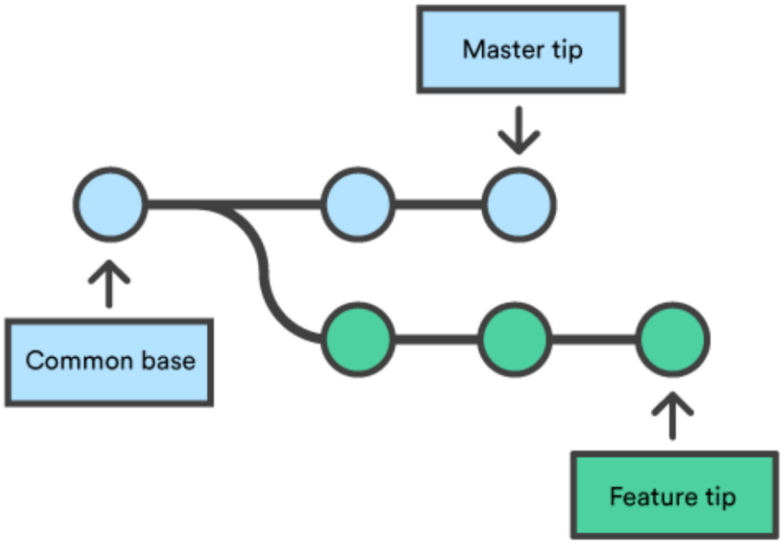


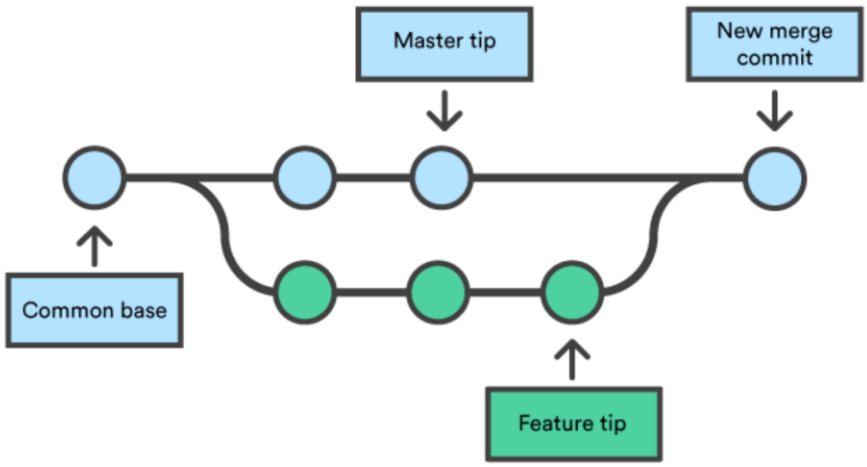
브랜치의 사용예

- '홍길동'은 새로운 기능을 개발 할 때마다 새로운 브랜치를 만들고 그안에서 작업을 하고 커밋을 합니다.
  기본 브랜치는 master 브랜치 입니다.
  회원가입기능을 만들려고 한다면 feature-usersignup 브랜치를 생성하고
  그 안에서 소스코드를 수정하고 커밋을 합니다.
  회원가입기능의 개발이 완료가 되면 그동안 커밋했던 feature-usersignup 브랜치의 내용들을 master 브랜치에 합칩니다.

  또 새로운 작업을 하려고 하면 master브랜치에서 다시 새로운 feature브랜치를 생성하고 작업을 합니다.

  회원가입 기능을 작업하고 있는데 급하게 버그를 수정해야 하는 경우가 발생합니다.
  그런경우 현재의 작업상태를 임시로 커밋해두고
  회원가입 기능 작업 시작이전 상태의 스냅샷으로 작업폴더를 변경할 수 있습니다.
  버그패치용 브랜치를 만들고 수정작업을 합니다.
  이후 작업이 완료되면 버그패치용 브랜치의 추가된 변경내역을 마스터 브랜치에 합칠 수 있습니다.





- 팀 '제주코드'는 '제주카페'라는 안드로이드앱을 런치한 후 지속적으로 업데이트 하고 있습니다.
  매주 월요일 오후마다 일주일동안의 수정내역을 반영한 앱을 앱스토어에 업데이트 합니다.
  팀은 두개의 브랜치를 가지고 있습니다. master와 develop 입니다.
  develop에는 평소 개발이 완료될때마다 지속적으로 커밋을 합니다.
  그리고 일주일에 한번씩 앱스토어에 배포하기전에 가장 최신의 develop브랜치의 내용을 master 브랜치에 반영시킨 후 그 소스코드를 가지고 앱을 빌드해서 배포를 합니다.

  따라서 월요일이 되기전의 master브랜치는 가장 최근에 앱스토어에 배포한 소스코드의 스냅샷 커밋을 항상 가르키고 있습니다.

- 현재 브랜치 목록과 현재 브랜치 확인

```bash
$ git branch
```

- branch 만들기
  브랜치를 새로 만들수 있다. testing 브랜치를 만들어 보자.

```bash
$ git branch testing
```

- checkout
  새로 만든 브랜치로 이동할 수 있다. testing 브랜치로 이동해보자.

```bash
$ git checkout testing
```

- 새로운 내용을 추가하고 커밋해 보자.

```bash
$ echo 'hello branch' >> branch.txt
$ git status
$ git add branch.txt
$ git commit -m "브랜치 테스트용 파일 추가"
```

- branch 병합
  이제 master브랜치에 testing브랜치에 추가된 내용을 병합해보자.

```bash
$ git checkout master
$ git log
$ git merge testing
$ git log
```

Github와 Bitbucket

- Github
  - Github가입하기
  - 프라이빗 리파지토리 생성하기
    - 리파지토리명 : hellogit
  - 만들었던 저장소 푸시 하기

```
$ cd hellogit
$ git status

$ git remote add origin https://github.com/beomjae/hellogit.git
$ git push -u origin master
```

Bash

  - 푸시한 프로젝트 페이지 살펴보기
  - github로 부터 클론하기

```
$ cd
$ mkdir hellogit2
$ cd hellogit2
$ git clone https://github.com/beomjae/hellogit.git .
```

Bash

  - 소스코드 수정 후 커밋과 푸시하기

```
$ echo "add more" >> a.txt
$ git status
$ git add .
$ git commit -m "추가작업내역입니다."
$ git push origin master
```

Bash

  - 푸시한 프로젝트 페이지 살펴보기

Atom을 이용한 Git 사용하기

- 소스코드 수정
- 변경내역 보기
- Stage 하기
- 커밋하기
- 푸시하기
- 체크아웃하기

Git에 관한 잡다한 이야기들

- 그래서 Git을 꼭 써야하나요?
  - Git은 이제 개발할 때 사실상 필수요소 입니다.
  - 백업을 해놓음으로서 부담없이 코딩할 수 있습니다.
    내가 코드를 잘못 수정해도 언제든지 쉽게 되돌아갈수 있는 마음이 생기기때문입니다.
- Git과 근무일지
  - git 커밋내역을 보면 누가, 언제, 어떤 작업을 했는지 적나라하게 기록되어있습니다.
    따라서 팀원간에 공동으로 작업을 한다면 다른 팀원이 어떤 개발을 했는지 묻지 않아도 커밋 로그만 봐도 알 수 있습니다.
    개발 안하고 농땡이 칠 수가 없어요.
- 커밋로그와 한글
  - 한국인으로만 이루어진 프로젝트라면 저는 커밋로그를 한글로 적습니다.
    영어로 문장을 길게 작성하기 어렵기 때문입니다.
    커밋로그는 말그대로 내가 작업한 내역을 적는 곳입니다.
    따라서 최대한 자세히 적으면 좋습니다.
    매번 커밋할때마다 영어로 작문하려면 스트레스 받고 그러다보면 자연스럽게 커밋로그가 짧아집니다.
    그러지 말고 그냥 한글로 자세하게 적는게 좋다고 생각합니다.
- Bitbuket과 Github
  - 비트버킷과 Slack의 연동
  - 비트버킷과 CI(젠킨스)의 연동
  - 간단한 이슈트래커를 제공하는 Bitbucket과 Github
  - 커밋로그에 이슈번호 적기

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

Day4

Python

1. 얕은물 강좌 스토리 미리보기
2. 인쇄용 PDF 파일 다운로드 링크
3. 환경설정
   (1) 라이캣의 탄생!
   (2) 생선을 잡아라
   (연습문제) 캣의 해골섬 출항!
   (3) 효율적으로 생선 잡기
   (4) 생선 회사를 운영하라
   (연습문제) 캣의 고객 관리
   (5) 라이캣의 EXIT
   (연습문제) 사자탈을 쓴 캣
   (스토리) 쏘아진 화살!
   (6) 파이와 썬의 알고리즘 7원석을 찾아서
   (연습문제) 캣의 모험 자금을 모아라!
   (연습문제) 파이와 썬이 남긴 단 하나의 단서
   (7) CheatSheet

PYTHON BOOTCAMP

· 얕은물 ·

이호준 김예원 김유진 차경림 김난화 김승미 고은 이승준 이진우

Weniv corp.

Contents

얕은물 강좌 스토리 미리보기 11

환경설정 14

1. Google Colab 간단한 사용법!
2. 상세한 사용법!

1편 라이캣의 탄생! 19

1. 캣의 일상
2. 캣의 정보 변경
3. 각 변수들의 타입을 알아보기
4. 각 변수들의 속성을 알아보기
   4.1 int의 속성 알아보기
   4.2 float의 속성 알아보기
   4.3 str의 속성 알아보기
   4.4 bool의 속성 알아보기
   4.5 list, tuple의 속성 알아보기
   4.6 dict의 속성 알아보기
   4.7 set의 속성 알아보기
   4.8 list, tuple, set, dict에 대해 더 알아 보기
5. 각 변수들을 형변환 해보기
6. 출력을 하는 여러가지 용법

2편 생선을 잡아라 64

1. 캣의 다짐
2. 캣의 생선팔기
3. 대입 연산자(할당 연산자)
4. 비교 연산자
5. 논리 연산자
6. 비트 연산자
7. 멤버 연산자
8. 식별 연산자
9. 연산자의 우선순위

캣의 해골섬 출항! 106

3편 효율적으로 생선 잡기 110

1. 캣의 고민
   1.1 들여쓰기
   1.2 함수의 사용
   1.3 print VS return
2. 함수를 사용한 캣의 계산
   2.1 상수(Constant)
3.전역변수 global
4. function 응용
   4.1 함수 안에 함수 만들기
5. 연산자 우선순위

4편 생선 회사를 운영하라 141

1. 캣의 생선회사
   1.1 캣의 회사 운영
   1.2 if 문의 기본 구조
2. if, else 문
3. if, elif, else

캣의 고객관리 163

5편 라이캣의 EXIT 171

1. 캣의 로봇(for)
   1.1 for문의 기본 구조
   1.2 for, else
   1.3 지능형 리스트(list comprehension)의 for
   1.4 다중인자 리스트 순회
   1.5 enumerate
   1.6 계산하는 로봇
2. 캣의 로봇(while)
   2.1 무한반복 while, break
   2.2 while, else
3. break
   3.1 break 문
4. continue, pass
5. else
6. 중첩 반복문

사자탈을 쓴 캣 240

스토리 : 쏘아진 화살! 244

6편 파이와 썬의 알고리즘 7원석을 찾아서 246

1. 라이캣의 모험
2. 클래스
   2.1 클래스 변수와 인스턴스 변수
   2.2 _init_함수
   2.3 상속
   2.4 다중 상속
   2.5 특별 메소드(magic method)
   2.6 캣의 준비물

캣의 모험 자금을 모아라! 270

파이와 썬이 남긴 단 하나의 단서 277

얕은물 강좌 스토리 미리보기

PYTHON BOOTCAMP

Python 부트캠프

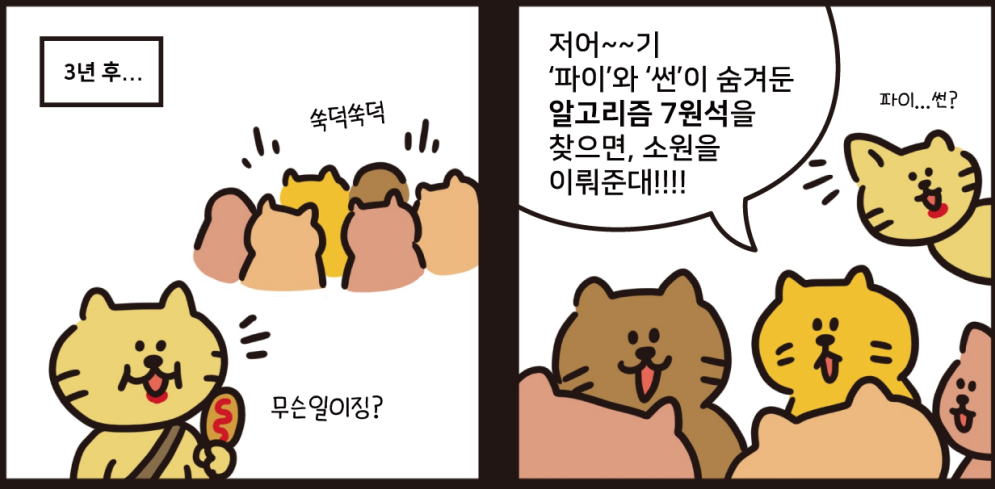
얕은물 강좌 스토리 미리보기

▶ 변수 : 라이캣의 탄생

▶ 연산 : 생선을 잡아라!

▶ 조건 : 생선회사를 운영하라

▶ 반복 : 라이캣의 EXIT



▶ 자료형 정리 : 파이와 썬의 알고리즘 7원석을 찾아서

스토리는 제코베 강의 '눈떠보니 코딩 테스트 전날'과 이어집니다.

환경설정

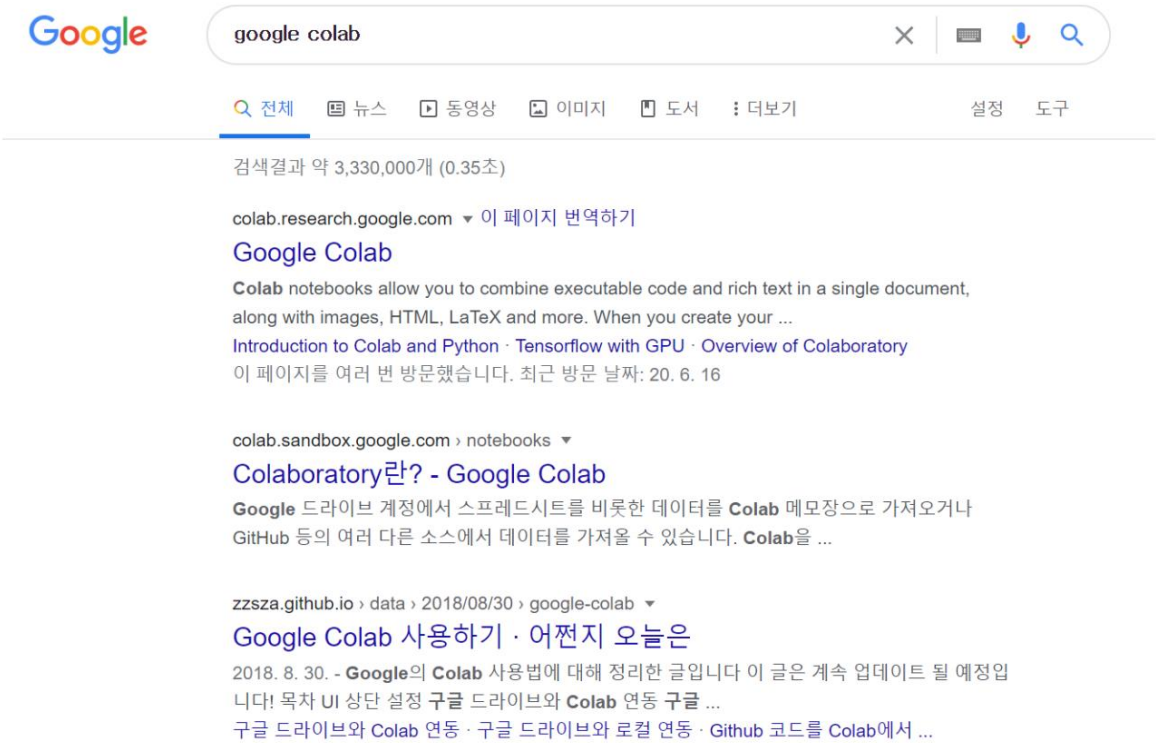
1. Google Colab 간단한 사용법!
2. 상세한 사용법!

Google 아이디만 있다면 사용할 수 있는 Google Colab!

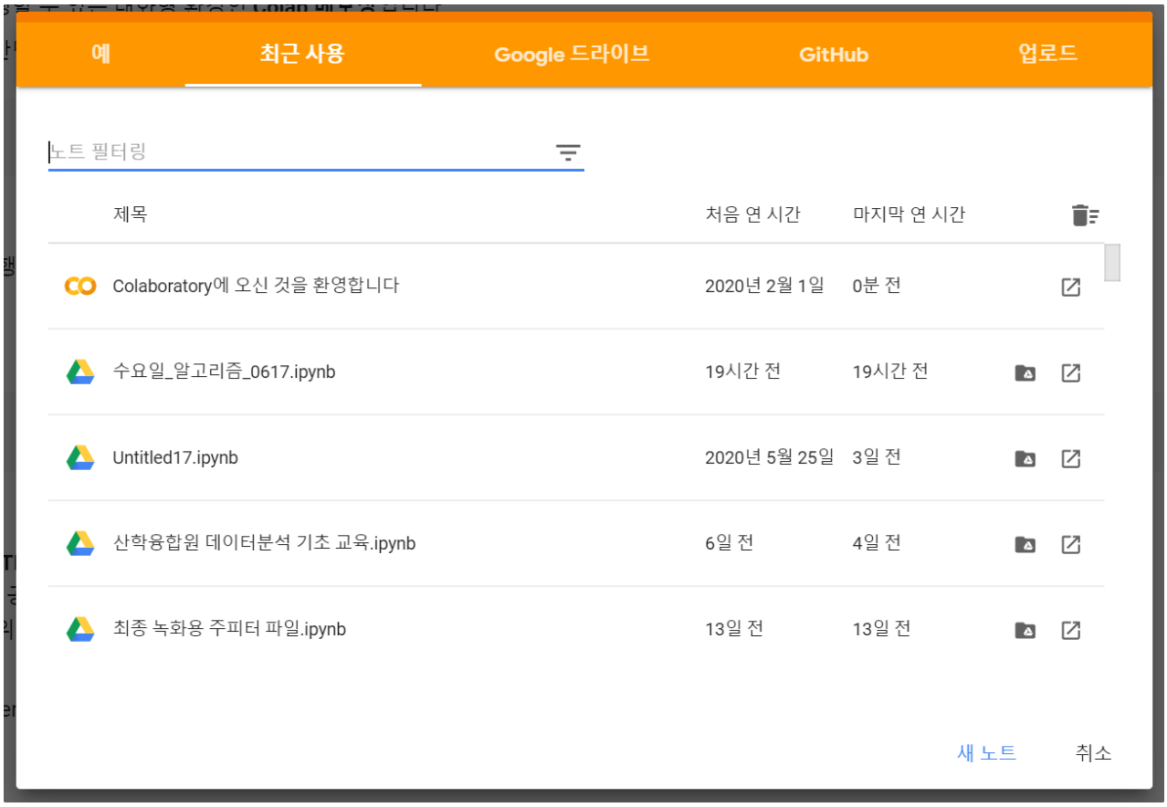
여러가지 불편하게 설치하는 작업 없이 여러분의 빠른 위니브 월드 진입을 도와줄 것입니다.

1. Google Colab 간단한 사용법!

1. google에 로그인을 해주시고, google에서 **google colab**이라고 검색을 합니다.



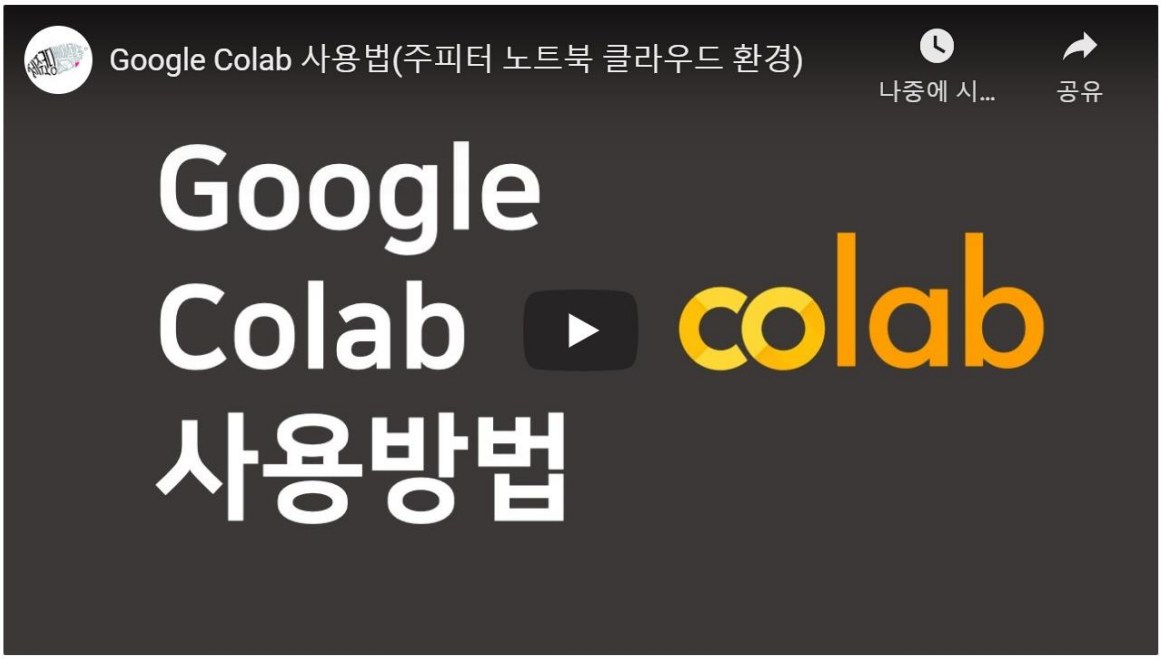
2. 다음과 같이 문서를 설정하는 창이 뜨는데요. 여기서 **새노트를 클릭**해주세요. 기존에 사용하신 파일이 있다면 업로드 해서 편집도 가능합니다. 그리고 **모든 파일은 자동으로 google drive에 저장**됩니다!



3. 아래와 같이 새 노트가 만들어집니다. 메뉴를 설명해 드리도록 하겠습니다!
    - 맨 위 제목을 설청하는 창이 있습니다. 누르시면 파일의 제목을 수정할 수 있어요. 오른쪽에 보시면 누군가와 파일을 공유할 수도 있답니다.
    - 파일, 수정, 보기, 삽입, 런타임, 도구, 도움말 등의 상태창이 떠요. 이건 지금 단계에서 필요하진 않으니 오른쪽에 위로 화살표가 그려진 것을 눌러 접어두세요. (상세 기능 설명은 아래 동영상을 참고하세요. 그러나, 초반에 너무 상세한 내용은 오히려 즐거운 코딩을 방해합니다.)
    - 그 아래 재생 버튼을 누르면 코드를 실행할 수 있어요! 단축키로는 `Ctrl + Enter` 입니다. 이 단축키는 많이 사용하니 알고 있으셔야 합니다!
    - 그 아래 코드를 삽입할 수도 있어요. 일반 텍스트를 삽입하여 설명을 추가할 수도 있죠. 일반 텍스트는 `마크다운` 을 활용하고 있어요. 코드를 실행시키고 바로 코드 셀을 추가하는 명령어는 `Alt + Enter` 입니다. 역시 많이 쓰는 단축키니 꼭 기억해두세요.

2. 상세한 사용법!

1. colab : 처음에 말씀드렸듯이, **초반에 접하는 상세한 내용은 즐거운 코딩을 방해**합니다. 가능하다면 이 챕터를 건너뛰시고, 어느정도 적응이 된 후 다시오세요. 😃



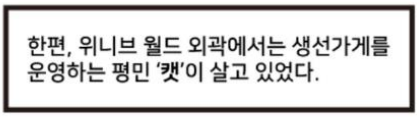
2. jupyter notebook : 로컬에서 jupyter notebook을 설치하시면 조금 더 쾌적한 환경에서 개발이 가능합니다. 그렇지만, 기억하세요. 구글 colab은 무려 Ram 할당량이 12G 입니다!

1편 라이캣의 탄생

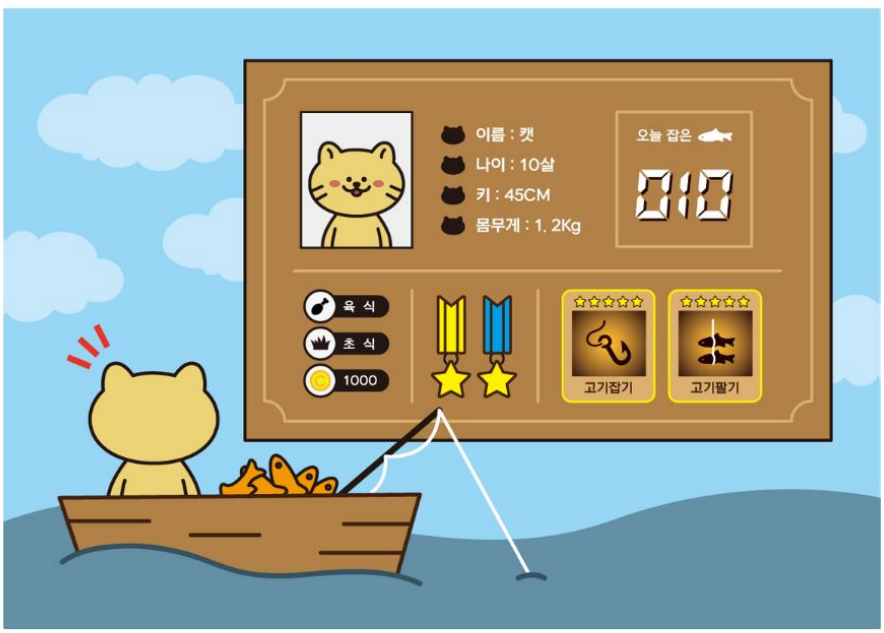
1. 캣의 일상
2. 캣의 정보 변경
3. 각 변수들의 타입을 알아보기
4. 각 변수들의 속성을 알아보기
    4.1 int의 속성 알아보기
    4.2 float의 속성 알아보기
    4.3 str의 속성 알아보기
    4.4 bool의 속성 알아보기
    4.5 list, tuple의 속성 알아보기
    4.6 dict의 속성 알아보기
    4.7 set의 속성 알아보기
    4.8 list, tuple, set, dict에 대해 더 알아 보기
5. 각 변수들을 형변환 해보기
6. 출력을 하는 여러가지 용법

PYTHON BOOTCAMP

1. 캣의 일상



생선 가게를 운영하는 평범한 소년 '캣'이 있습니다. 이 소년은 아픈 어머니를 대신하여 새벽이면 험한 바다에 나가 고기를 잡고, 오후가 되면 생선을 파는 생활을 반복하였어요.



캣에게는 비밀이 한가지 있었어요. 부모님께서 신신당부하여 남들에게 말하지 않았지만, 자신이 원할 때 자신의 능력치를 볼수 있는 신기하고 특별한 능력이 있었어요.

"나에겐 **고기잡기**와 **고기팔기** **스킬**이 있다냥, 매일매일 훈련해서 이 두개의 스킬 만큼은 최고의 스킬로 만들어보자냥!"

캣은 묵묵히, 매일 실력을 갈고닦았습니다.

```python
#[In]

# 라이캣의 개인정보 입력하기

이름 = '캣'
설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
나이 = 10
오늘_잡은_물고기 = '10'
키 = '45cm'
몸무게 = 1.2
육식 = True
초식 = True
돈 = 1000
훈장 = []
기술 = ['고기잡이', '고기팔기']

'''
라이캣의 개인정보입니다.
성장함에 따라 해당 값이 변합니다.

주의) 마음대로 성장시키지 마십시오.
'''
```

Python

이름, 나이, 오늘_잡은_물고기 등을 **변수**라고 합니다. **변수는 변할 수 있는 수**입니다. 또, 맨 위에 있는 `#`으로 되어 있는 부분과 `'''`로 감싸여져 있는 부분은 주석이라고 해요. Code의 양이 많거나 복잡하여 짧은 시간 내 이해하기 힘들 때 이해를 돕기 위해 주석이 필요합니다. 이 부분은 실행되지 않습니다. 코드를 잠시 보류하는 용도로도 사용하죠.

이렇게 하고 `Alt + Enter`를 실행해 보세요. 아래 셀이 추가되었죠? 따로 메시지는 뜨지 않았을 거에요. 무언가 실행시키지 않았기 때문입니다. 실행은 그 셀만 실행하는 `Ctrl + Enter`와 실행하고 셀 아래 셀을 추가하는 `Alt + Enter` 두가지가 있습니다.

💡 주석으로 변경을 원하는 코드에서 `Ctrl + /` 를 누르면 주석처리 됩니다. 여러줄도 가능해요.

2. 캣의 정보 변경

```python
#[In]

# 라이캣의 개인정보 변경하기

나이 = 11
이름, 나이
```

Python

이렇게 하고 `Alt + Enter`를 실행해 보세요. 나이에 11이 출력되었죠? 이처럼 변수는 변경할 수 있습니다. 그리고, 위에 있는 변수를 가져와 사용할 수도 있어요.

```python
#[In]

나이 = 나이 + 1
나이
```

Python

연산은 나중에 할 예정이지만, 이렇게 덧셈을 할 수도 있답니다. 그럼 아래와 같이 덧셈을 해볼게요.

```python
#[In]

# 오늘_잡은_물고기 = '10'
오늘_잡은_물고기 = 오늘_잡은_물고기 + 1
```

Python

이렇게 연산을 실행하게 되면 Error 문구가 뜨게 됩니다. 그 이유는 오늘 잡은 물고기는 문자열이기 때문인데요. 이처럼 파이썬과 같은 **고급프로그래밍 언어**에서는 그 형태를 구분하여 걸맞는 연산이 가능하도록 장치를 해두었답니다.

> 💡 Python이 무엇이고, 무엇을 할 수 있는지는 아래 영상을 참고해주세요. 고급 프로그래밍 언어가 어떤 뜻인지도 설명하고 있습니다. (책으로 보시는 분들은 YouTube에서 제주코딩베이스캠프를 검색해주세요.)

가능하다면, 1편인 Front-end도 시청해주세요. 😉

조금 더 알려드리자면, 변수에는 몇 가지 규칙이 있어요.

1. 변수 중간에 띄어쓰기를 하지 않습니다. 띄어쓰기를 명시하고 싶으실 때에는 언더스코어(`_`)를 사용하세요!
2. 첫 글자는 숫자나 특수문자를 쓰지 않습니다. 물론 언더스코어(`_`)제외입니다.
3. 첫 글자를 대문자로 쓰지 않습니다.(Class가 대문자로 쓰기 때문이지만, 대문자로 써도 실행됩니다.)
4. 예약어를 사용하지 않습니다. 예를 들어 뒤에 `print` 구문이 나오는데요. 이러한 함수명이나 구문은 변수명으로 사용하지 않습니다.
5. 변수명은 대소문자를 가립니다! `Apple`과 `apple`은 다른 변수가 됩니다.

```python
#[In]

# 오늘_잡은_물고기 = '10'
오늘_잡은_물고기 = int(오늘_잡은_물고기) + 1
오늘_잡은_물고기
```

Python

자, 위와 같이 변경하고 `Alt + Enter`를 눌러 실행시켜보세요! 연산이 되었죠? 그 이유는, `int`라는 형변환 함수가 문자열을 숫자로 바꾸어 주었기 때문입니다. 형변환은 이번 챕터 5장에서 자세히 살펴보겠습니다.

3. 각 변수들의 타입을 알아보기

이번에는 위에서 코드를 복사해와서 아래와 같이 작성 후 실행해보세요.

```python
#[In]

print(type(이름)) # '캣'
print(type(나이)) # 현재 12
print(type(오늘_잡은_물고기)) #현재 11
print(type(몸무게))
print(type(육식)) #True
print(type(기술)) #['고기잡이', '고기팔기']
```

Python

💡 `print`를 써도 되고 쓰지 않아도 되는 것인가요? 네, 주피터 노트북에서는 print를 쓰지 않아도 마지막에 있는 변수는 출력을 합니다.(Python 기본 에디터 사용시 Error!) 그러나 여러개를 출력할 때에는 꼭 `print`를 명시해주셔야 합니다.

---

4. 각 변수들의 속성을 알아보기

위에서 실행시킨 결과값은 아래 주석과 같이 나왔을거에요! 물론 더 많은 자료형이 있지만, 이 수업은 기초 수업이기 때문에 간단한 내용들만 살펴볼 것입니다.

```python
#[In]

print(type(이름)) # class 'str' - 문자열
print(type(나이)) # class 'int' - 정수
print(type(오늘_잡은_물고기)) # class 'int' - 정수
print(type(몸무게)) # class 'float'은 실수입니다.
print(type(육식)) # class 'bool' - 참거짓
print(type(기술)) # class 'list' - 배열
```

Python

각 변수들에는 속성이 있습니다. 예를 들어 스트링의 경우에는 덧셈을 하면 이어 붙인다던지 하는 속성이요.

```python
#[In]

x = '10'
y = 10
print(x + x)
print(y + y)
```

Python

위와 같이 실행을 시켰을 경우 `'10'+ '10'` 은 `'1010'` 이 되지만, `10 + 10` 은 `20` 이 됩니다. 이렇게 각 자료형마다 특징들이 있고, 이번 챕터에서는 이 특징들을 살펴볼 생각이에요. 그러나 걱정마세요. 너무 깊게 다루지는 않을 것입니다.

💡 Q : 깊게 다루지 않고 데이터 분석, 서비스 개발, IoT, 게임 개발 등을 할 수 있나요?
A : 아닙니다! 하지만, 초급자 분들은 문법에 매몰 되시면 안되요. 일단 한 두 서클을 돌아보시면, 자연스레 이해되는 문법들이 있고, 또 그렇게 한 번 성공한 결과물을 갖게되면 자신감도 생기거든요! 그러니 초급자 분들은 한 서클(결과물을 만드는 과정)을 도시는 것에 초점을 맞춰서 공부해주세요!

4.1 int의 속성 알아보기

int는 정수입니다! 정수가 가진 속성에 대해 알아보도록 하겠습니다. 우선 아래와 같이 실행시킨 다음에 말씀드리도록 하겠습니다.

```python
#[In]

나이 = 10
print(type(나이))
print(dir(나이))
```

Python

그럼 아래와 같이 무시무시한 길이의 코드가 나타납니다. 겁내지 마세요. 다 알 필요도 없을 뿐더러, 우리는 아주 얕고 넓게, 실용적인 부분만 배울 것이기 때문입니다.

일단 첫줄 먼저 설명해드릴게요. <class 'int'>, class는 맨 뒤에 챕터에서 다루게 되고, int는 우리가 앞에서 공부한 것처럼 **정수형**을 나타냅니다. 이처럼 type(**변수**)를 해보시면 **원하는 변수의 타입을 알아낼 수 있어요.**

```
#[Out]

<class 'int'>
['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__', '__delattr__', '__
dir__', '__divmod__', '__doc__', '__eq__', '__float__', '__floor__', '__floordiv__', '__f
ormat__', '__ge__', '__getattribute__', '__getnewargs__', '__gt__', '__hash__', '__index_
_', '__init__', '__init_subclass__', '__int__', '__invert__', '__le__', '__lshift__', '__
lt__', '__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__or__', '__pos__', '__pow_
_', '__radd__', '__rand__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__', '__
rfloordiv__', '__rlshift__', '__rmod__', '__rmul__', '__ror__', '__round__', '__rpow__',
'__rrshift__', '__rshift__', '__rsub__', '__rtruediv__', '__rxor__', '__setattr__', '__si
zeof__', '__str__', '__sub__', '__subclasshook__', '__truediv__', '__trunc__', '__xor__',
'bit_length', 'conjugate', 'denominator', 'from_bytes', 'imag', 'numerator', 'real', 'to_
bytes']
```

Python

여러분, 아래와 같이 정수와 실수를 더하면 얼마가 나올까요? 네, `20.1` 입니다. 그런데 다른 언어들은 이렇게 융통성이 있지 않아요. 만약 이 연산이 다른 언어였다면(JS는 더 융통성이 있으니 제외입니다.) 아래와 같이 말했을 거에요.

```
#[In]

#나이는 10이고, 몸무게는 10.1 입니다.

나이 + 몸무게
```

Python

> 실수로 더할꺼니? 그럼 실수로 바꿔줘야지. 정수로 더할꺼니? 그럼 뒤에 수에서 0.1을 포기해.

그런데 파이썬은 놀랍게도(!?), 이 연산이 가능합니다. 위 처럼 실행을 시켰다면 `20.1`의 출력 결과가 나올게요. 그럼 아래와 같은 코드는 어떨까요?

```
#[In]

10 + '10'
```

Python

이것을 실행시키면 또 아래와 같은 무시무시한 에러가 나옵니다. 앞으로 자주 만나게 될 에러명이기 때문에 한번 읽어보죠.

```
#[Out]

...
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Python

타입에러인데 정수형(int)와 문자열(str)의 덧셈을 허락하지 않는다고 합니다. 자, 이렇게 덧셈에 대해 어떻게 할 것인지에 대해 정의되어 있는 부분이, '__add__'입니다! 이부분은 class 챕터에서 더 배우게 되실거에요. 곱하기는 '__mul__', 비교는 '__eq__'랍니다.

아, 언더바(_)는 하나가 아니라 두개에요! 그래서 던더함수라고 부르기도 하죠. 매직메서드라고 부르는 것이 통상적입니다.

그렇다면, **언더바가 없는 것들의 정체**는 무엇일까요? 아래와 같은 코드를 실행해볼게요.

```
#[In]

나이 = 10
print(나이.bit_length())
print(bin(나이))

나이_90년후 = 100
print(나이_90년후.bit_length())
print(bin(나이_90년후))
```

Python

이것들은 `.`을 찍어 사용할 수 있어요. bit_length()만 사용해보았습니다. 이 **메서드**(지금은 점을 찍어서 사용할 수 있는 코드라고만 이해를 해주세요.)는 각 숫자를 bit로 변환한 자리숫자를 출력해준답니다! bit는 0과 1로 이루어진 세계입니다. **2진법**을 사용하죠.

자, 너무 많은 내용을 했어요. 모두 이해하실 필요 없습니다. 여기서 중요한 핵심은 `.`을 찍어 활용할 수 있는 코드를 `dir`로 볼 수 있다는 사실만 알고 넘어가도록 하겠습니다!

4.2 float의 속성 알아보기

float형은 실수입니다! 실수가 있으니 허수도 있을까요? 네, 물론입니다. 하지만, 우리 수업에서 그렇게 어려운 수학적인 내용은 다루지 않아요. 아래와 같이 실행해본 후 이 챕터는 넘어갑니다.

```
#[In]

# 몸무게는 10.1kg 입니다!
print(type(몸무게))
print(dir(몸무게))
```

Python

4.3 str의 속성 알아보기

`str` 자료형은 스트링이라 읽습니다. 이 자료형은 작은따옴표(‘ ’)나 큰따옴표(“ ”) 또는 삼중따옴표(“”” “””, ‘’’ ’’’)로 둘러싸서 나타냅니다. 삼중따옴표를 사용할 경우에는 줄단위의 문자열을 나타낼 수 있습니다. 이러한 문자열은 **시퀀스 자료형**에 해당됩니다.

Sequence, 순서가 있는 자료형이라는 뜻으로 각 요소들은 해당하는 **Index 값을 갖습니다**. 파이썬에서 **가장 많이 사용되는 형태의 자료형**이며, 시퀀스 자료형이 가지는 특성을 통하여 인덱싱, 슬라이싱, 반복 등 여러 연산으로 그 활용 범위가 넓습니다. 파이썬에서는 **문자열, 리스트** 등이 시퀀스 자료형에 해당됩니다.

```python
#[In]

# 설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
# 기술 = ['고기잡이', '고기팔기']

print(설명[0])
print(설명 [1])
print(설명 [2])
print(기술 [0])
print(기술 [1])
print(기술 [0][0])
print(기술 [0][1])
```

Python

```
#[Out]

위
니
브
고기잡이
고기팔기
고
기
```

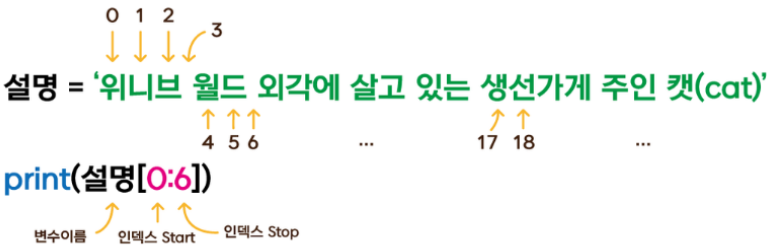
Python

여기서 `설명[0]` 안에 들어가는 0이라는 숫자를 **index**라고 부릅니다! 그리고 이렇게 index를 활용하여 특정 값을 호출하는 방식을 **indexing**이라고 하죠!

또, 이 index를 활용하여 문자열에서 **원하는 부분만을 추출**해낼 수 있습니다.

```python
#[In]

설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
print(설명[0:6])
```

Python



인덱스가 start 인 지점에서 end 미만인 지점까지 추출합니다. 여기서 start 를 생략했을 경우 문자열의 시작 지점, end 를 생략했을 경우 문자열의 끝지점이 설정됩니다.

```python
#[In]

캣의_생년월일 = "2220.02.22"
생년 = birth[:4]
월 = birth[5:7]
일 = birth[8:]
print("태어난 연도 : ", 생년)
print("태어난 월 : ", 월)
print("태어난 일 : ", 일)
```

Python

```
#[Out]

태어난 연도 : 2220
태어난 월 : 02
태어난 일 : 22
```

Python

`변수명[start:stop:step]`와 같은 형식으로 사용도 가능합니다. 이 경우 순회가능한 객체에 `start index`부터 `stop index` 바로 전만큼의 범위에서 k만큼 건너뛰게 됩니다.

```python
#[In]

숫자 = '123456789101112'
print(숫자[::-1])
print(숫자[1:7:2])
```

Python

```
#[Out]

'211101987654321'
'246'
```

4.4 bool의 속성 알아보기

`bool` 자료형은 부울형, 불리언 자료형이라 읽습니다. 이 자료형은 `True` 와 `False` 2가지 타입밖에 없어요.

💡 javascript에서는 `True` 를 소문자 형태인 `true` 로 표시하고 있어요! 이런 차이점에 대해서 알고 있으시면 좋답니다.

```python
#[In]

육식 = True
초식 = True
print(type(육식))
print(dir(초식))
```

Python

```
#[Out]

<class 'bool'>
['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__', '__delattr__', '__
dir__', '__divmod__', '__doc__', '__eq__', '__float__', '__floor__', '__floordiv__', '__f
ormat__', '__ge__', '__getattribute__', '__getnewargs__', '__gt__', '__hash__', '__index_
_', '__init__', '__init_subclass__', '__int__', '__invert__', '__le__', '__lshift__', '__
lt__', '__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__or__', '__pos__', '__pow_
_', '__radd__', '__rand__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__', '__
rfloordiv__', '__rlshift__', '__rmod__', '__rmul__', '__ror__', '__round__', '__rpow__',
'__rrshift__', '__rshift__', '__rsub__', '__rtruediv__', '__rxor__', '__setattr__', '__si
zeof__', '__str__', '__sub__', '__subclasshook__', '__truediv__', '__trunc__', '__xor__',
'bit_length', 'conjugate', 'denominator', 'from_bytes', 'imag', 'numerator', 'real', 'to_
bytes']
```

Python

4.5 list, tuple의 속성 알아보기

`list`는 두 개 이상의 값을 저장하고 싶을 때 사용하는 자료형입니다. `str`처럼 순서가 있는 시퀀스 자료형이며, 각 원소들은 변경이 가능합니다. 리스트의 기본 형태는 아래와 같습니다.

```python
#[In]

훈장 = []
기술 = ['고기잡이', '고기팔기']
print(type(기술))
print(dir(기술))
```

Python

```
#[Out]

<class 'list'>
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc_
_', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__has
h__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__l
en__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '_
_subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop',
'remove', 'reverse', 'sort']
```

Python

여기서 대괄호(`[]`)를 사용하지 않고 소괄호(`()`)를 사용하면 list가 아니라, 자료의 값이 변경 불가능한 `tuple`이 됩니다. tuple은 순서가 있고, 소괄호를 사용하며, 값을 변경할 수 없습니다. 얕은 물에서는 튜플을 상세하게 다루지 않으니 가볍게 훑고 넘어가주세요.

```python
#[In]

잡은물고기_튜플 = ('광어', '고등어', '오징어', '오징어', '광어', '광어', '고등어', '고등어', '백
상아리', '금붕어')
```

Python

앞서 말씀드린 것처럼 `list`는 순서가 있는 시퀀스형 자료형이기 때문에 아래와 같이 `str`처럼 **indexing, slicing이 가능**합니다.

```python
#[In]

잡은물고기 = ['광어', '고등어', '오징어', '오징어', '광어', '광어', '고등어', '고등어', '백상아리', '금붕어']
```

Python

```python
#[In]

print(잡은물고기[0])
print(잡은물고기[0:3])
print(잡은물고기[0:7:2])
```

Python

```python
#[Out]

광어
['광어', '고등어', '오징어']
['광어', '오징어', '광어', '고등어']
```

Python

그런데 마지막에 잡은 물고기가 금붕어죠? 잘못 표기한 것은 아래와 같이 바꿀 수 있답니다.

```python
#[In]

잡은물고기[-1] = '백상아리' # -1 인덱스는 마지막에 있는 값으로, 아래와 같은 의미를 지닙니다.
잡은물고기[9] = '백상아리'

잡은물고기
```

Python

```python
#[Out]

['광어', '고등어', '오징어', '오징어', '광어', '광어', '고등어', '고등어', '백상아리', '백상아리']
```

Python

여기서 광어는 얼마나 잡았는지, 고등어는 얼마나 잡았는지 알고 싶으면 어떻게 할까요? 여기서 앞서 말씀드린 **메서드**를 사용할 수 있답니다. 아래 코드를 실행해 보세요. **메서드 정리**는 **다른 강의**해서 해 드릴거에요. 여기는 얕은물 강좌이니, 어떻게 사용하는지 가볍게 살펴봅시다.

```
#[In]

잡은물고기.count('백상아리')
```

Python

```
#[Out]

2
```

Python

대단하군요. 고양이가 백상아리를 잡다니요. 그럼 훈장에 '**백상아리를 잡은 고양이**'를 추가해볼게요. 이 역시 메서드를 사용합니다.

```
#[In]

훈장.append('백상아리를 잡은 고양이')
훈장
```

Python

```
#[Out]

'백상아리를 잡은 고양이'
```

Python

4.6 dict의 속성 알아보기

그런데 이렇게 물고기를 저장하는 것은 비효율 적이겠죠? 우리는 '**백상아리!**' 하면 '**2마리!**' 이렇게 바로 알려주었으면 좋겠는데요. 이것이 가능한 자료형이 Dictionary입니다.

```
# Dictionary의 구조

dic = { 'key' : 'value' }
```

Python

자, 그러면 위에 리스트로 저장되어 있던 물고기를 딕셔너리로 정리해봅시다.

```
#[In]

잡은물고기_딕셔너리 = {'광어' : 3, '고등어' : 2, '오징어' : 2, '백상아리' : 2}
```

Python

어떤가요? 보다 간편해졌죠? 우리는 앞으로 아래와 같이 key 값으로 value를 호출할 수 있습니다.

```
#[In]

잡은물고기_딕셔너리['광어']
```

Python

```
#[Out]

2
```

Python

값을 추가하고 싶을 때에는 아래와 같은 방법을 사용할 수 있습니다.

```
#[In]

잡은물고기_딕셔너리['고래'] = 1
잡은물고기_딕셔너리
```

Python

```
#[Out]

{'광어' : 3, '고등어' : 2, '오징어' : 2, '백상아리' : 2, '고래' : 1}
```

Python

값을 지울 때에는 아래와 같은 방법을 사용합니다.

```python
#[In]

del 잡은물고기_딕셔너리['고래']
#del 키워드는 다른 자료형에서도 데이터를 지울 때 사용할 수 있습니다.

잡은물고기_딕셔너리
```

Python

```python
#[Out]

{'광어' : 3, '고등어' : 2, '오징어' : 2, '백상아리' : 2}
```

Python

Key값만 따로 알고 싶을 때와 Value값만 따로 알고 싶을 때에는 아래와 같은 방법을 사용합니다.

```python
#[In]

print(잡은물고기_딕셔너리.keys())
print(잡은물고기_딕셔너리.values())
print(잡은물고기_딕셔너리.items())
```

Python

```python
#[Out]

dict_keys(['광어', '고등어', '오징어', '백상아리'])
dict_values([3, 2, 2, 2])
dict_items([('광어', 3), ('고등어', 2), ('오징어', 2), ('백상아리', 2)])
```

Python

4.7 set의 속성 알아보기

set은 집합이에요. 중복을 허락하지 않죠! 혹시 차집합, 합집합, 교집합 아시나요? 그 집합입니다! 자, 그럼 어떻게 사용할까요?

```python
#[In]

잡은물고기_집합 = set(잡은물고기)
잡은물고기_집합
```

Python

```python
#[Out]

잡은물고기 = {'광어', '고등어', '오징어', '백상아리'}
```

Python

어떤가요? 중복이 사라졌습니다! 이제 잡은 물고기의 종류를 한 번에 볼 수 있죠!

4.8 list, tuple, set, dict에 대해 더 알아보기

자료형끼리의 사칙연산과 메서드는 한 번 정리할 필요가 있어요. 더 공부하기를 원하신다면, 아래 영상을 참고하여 각 자료형 옆에 있는 메모 노트에 메모해 주세요. 하지만 기억하세요. 먼저 빠르게 훑어 간단한 웹이나 앱, 게임이나 IoT와 같은 결과물을 만들고, 나중에 좀 더 깊게 파봐도 늦지 않습니다. 너무 많은 내용을 한꺼번에 담으려 하지 마세요.

Youtube '제주코딩베이스캠프'의 영상입니다😉

5. 각 변수들을 형 변환 해보기

자, 형 변환에 대해 좀 더 깊게 알아봅시다.

```python
#[In]

print(type(int(3.5)))
print(int(3.5))
print(type(float(3)))
print(float(3))
print(type(str(3)))
print(str(3))
```

Python

```
#[Out]

<class 'int'>
3
<class 'float'>
3.0
<class 'str'>
3
```

Python

앞서 말씀드린 것처럼 기존에 자료형에서 다른 자료형으로 바꾸는 것을 형 변환이라고 합니다. 아래처럼 `input` 함수를 이용하면 숫자나 문자를 입력받을 수 있는데요. 둘 다 `str`로 변수를 받기 때문에 주의해서 사용을 해야 합니다.

```python
#[In]

x = input('좋아하는 숫자를 입력하세요 :')
y = input('더할 숫자를 입력하세요 :')
print(x + y)
print(type(x))
print(type(y))
```

Python

```
#[Out]

좋아하는 숫자를 입력하세요 : 123
더할 숫자를 입력하세요 : 123
123123
<class 'str'>
<class 'str'>
```

Python

파이썬에서의 형 변환은 아주 많은 방법들이 있지만 **내장함수(built in-functions, 이 키워드는 매우 중요하니 꼭 암기해두세요.)**를 이용해서 아주 쉽게 할 수 있습니다. 어떤식으로 쓰이는지 간단한 예시를 통해 같이 한번 살펴봅시다.

형변환

✨ int	다른 자료형을 정수형으로 변환
✨ str	다른 자료형을 문자열로 변환
✨ float	다른 자료형을 실수형으로 변환
✨ list	다른 자료형을 리스트로 변환
✨ tuple	다른 자료형을 튜플형으로 변환
✨ set	다른 자료형을 집합 자료형으로 변환
✨ dict	다른 자료형을 사전형으로 변환
+ New	

- **int로 형변환**

```
#[In]

오늘잡은_물고기_수 = '371'
print(type(오늘잡은_물고기_수))
print(type(int(오늘잡은_물고기_수)))
print(int(오늘잡은_물고기_수)
```

Python

```
#[Out]

<class 'str'>
<class 'int'>
371
```

Python

- string으로 형변환

```
#[In]

오늘잡은_물고기_수 = 371
print(type(오늘잡은_물고기_수))
print(type(str(오늘잡은_물고기_수)))
print(str(오늘잡은_물고기_수))
```

Python

```
#[Out]

<class 'int'>
<class 'str'>
'371'
```

Python

- bool형으로 형변환

```
#[In]

print("bool('test') : ", bool('test!!'))
print("bool(1) : ", bool(1))
print("bool(0) : ", bool(0))
print("bool(-1) : ", bool(-1))
print("bool(' ') : ", bool(' '))
print("bool('') : ", bool(''))
print("bool(None) : ", bool(None))
```

Python

```
#[Out]

bool('test') : True
bool(1) : True
bool(0) : False
bool(-1) : True
bool(' ') : True
bool('') : False
bool(None) : False
```

Python

bool()함수는 **인자값**(아규먼트, argument, parameter와는 차이가 있으니 함수에서 정리해드리도록 하겠습니다.)을 Boolean 자료형으로 형변환하게 됩니다. 부울 값은 True와 False로 나뉩니다.

부울 값은 **직접 입력된 값일 수도 있고 부울 연산에 의해 나온 결과값**일 수도 있습니다.
예를 들어 x = 10 일 때 x > 100은 False이죠. 또한 이미 Python 내에서 규정한 부울 값일 수도 있습니다. 예를 들어 0은 False, 0을 제외한 다른 숫자는 True입니다.

💡 list, tuple, dict, set은 형변환을 어떻게 할까요? 옆 노트에 정리해보세요.

---

6. 출력을 하는 여러가지 용법

`print` 내장함수(built-in functions, 빌트인펑션)를 사용하여 출력하는 방법에는 여러가지 방법이 있습니다. 물론 우리가 사용하는 `colab`이나 `jupyter notebook`은 마지막 라인에 한하여 `print`를 쓰지 않아도 출력합니다.

```
print('1. 제 이름은 ', 이름, '입니다. 제 나이는 ', 나이, '입니다')
print(f'2. 제 이름은 {이름}입니다. 제 나이는 {나이}입니다.')
print('3. 제 이름은 {}입니다. 제 나이는 {}입니다.'.format(이름, 나이))
print('4. 제 이름은 %s입니다. 제 나이는 %d입니다.'%(이름, 나이))
```

Python

여기서 비교적 최근에 나온 2번 문법이 자주 사용됩니다. 4번은 잘 사용되지 않으니 참고바랍니다.

1. `,(콤마)` 로 연결하는 방법입니다. 콤마의 단위는 오브젝트입니다. `이름`, `나이` 모두 오브젝트의 한 단위이고 `'1. 제 이름은'`, `'입니다. 제 나이는'` 등의 문자열도 모두 오브젝트입니다. 앞에서는 변수라고 배웠죠? 변수에 넣을 수 있는 모든 값 또는 변수도 오브젝트입니다. 예를 들어 아래와 같이 변수나 그 값을 직접 넣는 것은 같은 값을 출력합니다.

```python
힘 = 12
print('제 힘은 ', 힘, '입니다.')
print('제 힘은 ', 12, '입니다.')
```

Python

2. python 3.6 version 이상에서는 f-string 용법을 지원합니다. 사용하는 방식은 `{}(중괄호)`에 변수 이름을 넣으시면 됩니다.
3. format을 이용한 문자열 포매팅 방식입니다. 앞서 `dir('문자열')`에서 던더함수 뒤에 있는 메서드를 확인해보시면 `format` 메서드가 있습니다. `{}(중괄호)`로 변수의 자리를 비워놓고 `'문자열'.format()` 괄호 안에 넣고 싶은 오브젝트를 넣어주는 방법입니다. 역시 변수는 콤마로 구별해 넣습니다.
4. 잘 사용하지 않는 방법입니다. 포맷코드를 이용한 문자열 포매팅입니다. 넣고 싶은 오브젝트는 % 뒤에 넣습니다. 넣을 값에 맞춰서 알맞은 자료형에 대한 포맷코드를 선택해야 합니다. 아래 코드표는 참고삼아 훑고 넘어가세요.

포멧 코드의 종류

%c	문자
%s	문자열
%d	정수
%f	실수
%b	2진수
%o	8진수
%x	16진수

+ New

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

2편 생선을 잡아라

1. 캣의 다짐
2. 캣의 생선팔기
3. 대입 연산자(할당 연산자)
4. 비교 연산자
5. 논리 연산자
6. 비트 연산자
7. 멤버 연산자
8. 식별 연산자
9. 연산자의 우선순위

1. 캣의 다짐

어느날 생선가게를 운영하며 열심히 살아가던 캣에게 슬픔이 찾아왔습니다. 아프던 캣의 어머니의 병세가 더 위독해진 것이었어요.

당장 병원에 가야했지만 '위니브 월드'에서의 병원은 왕족인 사자와 부자에게만 허락된 공간이었습니다. 평민이었던 캣의 어머니는 병원에 가지 못하였고 캣에게 생선가게를 부탁하였습니다.

"생선가게를 부탁하마..."

"엄마.."

캣은 다짐했습니다.

> 평민도 이용할 수 있는 병원을 세울꺼다냥!!

캣은 병원을 세우기 위해 자신의 스킬을 활용하여 더 열심히 생선을 잡고 팔았습니다.

- **라이캣의 현재 상태창!**

```python
#[In]

이름 = '캣'
설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
나이 = 13
오늘_잡은_물고기 = '10'
키 = '45cm'
몸무게 = 1.2
육식 = True
초식 = True
돈 = 5000
훈장 = ['백상아리를 잡은 고양이']
기술 = ['고기잡이', '고기팔기']
```

Python

캣은 장사를 할 때 캣만의 규칙이 있었습니다. 잡은 물고기를 당일 다 팔아야 하고 다음날 매출은 오늘 매출의 2배로 목표를 잡습니다.

```python
#[In]

# 캣의 규칙
오늘_잡은_물고기 = 0
오늘_판_물고기 = 0
매출 = 0

# 물고기를 10마리 잡았습니다.
# 모든 물고기는 100원입니다.

오늘_잡은_물고기 = 오늘_잡은_물고기 + 10
오늘_판_물고기 = 오늘_잡은_물고기
재고 = 오늘_잡은_물고기 - 오늘_판_물고기
매출 = 100*10

#f-string 출력 방법을 사용하셔도 좋습니다.
print('나는 오늘', 오늘_잡은_물고기, '마리를 잡았고', 오늘_판_물고기,'마리를 팔았다냥')
print('재고는', 재고,'마리다냥!')
print('오늘', 매출,'원을 벌었으니 내일은', 매출*2,'원을 벌테다!')
```

Python

`오늘_잡은_물고기 + 10` , `오늘_판_물고기 = 오늘_잡은_물고기` , `오늘_잡은_물고기 - 오늘_판_물고기` , `100*10` 에서 볼 수 있는 +, =, -, * 와 같이 값을 계산하거나 대입하는 것을 연산자라고 합니다.

연산자의 종류로는 **산술 연산자**, **대입 연산자(할당연산자)**, **비교 연산자**, **논리 연산자**, **비트 연산자**, **멤버 연산자**, **식별 연산자**가 있습니다.

💡 종류가 많기 때문에 어려움을 느끼실 수 있지만 이 연산자들을 한번에 알려고 하는 것은 큰 의미가 없습니다. 필요할 때 사용해 보면서 익히는 것이 중요합니다.

---

2. 캣의 생선 팔기

캣은 매출을 올리기 위해 모든 물고기를 같은 가격이 아닌 등급에 따라 다르게 책정하기로 합니다.

> 물고기 등급에 따라 가격을 다르게 팔아볼까냥?

```python
#[In]

# A등급 : 1000원, B등급 : 500원, C등급 : 100원
# 오늘 잡은 물고기 : A등급 5마리, B등급 7마리, C등급 10마리

A등급 = 5
B등급 = 7
C등급 = 10
매출 = 0
```

Python

등급에 따라 가격을 책정하여 장사를 시작한 캣의 생선 가게에 손님이 찾아왔습니다.

```python
#[In]

# 손님의 주문
# A등급 2마리, B등급 3마리 주세요.
# 받은 돈은 4000원 입니다.

A등급 = A등급 - 2
B등급 = B등급 - 3

합계 = 1000*2 + 500*3
받은_돈 = 4000

print(f'감사합니다. 여기 거스름 돈 {받은_돈 - 합계}원 입니다.')

재고 = A등급 + B등급 + C등급
매출 = 매출 + 합계

print(f'현재 매출 : {매출}, 재고 : {재고}')
```

Python

여기서 `Alt + Enter` 를 눌러주면 아래와 같이 출력됩니다.

> 💡 물고기의 등급과 가격을 dict로 구현해보시고, 계산해보세요!

```
#[Out]

감사합니다. 여기 거스름돈 500 원입니다.
현재 매출 : 3500 재고 : 17
```

Python

오늘도 잡은 물고기를 다 팔아 재고를 0으로 장사를 마무리 하는 캣. 하지만 위니브 월드에서는 당일 매출의 4분의 1을 세금으로 납부해야 한다고 합니다.

> 에휴 오늘도 세금을 납부해야겠구냥... 오늘은 또 얼마를 내야하냥?

```
#[In]

# 세금 납부하기
총매출 = 10000
세금 = 총매출/4
순이익 = 총매출-세금

print(총매출, 세금, 순이익)
# 10000 2500.0 7500.0
```

Python

+, -, *, / 와 같은 연산자를 **산술연산자**라고 합니다. 산술연산자는 가장 많이 접할 수 있고, 자주 사용되는 연산자입니다. 사칙연산과 더불어 제곱, 나머지 연산자 등이 있습니다.

산술 연산자

Aa 기호	이름	설명	예제
+	덧셈	연산자의 왼쪽과 오른쪽 값을 더합니다.	a + b
-	뺄셈	연산자의 왼쪽 값에서 오른쪽 값을 뺍니다.	a - b
*	곱셈	연산자의 왼쪽 값과 오른쪽 값을 곱합니다.	a * b
**	제곱	연산자의 왼쪽 값에서 오른쪽 값만큼 제곱합니다.	a ** b
/	나눗셈	연산자의 왼쪽 값을 오른쪽 값으로 나눕니다.	a / b
//	몫	연산자의 왼쪽 값을 오른쪽 값으로 나눈 몫을 구합니다. (따라서 결과는 정수)	a // b
%	나머지	연산자의 왼쪽 값을 오른쪽 값으로 나눈 나머지를 구합니다.	a % b

COUNT 7

```python
#[In]

a = 5
b = 2

print('a + b = ', a + b)    # 7
print('a - b = ', a - b)    # 3
print('a * b = ', a * b)    # 10
print('a / b = ', a / b)    # 2.5
print('a ** b = ', a ** b)  # 25
print('a // b = ', a // b)  # 2
print('a % b = ', a % b)    # 1
```

Python

💡 다른 프로그래밍 언어에서 연산자를 이미 학습하신 분들은 눈치채셨을 수도 있으시겠지만, 파이썬에는 전/후위 연산자(++, --)가 존재하지 않습니다!

자, 성실히 세금을 납부했으니 훈장을 부여하도록 합시다.

```python
#[In]

훈장.append('성실한 납세자')
```

Python

3. 대입 연산자(할당 연산자)

대입 연산자는 **산술 연산의 코드를 축약**해서 사용할 수 있도록 지원하는 기능입니다.

피연산자를 한번만 사용하기 때문에 대입 연산자를 잘 활용한다면 **코드를 좀더 간결하게 사용할 수 있다**는 장점이 있습니다.

위의 예제에서 사용했던 코드를 가져와서 알아봅시다.

```
#[In]

# 1번 예제
오늘_잡은_물고기 = 오늘_잡은_물고기 + 10
오늘_판_물고기 = 오늘_잡은_물고기
```

Python

```
#[In]

# 대입 연산자
오늘_잡은_물고기 += 10
오늘_판_물고기 = 오늘_잡은_물고기
```

Python

`오늘_판_물고기 = 오늘_잡은_물고기` 는 대입 연산자로 오른쪽 값을 왼쪽 변수에 할당하였습니다.
`오늘_잡은_물고기 = 오늘_잡은_물고기 + 10` 는 덧셈 대입을 사용하여 `오늘_잡은_물고기 += 10` 로 축약하여 사용할 수 있고 연산 결과는 동일합니다.

다른 산술 연산자도 모두 대입 연산자로 축약하여 사용할 수 있습니다.

```
#[In]

a = 10
b = 2

a += b   # 12
a -= b   # 10
a *= b   # 20
a /= b   # 10.0
a **= b  # 100.0
a //= b  # 50.0
a %= b   # 0.0
```

Python

대입 연산자(할당 연산자)

Aa 기호	이름	설명	예제	동일
=	대입	연산자의 오른쪽 값을 왼쪽 변수에 할당합니다.	a = b	a = b
+=	덧셈 대입	연산자의 왼쪽 변수의 값과 오른쪽 값을 더한 결과를 왼쪽 변수에 할당합니다.	a += b	a = a + b
-=	뺄셈 대입	연산자의 왼쪽 변수의 값에서 오른쪽 값을 뺀 결과를 왼쪽 변수에 할당합니다.	a -= b	a = a - b
*=	곱셈 대입	연산자의 왼쪽 변수의 값과 오른쪽 값을 곱한 결과를 왼쪽 변수에 할당합니다.	a *= b	a = a * b
**=	제곱 대입	연산자의 왼쪽 변수의 값에서 오른쪽 값만큼 제곱한 결과를 왼쪽 변수에 할당합니다.	a **= b	a = a ** b
/=	나눗셈 대입	연산자의 오른쪽 변수의 값을 오른쪽 값만큼 나눈 결과를 왼쪽 변수에 할당합니다.	a /= b	a = a / b
//=	몫 대입	연산자의 왼쪽 변수의 값을 오른쪽 값만큼 나눈 몫을 왼쪽 변수에 할당합니다.	a //= b	a = a // b
%=	나머지 연산 대입	연산자의 왼쪽 변수의 값을 오른쪽 값만큼 나눈 나머지를 왼쪽 변수에 할당합니다.	a %= b	a = a % b

COUNT 8

익숙하지 않고, 혼란의 여지가 있을 경우에는 산술 연산자인 형태로 쓰셔도 됩니다. 실질적으로 연산의 결과값은 동일하기 때문에 상관 없습니다.

하지만 간결한 코드를 중요하게 여겨지는 순간이 온다면 그 때 대입 연산자로 활용하는 훈련을 조금씩 하면 됩니다. 물론 함께 일하는 팀의 컨벤션 등이 정해진다면 그것에 따르는 것이 좋습니다.

4. 비교 연산자

비교 연산자는 왼쪽 피연산자와 오른쪽 피연산자의 값을 비교하여 boolean 값(True, False)으로 반환해주는 역할을 합니다. 간단히 말하면 **값을 비교하여 참과 거짓을 구분합니다.**

오늘 잡은 물고기와 판매한 물고기의 수를 비교해 봅시다.

```python
#[In]

오늘_잡은_물고기 = 10
오늘_판_물고기 = 5

print(오늘_잡은_물고기 > 오늘_판_물고기)  # True
print(오늘_잡은_물고기 < 오늘_판_물고기)  # False
print(오늘_잡은_물고기 >= 오늘_판_물고기) # True
print(오늘_잡은_물고기 <= 오늘_판_물고기) # False
print(오늘_잡은_물고기 == 오늘_판_물고기) # False
print(오늘_잡은_물고기 != 오늘_판_물고기) # True
```

Python

비교 연산자는 `if` 조건문과 함께 가장 많이 사용됩니다. `if` 문은 뒤의 챕터에서 자세히 다룰 예정이니 이번 챕터에서는 어떻게 실행되는지만 간단히 알아봅시다.

`if`문은 조건문이 참일 경우에만 실행문이 실행되며 거짓일 경우에는 실행되지 않습니다.

```python
if 조건문 :
    실행문
```

Python

앞의 예제에서 캣의 생선 가게에는 오늘 잡은 물고기를 당일 다 판매해야 하는 규칙이 있었죠?

```python
오늘_잡은_물고기 = 10
오늘_판_물고기 = 5

if 오늘_잡은_물고기 == 오늘_판_물고기 :
    print('오늘 물고기 판매 끝!')

if 오늘_잡은_물고기 != 오늘_판_물고기 :
    print('아직 물고기를 다 팔지 않았습니다.')
```

Python

이렇게 잡은 물고기의 수와 판매한 물고기의 수를 비교하여 값이 같지 않을 경우와 같을 경우에 따라 다른 문구를 출력할 수 있습니다.

위의 코드는 `if-else` 문을 통해 간결하게 표현할 수도 있습니다.

```python
if 오늘_잡은_물고기 == 오늘_판_물고기 :
    print('오늘 물고기 판매 끝!')
else :
    print('아직 물고기를 다 팔지 않았습니다.')
```

Python

`if` 문의 조건이 거짓일 경우에는 if문의 실행문이 실행되고 거짓을 경우에는 `else` 문의 실행문이 실행됩니다. 이 내용도 뒤에서 자세히 배울 내용이니 간단하게 알아보고 넘어가시길 바랍니다.

비교 연산자

Aa 기호	이름	설명	참(True)	거짓(Fal...
>	~보다 큰	왼쪽의 피연산자가 오른쪽의 피연산자보다 크면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	3 > 0	0 > 3
<	~보다 작은	왼쪽의 피연산자가 오른쪽의 피연산자보다 작으면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	3 < 0	3 > 0
>=	~보다 크거나 같은	왼쪽의 피연산자가 오른쪽의 피연산자보다 크거나 같으면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	3 >= 3 3 >= 0	0 >= 3
<=	~보다 작거나 같은	왼쪽의 피연산자가 오른쪽의 피연산자보다 작거나 같으면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	3 <= 3 3 <= 0	0 <= 3
==	같은	피연산자들이 같으면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	3 == 3	3 == 0
!=	다른	피연산자들이 다르면 참(True)를 반환하고 그렇지 않으면 거짓(False)를 반환합니다.	0 != 3	3 != 3

COUNT 6

5. 논리 연산자

논리 연산자는 boolean과 함께 쓰이며, 결과값으로 boolean 값(True, False)을 반환합니다. 논리 연산자에는 **and, or, not** 연산이 있습니다.

```python
#[In]

a = True
b = False

print(a and a) # True
print(a and b) # False
print(a or a)  # True
print(a or b)  # True
print(not a)   # False
print(not b)   # True
```

Python

and 연산은 피연산자 모두 참일 경우에만 True를 반환하며, **or 연산**은 피연산자 중 하나라도 참이면 True를 반환합니다. **not 연산**은 True일 때는 False를 False일 때는 True로 값을 반전시켜 반환합니다.

논리 연산자는 앞서 배웠던 비교 연산자와의 조합을 통해 조건을 더 까다롭고 명확하게 작성할 수 있습니다.

캣의 생선 가게에는 또 다른 규칙이 있었죠? 바로 오늘 매출은 전날 매출의 2배를 목표로 한다는 규칙입니다. 앞의 예제에 규칙을 추가해 봅시다.

```python
#[In]

오늘_잡은_물고기 = 10
오늘_판_물고기 = 10
전날_매출 = 10000
오늘_매출 = 20000

if (오늘_잡은_물고기 == 오늘_판_물고기) and (전날_매출*2 <= 오늘_매출):
    print('오늘 물고기 판매 끝!')
    print('매출 목표 달성!')
else :
    print('오늘의 목표를 달성하지 못했습니다.')
```

Python

and 연산을 사용하여 모두 참일 경우에는 판매 종료와 매출 목표 달성을 알리고 둘 중 하나라도 거짓일 경우에는 "오늘의 목표를 달성하지 못했습니다."라는 문구가 출력됩니다.

or 연산을 사용할 경우에는 둘 중 하나만 조건을 만족하면 실행문이 실행됩니다.

```python
#[In]

if (오늘_잡은_물고기 == 오늘_판_물고기) or (전날_매출*2 <= 오늘_매출):
    print('오늘 물고기 판매 끝!')
    print('매출 목표 달성!')
```

Python

이번에는 not 연산을 사용해 봅시다. boolean 값을 변수에 저장하여 사용할 수도 있습니다.

```python
#[In]

결과값 = 오늘_잡은_물고기 == 오늘_판_물고기

if 결과값 :
    print('오늘 물고기 판매 끝!')

if not 결과값 :
    print('아직 물고기를 다 팔지 않았습니다.')
```

Python

논리 연산자

Aa 기호	이름	설명	참(True)	거짓(False)
and	그리고	양 쪽의 피연산자들 모두 참일 때만 참(True)을 반환하고 피연산자들 중 하나라도 거짓이면 거짓(False)를 반환합니다.	True and True	True and False False and True False and False
or	또는	양 쪽의 피연산자들 중 하나라도 참인 경우에 참(True)을 반환하고 피연산자들이 모두 거짓인 경우에만 거짓(False)를 반환합니다.	True or True True or False False or True	False or False
not	부정	오른쪽 피연산자의 논리 상태를 반전시킵니다.	not False	not True

COUNT 3

6. 비트 연산자

비트 연산자는 **2진수로 이루어진 32비트의 0과 1로 이루어진 집합으로 표현을 하는 연산**이며 진수 체계로 이루어진 숫자들(2진수, 8진수, 10진수, 12진수 등)로 계산하는 것이 아닙니다.

물론 값을 넣을 때는 숫자를 사용합니다. 조금 이해하기 어렵다면 2진수로 논리 연산을 하신다고 생각하시면 됩니다.

비트 연산자

Aa 기호	이름	설명
&	AND 연산	두 피연산자의 각 자리 비트의 값이 둘다 1인 경우 해당 자리에 1을 반환합니다. 하나라도 0인 경우에는 해당 자리에 0을 반환합니다.
|	OR 연산	두 피연산자의 각 자리 비트의 값이 둘다 0인 경우 해당하는 자리에 0을 반환합니다. 하나라도 1인 경우에는 해당 자리에 1을 반환합니다.
^	XOR 연산	두 피연산자의 각 자리 비트의 값이 같을 경우 해당하는 자리에 0을 반환합니다. 두 피연산자의 각 자리 비트의 값이 다른 경우 해당하는 자리에 1을 반환합니다.
~	보수연산	피연산자의 각 자리 비트를 뒤집습니다.
<<	왼쪽 쉬프트 연산	오른쪽에서 0들을 이동시키면서 왼쪽 피연산자 이진수의 각 비트를 오른쪽 피연산자 비트만큼 왼쪽으로 이동시킨 값을 반환합니다.
>>	오른쪽 쉬프트 연산	사라지는 비트를 버리면서 왼쪽 피연산자 이진수의 각 비트를 오른쪽 피연산자 비트만큼 이동시킨 값을 반환합니다.

COUNT 6

💡 비트연산자는 초급 단계에서 사용할 일이 많이 없기 때문에 당장에 어려움을 느끼신다면 이 부분은 넘어가셔도 좋습니다.

```python
#[In]

a = 2
b = 3

print(a & b) # 결과값 : 0
```

Python

위에서 2와 3을 가지고 계산을 했습니다. 이것의 계산 과정을 살펴보겠습니다.

```
# 계산 과정

# a의 값 2를 2진수로 풀어줍니다.
0010

# b의 값 3을 2진수로 풀어줍니다.
0011

# 위아래로 붙여보겠습니다.
0010
0011

# & 연산의 경우 위와 아래 한짝씩 맞춰 and 비교를 해봅시다.
# 이때 1 = True, 0 = False 입니다.
0010
0011
----
0010

# 결과값을 다시 10진수로 변환합니다.
2
```

Python

이번에는 `|` 연산을 해보도록 하겠습니다.

```
#[In]

a = 7
b = 8

print(a | b) # 결과값 : 15
```

Python

```
# 계산 과정

# a의 값 7을 2진수로 변환합니다.
0101

# b의 값 8을 2진수로 변환합니다.
1000

# 위아래로 붙여보겠습니다.
0101
1000

# 연산의 경우 위와 아래 한짝씩 맞춰 or 비교를 해봅니다.
# 이때 1 = True, 0 = False 입니다.
0101
1000
----
1101

# 결과값을 다시 10진수로 변환해줍니다.
15
```

Python

비트 연산자의 연산의 전개 과정은 다음과 같습니다.

1. 10진수를 2진수로 변환
2. 2진수에서 비트연산을 전개
3. 결과값을 다시 10진수로 변환

`XOR` 과 `보수연산` 또한 위의 전개 과정과 같기에 생략하도록 하겠습니다.

왼쪽 쉬프트 연산을 한 번 해보겠습니다. 이 때 주의할 점은 왼쪽 쉬프트 연산의 오른쪽 값은 2진수로 바꾸는 용도가 아닙니다. 그 만큼 비트 이동을 하겠다는 의미입니다.

```
#[In]

a = 7
b = 2

print(a << b)  # 결과값 :28
```

Python

```
# 계산 과정

# a의 값 3을 2진수로 바꿉니다.
0111

# b 만큼 이동할 것이기 때문에 b는 바꾸지 않음

# a의 값들(집합)을 왼쪽으로 b만큼 이동시키고 뒤에 0을 붙임
011100

# 결과 값을 10진수로 다시 변환합니다.
28
```

Python

이번에는 오른쪽 쉬프트 연산을 해보도록 하겠습니다.

```
#[In]

a = 7
b = 2

print(a >> b)  # 결과값 : 1
```

Python

```
# 계산 과정

# a의 값 7을 2진수로 바꿉니다.
0111

# b만큼 이동할 것이기 때문에 b는 바꾸지 않음

# a의 값들(집합)을 오른쪽으로 b만큼 이동시키고 넘어간 것을 잘라냄
0001(11) # ()는 넘어간 것을 의미합니다. 이것들은 잘라내어 버립니다.

0001

# 결과 값을 10진수로 다시 변환합니다.
1
```

Python

7. 멤버 연산자

멤버 연산자는 **특정 항목이 배열에 들어있는지 확인하는 연산**입니다. 결과값이 참이면 **True**, 거짓이면 **False**를 반환합니다.

배열 `오늘_잡은_물고기` 에는 'A등급'이 있기 때문에 in 연산의 결과값은 True, not in 연산의 결과값은 False가 출력됩니다.

```python
#[In]

# 오늘 A등급 물고기를 잡았나요?
오늘_잡은_물고기 = ['A등급', 'A등급', 'B등급', 'C등급', 'C등급', 'C등급']

print('A등급' in 오늘_잡은_물고기) # True
print('A등급' not in 오늘_잡은_물고기) # False
```

Python

배열 `오늘_잡은_물고기` 에는 'D등급'이 없기 때문에 in 연산의 결과값은 False, not in 연산의 결과값은 True가 출력됩니다.

```python
# 오늘 D등급 물고기를 잡았나요?
오늘_잡은_물고기 = ['A등급', 'A등급', 'B등급', 'C등급', 'C등급', 'C등급']

print('D등급' in 오늘_잡은_물고기) # False
print('D등급' not in 오늘_잡은_물고기) # True
```

Python

멤버 연산자

✨ in	왼쪽 항목이 오른쪽 배열에 포함되면 True 그렇지 않으면 False
✨ not in	왼쪽 항목이 오른쪽 배열에 포함 안 되면 True 그렇지 않으면 False

8. 식별 연산자

식별 연산자는 **해당 값이 들어있는** `주소` **를 비교하는 연산자**입니다. C, C++ 언어를 학습하신 분이라면 포인터로 이해하시면 됩니다.

그렇지 않으신 분을 위해 간단히 설명하자면, **변수를 선언해서 값을 저장할 때 그 값을 저장하는 공간이 필요한데 그 공간의 위치를 주소**라고 합니다. 저희들이 사용하는 주소와 비슷한 개념이라고 보시면 됩니다. OO시 OO동 같은 주소 체계를 컴퓨터에서도 가지고 있는 것입니다.

식별 연산자

✨ is	피연산자들의 위치(주소)가 같다면 True 그렇지 않으면 False
✨ is not	피연산자들의 위치(주소)가 다르면 True 그렇지 않으면 False

`위니브윌드` 에서 `licat` 이라는 사람이 일하고 있다고 가정합시다. 또 `바울랩` 이라는 회사에도 `licat` 이라는 사람이 일하고 있을 경우 이 둘이 동일인물인지 확인해봅시다.

```python
#[In]

위니브윌드 = "licat"
바울랩 = "licat"

print(위니브윌드 is 바울랩)  # True
print(위니브윌드 == 바울랩)  # True
```

Python

같다면 동일인물이라는 것을 알 수 있습니다.

즉, 이 두 회사에 다니고 있는 `licat` 이라는 이름을 가진 사람은 `OO시 OO동 XX번지` 에서 살고 있는 동일인물이라고 생각하면 됩니다.

> 💡 물론 실제로 컴퓨터 내에서의 주소는 이런 주소가 아닌 0012FF64 와 같은 16진수 주소 체계를 가집니다.

이때 그럼 `==` 와 차이가 뭔지 궁금해할 수 있습니다. `==` 는 값이 같은지 찾는 비교 연산자입니다. 위의 예시로 보자면 같은 `licat` 이라는 이름을 가졌는지를 보는 것입니다.

다음 예시를 보면서 확인해 보겠습니다.

```python
#[In]

a = [1, 2, 3]
b = [1, 2, 3]
c = a

print('a == b', a == b)
print('a == c', a == c)
print('a is b', a is b)
print('a is c', a is c)
```

Python

```
#[Out]

a == b True
a == c True
a is b False
a is c True
```

Python

위에서는 `licat` 이라는 문자열을 예시로 들었습니다. 파이썬의 경우 동일한 문자열이 있다면 해당 문자열을 가진 주소를 그대로 참조하게 되어있습니다. 그렇기 때문에 동일한 문자열을 생성하는 경우에는 동일한 주소를 보고 있게 됩니다. (항상 그런 것은 아닙니다.)

배열과 같은 객체는 문자열과 달리 새로 생성할 때 새로운 주소에 담습니다. 따라서 값은 동일하더라도 다른 주소를 가지게 될 수 있습니다.

따라서 위의 예시와 같이 `a`, `b`, `c`의 값을 비교하는 것은 `True`로 나오지만 주소를 비교했을 때는 다르게 나오는 것을 볼 수 있습니다.

9. 연산자의 우선순위

연산자의 우선순위는 어떤 연산을 먼저 할 것인지 정하는 규칙입니다. 따라서 우선순위가 높은 연산자부터 연산을 진행합니다.

연산자의 우선순위

Aa 순위	연산자 기호	설명
1	(), {}, []	Tuple, Set, List, Dictionary
2	**	거듭제곱
3	+, -, ~	단항 연산자
4	*, /, //, %	곱하기, 나누기, 몫(정수), 나머지
5	+, -	더하기, 빼기
6	<<, >>	쉬프트 연산
7	&	비트연산 AND
8	^, |	비트연산 XOR, 비트연산 OR
9	in, not in, is, is not, <, <=, >, >=, ==, !=	멤버 연산자, 식별 연산자, 비교 연산자
10	not	논리 부정
11	and	논리 AND
12	or	논리 OR

COUNT 12

연산자의 우선순위는 중요한 개념이나 위 표를 보고 모두 익히기에는 어렵습니다.

먼저, 사칙연산의 우선순위는 대부분 다 아실 것입니다. (`+` , `-` , `*` , `/`) 곱셈과 나눗셈이 우선이고, 덧셈과 뺄셈이 그 다음 순위입니다.

그 외의 연산자들은 천천히 사용하면서 익혀보며 혼란이 있을 경우에는 가장 높은 우선순위인 괄호 `()` 를 이용하여 묶어서 처리하는 것이 더 직관적일 수 있습니다.

```python
#[In]

a = 10 + 3 * 5
print(a)  # 25

a = (10 + 3) * 5
print(a)  # 65
```

Python

이렇게 괄호를 사용하면 우선순위가 달라져 연산의 결과가 달라질 수 있습니다.

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

캣의 해골섬 출항!

PYTHON BOOTCAMP

1. **캣의 통장에 노드(위니브 월드의 통화단위)를 저축하시오!**

   캣은 물고기를 잡아 파는 행위를 반복하였어요. 어느새 캣의 통장에는 100만 노드라는 돈이 쌓였습니다. 또, 매일 스킬이 오르고, 요령도 생겨 물고기를 10마리씩 더 잡게 되었습니다.

   물고기의 가격은 30노드이며 이번달 1일에는 110마리, 2일에는 120마리, 3일에는 130마리 씩 10일간 물고기를 잡아 팔았다고 했을 때 캣의 통장에는 얼마의 돈이 쌓여 있을까요?

```python
캣의_통장 = 1000000
물고기의_가격 = 30
잡은_물고기의_수 = '정답을 입력하세요.'
물고기를_팔고_난_후_캣의_통장 = '정답을 입력하세요.'
```

Python

2. **물고기가 잘 잡히는 반경(넓이)을 구하시오!**

   캣은 물고기를 잡으러 갈 때마다 물고기가 잘 잡히는 곳이 있다는 사실을 알게 되었습니다. 물고기가 잘 잡히는 곳은 저 멀리 해골섬 주위로 반경 500m의 큰 원을 그리고 있습니다.

```python
반지름 = 500
고기가_잘_잡히는_반경 = '정답을 입력하세요.'
```

Python

3. **물고기를 다 잡고난 후 캣의 통장 잔고를 구하시오!**

   원의 1 넓이당 물고기가 110마리씩 살고 있습니다. 해골섬의 넓이는 314입니다. 이 물고기를 다 잡았을 때 통장 잔고에 얼마가 있을지 구하세요. 현재 통장 잔고는 '물고기를_팔고_난_후_캣의_통장'입니다.

```python
해골섬의_넓이 = 314
반지름 = 500
물고기를_다_잡고_난_후_캣의_통장 = '정답을 입력하세요.'
```

4. **번 돈과 비율을 정리하시오!**

   A등급 물고기와 B등급 물고기가 각각 100노드, 50노드이고 운이 좋게도 A등급은 350마리, B등급은 700마리를 잡았다면 총 얼마의 돈을 벌었을까요? 그리고 총액에서 A등급이 차지하는 비율과 B등급이 차지하는 비율을 구하십시오. 여기서 비율은 Dictionary로 구하십시오.

```python
물고기_등급 = {'A등급':100, 'B등급':50}
총액 = '정답을 입력하세요.'
비율 = '정답을 입력하세요.'
```

모두 완료하였다면 아래 훈장을 추가하세요!

```python
훈장.append('해골섬 낚시꾼')
```

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

3편 효율적으로 생선 잡기

1. 캣의 고민
    1.1 들여쓰기
    1.2 함수의 사용
    1.3 print VS return
2. 함수를 사용한 캣의 계산
    2.1 상수(Constant)
3.전역변수 global
4. function 응용
    4.1 함수 안에 함수 만들기
5. 연산자 우선순위

1. 캣의 고민

해골섬에서 잡은 물고기는 살이 통통하고 맛이 일품이라 인기가 날이 갈수록 높아졌습니다. 심지어 웃돈을 주고 생선을 사기까지 이르렀어요. 캣은 넘쳐나는 수요에 깊은 고민에 빠졌습니다.

"더 효율적으로 많은 생선을 잡을 방법이 없을까냥?"

이때 지나가던 상인이 작게 귀띔을 해주었습니다.

"위니브 왕국에서는 아주 큰 그물과 통발을 사용하여 생선을 많이 잡아들인다는 소문이 있소냥~"

캣은 조언에 귀를 기울이고 큰 그물과 통발을 구하러 다녔습니다. 그리고 그 사용법을 익히기 시작했어요.

"여러 도구를 사용하여 다양한 방법으로 생선을 잡아봐야겠다냥!"

```python
#[In]

# 현재 캣이 보유한 기술
# 기술 = ['고기잡이', '고기팔기']
기술.append('낚시_Lv1')
기술.append('통발_Lv1')
기술.append('큰그물_Lv1')
```

Python

- **라이캣의 현재 상태창!**

```python
#[In]

이름 = '캣'
설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
나이 = 15
오늘_잡은_물고기 = '1000'
키 = '45.5cm'
몸무게 = 1.3
육식 = True
초식 = True
돈 = 1000
훈장 = ['백상아리를 잡은 고양이', '성실한 납세자', '해골섬 낚시꾼']
기술 = ['고기잡이', '고기팔기', '낚시_Lv1', '통발_Lv1', '큰그물_Lv1']
```

Python

- 규칙 -
1. 낚시는 A등급 생선을 한 마리씩 잡을 수 있습니다.
2. 그물로는 한 번에 A등급 3마리, B등급 3마리, C등급 4마리를 잡을 수 있습니다.
3. 통발은 하루에 한 번만 확인하며 문어 한 마리를 잡을 수 있습니다.

```python
#[In]

# 캣의 생선 잡기
# 함수 정의하기
def 낚시():
    print('A등급 생선을 한 마리 잡았습니다.')

def 그물():
    print('A등급 3마리, B등급 3마리, C등급 4마리를 잡았습니다.')

def 통발():
    print('문어를 하나 잡았습니다.')

# 함수 호출하기
낚시()
그물()
통발()
```

Python

여기서 `Alt + Enter` 를 누르면 아래와 같이 출력됩니다.

```
#[Out]

A등급 생선을 한 마리 잡았습니다.
A등급 3마리, B등급 3마리, C등급 4마리를 잡았습니다.
문어를 하나 잡았습니다.
```

Python

1.1 들여쓰기

Python에서는 **들여쓰기(intend)로 함수의 범위**를 정합니다. 함수의 범위를 규정하는 것은 화이트 스페이스(공백)입니다. 탭으로 한번에 들여 쓸 수 있지만 파이썬 개발 제안서(PEP 8)에서는 **스페이스 4번**으로 하기로 약속했으니 스페이스를 사용해주시길 바랍니다.

```
#[In]

def 낚시():
    print('A등급 생선을 한 마리 잡았습니다.')
    print('B등급 생선을 한 마리 잡았습니다.')

낚시()
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

A등급 생선을 한 마리 잡았습니다.
B등급 생선을 한 마리 잡았습니다.
```

Python

```
#[In]

def 낚시():
    print('A등급 생선을 한 마리 잡았습니다.')

print('B등급 생선을 한 마리 잡았습니다.')
낚시()
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

B등급 생선을 한 마리 잡았습니다.
A등급 생선을 한 마리 잡았습니다.
```

Python

💡 이처럼 들여쓰기를 통해 함수의 범위를 다르게 하면 결과가 달라지므로 함수를 사용할 때는 들여쓰기에 유의해야 합니다.

1.2 함수의 사용

```
#[In]

def function(x, y):
    z = x + y
    return z

# A등급 10마리, B등급 9마리
print('오늘 잡은 생선 :', function(10, 9))
```

Python

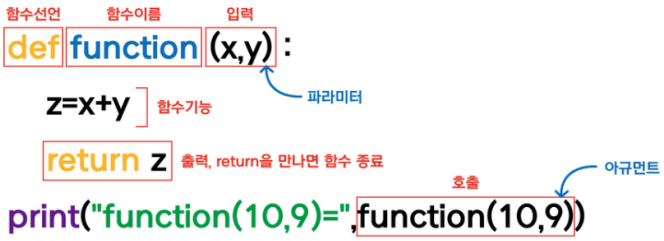
위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

오늘 잡은 생선 : 19
```

Python

이처럼 **입력과 기능 및 연산, 출력 값이 있는 것을 '함수'**라고 말합니다. 함수를 선언하였다면 호출해야만 함수가 작동하게 됩니다.

위 예제에서는 function(10, 9)라는 문장으로 함수를 호출합니다.



입력과 출력값 등 함수의 모든 요소가 필요충분조건인 것은 아닙니다.

```python
#[In]

def 낚시():
    print('A등급 생선을 한 마리 잡았습니다.')
```

Python

위 예제의 함수 `낚시()` 와 같이 input이 없는 함수도 있고, return 값이 없는 함수도 있습니다. return 값이 없는 함수는 자동으로 None이 할당됩니다. 또한 function이 없는 함수도 있습니다.

1.3 print VS return

```python
#[In]

def function(x, y):
    z = x + y
    print(z)

print('오늘 잡은 생선 :', function(10, 9))
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

19
오늘 잡은 생선 : None
```

Python

위 예제에서 `function(10, 9)`를 호출한 값을 출력해 보았더니 None이 출력됩니다. 이는 이 함수의 return 값이 없기 때문입니다.

> 💡 종종 print와 return을 헷갈려 하시는 분들이 계십니다. print는 다른 함수라는 걸 기억해주세요!

2. 함수를 사용한 캣의 계산

```python
#[In]

def 계산(가격, 개수):
    return 가격 * 개수

print(계산(1000, 5), '원입니다.')
```

Python

가격과 개수를 매개변수로 입력하여 금액을 출력할 수 있습니다. 실행하면 아래와 같이 출력됩니다.

```python
#[Out]

5000 원입니다.
```

Python

> 등급 가격은 정해져 있는데 매번 입력하는 것은 귀찮다냥!

캣의 생선 가게에서는 등급에 따라 가격이 정해져 있기 때문에, 매번 값을 입력하는 것이 귀찮은 캣!
"매개변수와 연산을 수정하면 되겠다냥~"

```python
#[In]

# A등급:1000원, B등급:500원, C등급:100원

def 계산(a, b, c):
    합계 = a*1000 + b*500 + c*100
    return 합계

print(계산(5, 2, 3), '원입니다.')
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```python
#[Out]

6300 원입니다.
```

Python

2.1 상수(Constant)

상수(Constant)란 **변경할 수 없는 수**입니다. C언어에서는 define이라는 문장을 사용하여 상수를 정의하지만, Python에서는 이러한 상수를 선언하는 문법이 없습니다. 다만 **일반적으로 이름 전부를 대문자로 선언**하면 상수라고 이해합니다.

```python
#[In]

PI = 3.14
def circle(r,inputpi):
    z = r*r*inputpi
    return z
result = circle(10, PI)
print(result)
```

Python

위 코드를 실행시키면 아래와 같이 출력이 됩니다.

```python
#[Out]

result : 314.0
```

Python

여기서는 `PI = 3.14` 이 상수가 됩니다. 한번 더 주의 깊게 보아야 할 내용은 함수에서 인자값으로 받는 변수가 `PI` 가 아니라 `inputpi` 라는 것입니다. 이 개념에 대해서는 다음장 지역변수와 전역변수를 다루며 말씀드리도록 하겠습니다.

```python
#[In]

π = 3.14
def circle(r, inputpi):
    z = r*r*inputpi
    return z
result = circle(10,π)
print(result)
```

Python

이처럼 특수문자로도 사용이 가능합니다. 같은 원리로 한글 코딩도 가능합니다. 한글 코딩의 장점으로는 고유명사를 표현하기 좋으며, 교육용으로도 주목받고 있습니다.

HTML, CSS, Javascript등 다양한 분야에서 한글코딩이 주목받고 있으며 자세한 내용은 '한글코딩.org'에서 확인할 수 있습니다. 아래는 한글코딩 예시입니다.

```python
#[In]

def 한글코딩(글자):
    print(글자)
한글코딩('안녕, 세상아.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```python
#[Out]

안녕, 세상아
```

Python

3. 전역변수 global

지역변수는 함수 내부에서만 사용되는 변수입니다. 함수가 호출되는 동안에만 효력을 발휘하고 함수가 끝나는 동시에 소멸됩니다.

전역변수는 프로그램 어디에서나 접근이 가능한 변수이며 함수 내에서 **global**을 입력하면 전역변수가 됩니다.

```python
#[In]

A등급 = 0

def 낚시():
    global A등급
    A등급 += 1

낚시()
낚시()
낚시()

print('잡은 A등급 :', A등급, '마리')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```python
#[Out]

잡은 A등급 : 3 마리
```

Python

위 코드에서는 함수가 호출될 때마다 전역변수 A등급에 1씩 더합니다. 총 3번 호출되었으므로 전역변수 A등급에는 3이 저장됩니다.

이번에는 낚시, 그물, 통발을 모두 사용하여 잡은 생선의 총 합계를 구해봅시다.

```python
#[In]

# 캣의 생선 잡기

A = 0
B = 0
C = 0
문어 = 0

def 낚시():
    global A
    A += 1

def 그물():
    global A, B, C
    A += 3
    B += 3
    C += 4

def 통발():
    global 문어
    문어 += 1

낚시()
그물()
통발()

합계 = A + B + C + 문어
print('오늘 잡은 생선의 합계 :', 합계)
```

Python

위 코드에서는 모든 함수에서 전역변수를 사용하기 때문에 함수가 변수의 값에 영향을 줍니다. 실행시키면 아래와 같이 출력됩니다.

```python
#[Out]

오늘 잡은 생선의 합계 : 12
```

Python

지역변수와 전역변수를 왜 나누어 놓았나요?

예를 들어 물고기를 어떻게 잡았는지 구분하는 업무를 하게 되었다고 가정해 보겠습니다. 캣은 통발로 잡았을 때 x라는 변수를 사용하고 캣의 직원은 낚시로 잡았을 때 x라는 변수를 사용하는데, 이 변수를 자신이 만든 프로그램이 아닌 다른 프로그램에서 조작이 되어진다면 프로그램을 만드는 과정에서 혼선이 생길 수 있습니다.

따라서 이러한 혼선을 막기 위해서 프로그래밍 언어에서는 **함수 내에서 선언된 변수와 구현 내용을 외부에서 만지지 못하도록 지원**하고 있습니다.

```python
#[In]

# global 사용전
a = 100
def f():
    a = a + 1
f()
```

Python

```python
#[In]

# global 사용후
a = 100
def f():
    global a
    a = a + 1
f()
```

Python

함수 안에서는 함수 밖에 있는 변수에 접근하지 못합니다. 이를 위해 **global**이라는 함수를 사용하여 모든 함수에서 사용 가능하도록 만들 수 있어요.

💡 그러나 프로그래밍에서 전역변수는 권장하지 않습니다. 연산을 하고 싶다면 함수에 아규먼트(함수의 input)를 대입하여 return으로 받는게 객체 지향 프로그래밍에 적합합니다.

4. function 응용

4.1 함수 안에 함수 만들기

```python
def 함수이름1():
    코드
    def 함수이름2():
        코드
```

Python

이번에는 위 코드처럼 **함수 속에 함수**를 만드는 방법입니다. 위와 같이 def 로 함수를 만들고 그 안에 다시 def로 함수를 만들면 됩니다.

```python
#[In]

def print_text():
    text = '생선을 잡아보자!'
    def txt():
        print(text)
    txt()

print_text()
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

생선을 잡아보자!
```

Python

함수 `print_text()` 안에서 다시 def로 함수 `txt()`를 만들었습니다. 그리고 `print_text()`안에서 `txt( )`처럼 함수를 호출했습니다. 하지만 아직 함수를 정의만 한 상태이므로 아무것도 출력되지 않습니다.

두 함수가 실제로 동작하려면 바깥쪽에있는 `print_text()`를 호출해주어야 합니다. 즉, `print_text()` → `print()` 순으로 실행됩니다.

이 예제에서도 마찬가지로 함수들이 실제로 동작하기 위해서는 함수 밖에서 `생선잡기()` 를 호출해주어야 합니다.

내가 나를 호출하는 **재귀함수**에 대해서는 깊은 물에서 알아보도록 하겠습니다! 아래 간단한 재귀함수에 대한 코드가 있는데요. 이렇게 사용할 수 있구나 정도를 파악하고 가시면 좋을 것 같아요.

```python
def factorial(x):
    if x == 1:
        return 1
    else:
        return x * factorial(x-1)

factorial(5)
```

Python

5. 연산자 우선순위

연산자 우선순위는 앞서 <**2편 생선을 잡아라**>에서 알아보았습니다. 아래 표에 추가된 내용을 한 번 살펴보고 넘어가시길 바랍니다.

다음은 파이썬의 연산자 우선순위입니다.

연산자 우선순위

Aa 우선순위	연산자	설명	+
1	(값...), [값...], {키:값...}, {값...}	튜플, 리스트, 딕셔너리, 세트 생성	
2	x[인덱스], x[인덱스:인덱스], x(인수...), x 속성	리스트(튜플) 첨자, 슬라이싱, 함수 호출, 속성 참조	
3	await x	await 표현식	
4	**	거듭제곱	
5	+x, -x, ~x	단항 덧셈(양의 부호), 단항 뺄셈(음의 부호), 비트 NOT	
6	*, @, /, //, %	곱셈, 행렬 곱셈, 나눗셈, 버림 나눗셈, 나머지	
7	+, -	덧셈, 뺄셈	
8	<<, >>	비트 시프트	
9	&	비트 AND	
10	^	비트 OR	
11	|	비트 XOR	
12	in, not in, is, is not <, <=, >, >=, ==, !=	포함 연산자, 객체 비교 연산자, 비교 연산자	
13	not x	논리 NOT	
14	and	논리 AND	
15	or	논리 OR	
16	if else	조건부 표현식	
17	lamda	람다 표현식	

+ New

COUNT 17

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

4편 생선 회사를 운영하라

1. 캣의 생선회사
    1.1 캣의 회사 운영
    1.2 if 문의 기본 구조
2. if, else 문
3. if, elif, else

1. 캣의 생선 회사

캣의 생선 가게는 어느새 마을에서 가장 유명한 생선 가게가 되었습니다. 장사가 잘 되어 기쁜 캣은 한편으로는 걱정이 되었습니다. 캣의 작은 가게는 불어난 손님을 다 수용하기에는 역부족이었기 때문입니다.

"바쁘다 바빠! 이젠 작은 생선가게에서 벗어날 때가 된 것 같다냥!"

캣은 작은 생선가게를 큰 생선회사로 업그레이드 시키기로 결심하였습니다.

"생선회사로 업그레이드! 체계적인 관리와 시스템으로 많은 돈을 벌어보겠다냥! 그 전에, 직원부터 뽑자냥!"

- 라이캣의 상태창!

```
#[In]

이름 = '캣'
설명 = '위니브 월드 외각에 살고 있는 생선가게 주인 캣(cat)'
나이 = 17
오늘_잡은_물고기 = '10000'
키 = '45.7cm'
몸무게 = 1.4
육식 = True
초식 = True
돈 = 100000
훈장 = ['백상아리를 잡은 고양이', '성실한 납세자', '해골섬 낚시꾼']
기술 = ['고기잡이', '고기팔기', '낚시_Lv3', '통발_Lv3', '큰그물_Lv3']
```

Python

- 라이캣의 기술 Update!

```
#[In]

기술[2] = 기술[2][:-1] + '3'
기술[3] = 기술[3][:-1] + '3'
기술[4] = 기술[4][:-1] + '3'
```

Python

1.1 캣의 회사 운영

직원을 채용하기로 한 캣은 직원을 뽑을 때 성실성과 프로그래밍 능력을 보기로 합니다.
성실성이 80점 이상이고 프로그래밍 능력이 70점 이상일 경우에 합격입니다.

```python
#[In]

#캣의 직원 채용

성실성 = 85
프로그래밍능력 = 70

if 성실성 >= 80 and 프로그래밍능력 >= 70:
    print('합격입니다.')
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```
#[Out]

합격입니다.
```

Python

이처럼 **조건을 판단하여 참일 경우에 해당 조건에 맞는 것을 실행하는 것을 if문**이라고 합니다.

위 예제에서 조건문은 `성실성 >= 80 and 프로그래밍능력 >= 70` 입니다. 성실성은 85이고 프로그래밍 능력은 70이므로 조건에 만족하여 `print('합격입니다.')` 가 실행됩니다.

생선 회사의 매출은 모두 손님 덕분! 보답하겠다냥!

직원 채용으로 차츰 안정되고 매출도 꾸준히 늘고있는 캣의 생선회사. 캣은 손님들을 향한 고마움을 '**할인 제도**'를 통해 보답하고자합니다.

"만원이상 구입하면 천원을 할인한다냥!"

```python
#[In]

#생선회사의 할인제도
# A등급:1000원, B등급:500원, C등급:100원

def 계산(a, b, c):
    가격 = {'A등급':1000, 'B등급':500, 'C등급':100}
    합계 = a*가격['A등급'] + b*가격['B등급'] + c*가격['C등급']
    return 합계

총합 = 계산(10, 2, 3)

if 총합 >= 10000:
    print('할인을 해주겠다냥!')
    print(f'총 {int(총합 * 0.9)}노드를 내면 된다냥!') #10% 할인
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```python
#[Out]

할인을 해주겠다냥!
총 10170노드를 내면 된다냥!
```

Python

위 예제에서 함수 `계산(a, b, c)` 를 통해 총합을 구합니다. 조건문은 `총합 >= 10000` 으로 총합이 13000으로 10000 이상이므로 출력문이 실행됩니다.

이 취업난 시대에 한명의 청년이라도 더 고용하는 라이캣에게 훈장을 수여하도록 하겠습니다.

```python
#[In]

훈장.append('청년 고용 착한 기업')
훈장.append('회사를 설립한 자')
```

Python

1.2 if 문의 기본 구조

배운 내용을 if 문의 기본 구조를 통해 개념을 정리하고 넘어갑시다.

```python
#if 문의 기본구조
if 조건문 :
  수행할 문장 1
  수행할 문장 2
```

Python

조건문이 참이면 `수행할 문장 1` 과 `수행할 문장 2` 를 수행합니다.

```python
if 조건문 :
    수행할 문장 1
수행할 문장 2
```

Python

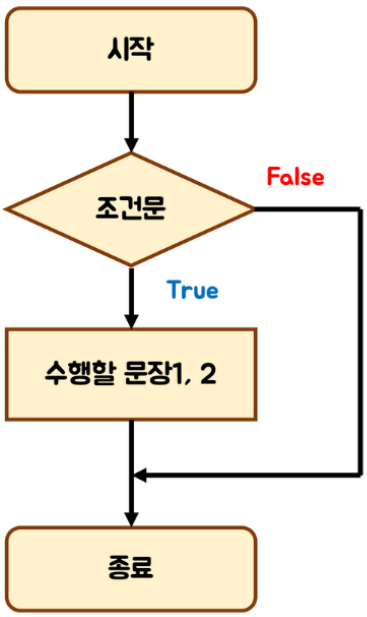
위의 if문에서는 조건문이 참일 경우에 `수행할 문장 1` 을 수행하고 `수행할 문장 2` 는 **if문 범위 밖**에 있기 때문에 조건문에 상관없이 수행됩니다.

```python
if True:
    print('True')
```

Python

또한 위 예제 코드처럼 조건문에 직접 boolean 값(True, False)을 넣을 수도 있습니다.

- if 문의 순서도(Flow Chart)



2. if, else 문

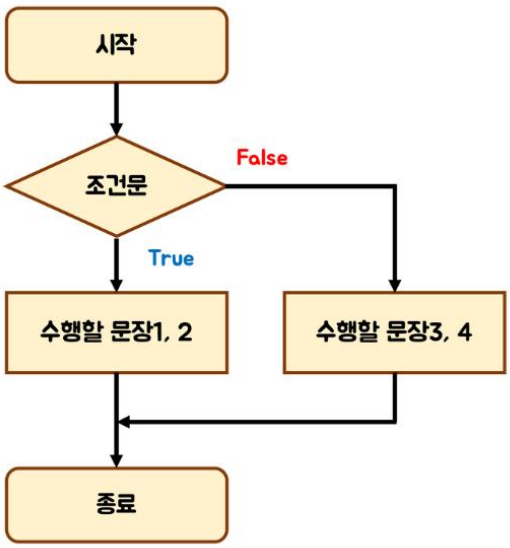
else 문은 if 문의 **조건이 거짓일 경우에 실행**됩니다. 아래 if, else 문의 기본 구조를 통해 알아봅시다.

```python
#if,else문의 기본구조
if 조건문:
  수행할 문장 1
  수행할 문장 2
else:
  수행할 문장 3
  수행할 문장 4
```

Python

if 문의 조건문을 확인해서 만약 **참**이면 `수행할 문장 1` 과 `수행할 문장 2` 를 수행하고, 조건이 **거짓**이면 else 문이 실행되어 `수행할 문장 3` 과 `수행할 문장 4` 가 수행됩니다.

- if, else 문의 순서도(Flow Chart)




```python
#[In]

#캣의 직원 채용
성실성 = int(input('성실성을 입력하세요 :'))
프로그래밍능력 = int(input('프로그래밍능력을 입력하세요 :'))

if 성실성 >= 80 and 프로그래밍능력 >= 70:
    print('합격입니다.')
    if 성실성 >= 90:
        print('보너스를 드립니다.')
else:
    print('불합격입니다.')
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```
#[Out]

성실성을 입력하세요 :80
프로그래밍능력을 입력하세요 :80
합격입니다.
```

Python

위 예제에서는 성실성과 프로그래밍 능력을 `input()` 함수를 통해 입력받습니다.

if 문의 조건문을 만족할 경우 `합격입니다.` 가 출력됩니다. 조건문을 만족하지 못할 경우에는 else 문으로 넘어가 `불합격입니다.` 가 출력됩니다.

위 예제에서는 성실성과 프로그래밍 능력을 `input()` 함수를 통해 입력받습니다.

if 문의 조건문을 만족할 경우 `합격입니다.` 가 출력됩니다. 조건문을 만족하지 못할 경우에는 else 문으로 넘어가 `불합격입니다.` 가 출력됩니다.

> 💡 `input()` 은 Python에서 입력을 받을 때 사용하는 함수이며, 입력받은 값을 `int()` 를 통해 정수형으로 변환할 수 있습니다.

이번에는 총합을 입력받고 else문을 추가하여 조건이 거짓일 경우에는 할인을 하지 않은 금액을 출력해 봅시다.

```python
#[In]

#생선회사의 할인제도
총합 = int(input('가격을 입력하세요 :'))

if 총합 >= 10000:
    print('할인을 해주겠다냥!')
    print(f'총 {int(총합 * 0.9)}노드를 내면 된다냥!') #10% 할인
else:
    print(f'총 {총합}노드를 내면 된다냥!')
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```
#[Out]

# 1
가격을 입력하세요 :20000
할인해드립니다!
총 18000 원입니다.

# 2
가격을 입력하세요 :9000
총 9000 원입니다.
```

Python

위 예제에서는 총합을 입력받고 if 문에서 조건을 검사합니다. 총합이 10000 이상일 경우에는 if 문 다음 문장이 수행되고 거짓일 경우에는 else 문이 실행됩니다.

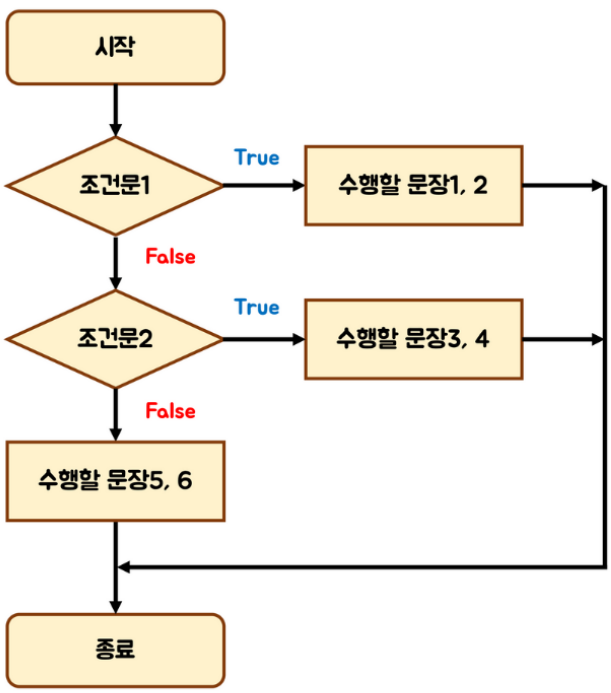
3. if, elif, else

```python
#if,elif,else문의 기본구조

if 조건문1:
  수행할 문장 1
  수행할 문장 2
elif 조건문2:
  수행할 문장 3
  수행할 문장 4
else:
  수행할 문장 5
  수행할 문장 6
```

Python

if 문에서 조건문을 확인해서 만약 참이면 `수행할 문장 1` 과 `수행할 문장 2` 를 수행합니다. 그렇지 않고 거짓일 경우에는 elif 문의 조건문을 확인합니다. elif 문에서도 마찬가지로 조건문이 참일 경우에는 `수행할 문장 3` 과 `수행할 문장 4` 를 수행하고 거짓이면 else 문이 실행됩니다.

- **if, elif, else문의 순서도**(Flow Chart)



이번에는 가격과 손님에게 받은 돈을 입력받아 작성해 봅시다.

```python
#[In]

# 거스름돈
가격 = int(input('가격을 입력하세요 :'))
받은돈 = int(input('받은돈을 입력하세요 :'))

if 가격 < 받은돈:
    print('여기 거스름돈', 받은돈-가격, '원입니다.')
elif 가격 > 받은돈:
    print(가격-받은돈, '원이 부족합니다.')
else:
    print('감사합니다.')
```

Python

위 코드를 실행하면 아래와 같이 출력됩니다.

```python
#[Out]

# 1
가격을 입력하세요 :9000
받은돈을 입력하세요 :10000
여기 거스름돈 1000 원입니다.

# 2
가격을 입력하세요 :9000
받은돈을 입력하세요 :8000
1000 원이 부족합니다.

# 3
가격을 입력하세요 :9000
받은돈을 입력하세요 :9000
감사합니다.
```

Python

위 예제에서는 `가격`과 `받은돈`을 입력받아 둘의 값을 비교합니다. 가격보다 받은돈이 클 경우에는 `받은돈-가격` 의 연산 결과인 거스름돈을 출력하고, 작을 경우에는 돈이 부족하다는 문구를 출력합니다.

if 문과 elif 문의 조건이 모두 거짓일 경우 즉, 가격과 받은돈의 값이 같을 경우에는 "감사합니다."라고 출력합니다.

위에서 배운 내용을 가지고 할인 내역을 좀 더 세분화 해봅시다.

```python
#[In]

# A등급:1000원, B등급:500원, C등급:100원

def 계산(a, b, c):
    가격 = {'A등급':1000, 'B등급':500, 'C등급':100}
    합계 = a*가격['A등급'] + b*가격['B등급'] + c*가격['C등급']
    return 합계

총합 = 계산(20, 2, 3)
print(총합)

if 총합 >= 20000:
    print('20% 할인을 해주겠다냥!')
    print(f'총 {int(총합 * 0.8)}노드를 내면 된다냥!') #20% 할인
elif 총합 >= 10000:
    print('10% 할인을 해주겠다냥!')
    print(f'총 {int(총합 * 0.9)}노드를 내면 된다냥!') #10% 할인
elif 총합 >= 5000:
    print('5% 할인을 해주겠다냥!')
    print(f'총 {int(총합 * 0.95)}노드를 내면 된다냥!') #5% 할인
else:
    print(f'총 {총합}노드를 내면 된다냥!')
```

Python

위 코드의 실행 결과는 아래와 같습니다.

```
#[Out]

21300
20% 할인을 해주겠다냥!
총 17040노드를 내면 된다냥!
```

Python

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

캣의 고객관리

PYTHON BOOTCAMP

캣은 효율적으로 고객들을 관리하기 위해 고객들을 회원으로 등록하고 구매 금액의 10%를 적립해주는 "**적립금 제도**"를 시작하였습니다.

단, 적립금은 **회원인 경우에만 적립**할 수 있으며 **10,000노드 이상이 모이면 생선 구매 시 현금처럼 사용**할 수 있습니다.

---

1. **회원으로 등록하겠습니까?**

   손님 'A캣'과 'D캣'은 캣의 생선 회사에 회원인지 확인하고 싶습니다. 만약 회원이 아닐 경우에는 회원으로 등록하고 싶다고 합니다.

   아래 코드를 작성하여 'A캣'과 'D캣'이 회원인지 확인하고 회원으로 등록해 주세요.

```python
#[In]

회원명 = input('회원명 입력하세요 : ')
회원 = ['A캣', 'B캣', 'C캣']

def 회원등록():
    '정답을 입력해주세요.'

if '정답을 입력해주세요.':
    '정답을 입력해주세요.'
else:
    회원등록()
```

Python

```
#[Out]

회원명 입력하세요 : A캣
이미 등록된 회원입니다.

---

회원명 입력하세요 : D캣
['A캣', 'B캣', 'C캣', 'D캣']
회원으로 등록되었습니다.
```

Python

2. **회원에게 적립금을 지급하라!**

   손님 '애옹'이와 '냐옹'이는 생선을 각각 15,000노드, 5,000노드씩 구입하였습니다. 이때 '애옹'이와 '냐옹'이가 회원인 경우를 체크합니다. 회원인 경우 적립받게 되는 금액을 출력하고, 회원이 아닌 경우에는 아직 회원이 아닙니다를 출력해주세요.

```
#[In]

구매가격 = input('가격을 입력하세요 :')
회원명= input('회원명을 입력하세요 :')

회원 = ['씨-캣', '자바캣', '파이캣', '썬캣', '애옹']
```

Python

```
#[Out]

회원명 입력하세요 : 애옹
회원입니다.
15000노드 중 1500노드가 적립됩니다.

---

회원명 입력하세요 : 냐옹
회원이 아닙니다. 회원가입을 해주시기 바랍니다.
```

Python

3. **적립금을 사용하고 싶어요!**

   손님 '파이캣'과 '썬캣'은 생선을 각각 10,000노드씩 구매하였습니다. 두 손님은 적립금을 사용할 수 있는지 알고 싶습니다.

   사용할 수 있다면 적립금 5,000노드를 사용합니다. 이때 구매 가격과 남은 적립금은 얼마인지 출력하고 남은 적립금은 Dictionary에 반영해 주세요.

   그렇지 않을 경우에는 현재 적립금은 얼마인지 출력해주세요.

```python
#[In]

구매가격 = input('가격을 입력하세요 :')
회원명 = input('회원명을 입력하세요 :')

#회원 = {회원명:적립금}
회원 = {'씨-캣': 5000, '자바캣': 3500, '파이캣': 15000, '썬캣': 7000}
```

Python

```
#[Out]

In 가격을 입력하세요 : 10000
In 회원명을 입력하세요 : 파이캣
Out 적립금을 사용할 수 있습니다. 현재 적립금액은 15000노드 입니다.
In 사용할 적립금 노드를 입력하세요 : 5000
Out 5000노드를 사용하였습니다. 남은 적립금은 10000노드입니다. 결제금액은 5000원 입니다.

---

가격을 입력하세요 : 10000
회원명을 입력하세요 :썬캣
현재 적립금이 7000노드이므로 아직 적립금을 사용할 수 없습니다. 결제금액은 10000노드이고, 적립 포인트는 1000노드, 합산 포인트는 8000 노드입니다.
```

Python

4. **적립금 이벤트를 진행합니다!(심화, google에서 Factory 함수라고 검색해보세요.)**

   캣은 매달 월과 같은 요일에 적립금 이벤트를 진행하기로 했습니다.
   예를들어, 2월 2일에는 적립금의 2배를, 3월 3일에는 적립금의 3배를 적립해줍니다.

   만약 손님이 2월 2일에 5000원의 생선을 구매하고, 3월 3일에 15000원을 구매 했다면 각각 얼마의 적립금을 받을 수 있을까요? 중첩 함수를 사용하여 풀어보세요!

```python
def 배수(n):
    def 적립(value):
        '정답을 입력하세요.'
    return 적립

Feb = 배수(2)
Mar = 배수(3)

print('2월 적립금 이벤트 :', '정답을 입력하세요.')
print('3월 적립금 이벤트 :', '정답을 입력하세요.')
```

Python

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

5편 라이캣의 EXIT

1. 캣의 로봇(for)
    1.1 for문의 기본 구조
    1.2 for, else
    1.3 지능형 리스트(list comprehension)의 for
    1.4 다중인자 리스트 순회
    1.5 enumerate
    1.6 계산하는 로봇
2. 캣의 로봇(while)
    2.1 무한반복 while, break
    2.2 while, else
3. break
    3.1 break 문
4. continue, pass
5. else
6. 중첩 반복문

1. 캣의 로봇(for)

캣의 생선회사는 어느덧 위니브 월드 전체에서 가장 빠르게 성장하는 생선회사가 되었습니다. 대부분의 스타트업이 그렇듯이 성장세에 일만하다 보니 높은 연봉에도 직원들의 불만이 늘어가고 있었습니다.

"자율성과 창의적 생각이 보장되고, 개인이 성장하면서, 각자 하는 일에 대한 명료한 목적과 동기부여를 줄 수는 없을까냥?"

캣은 어느새 대표다운 생각을 하고 있었어요. 매주 아침 하는 회의에서 캣은 말했습니다.

"생산성을 극대화 하면서도, 직원들의 휴식시간을 늘려줄 수 있는 로봇을 만들어보는 것은 어떨까냥?"

하지만 이미 불만이 가득차 있는 직원들은 냉담한 반응이었습니다.

"누가 만드냥!? 대표가 만드냥!?"

캣은 더이상의 회의가 무의미하다는 생각에 회의를 종료했습니다. 그리고, 정말 혼자 만들기 시작했어요.

"밤을 새서라도 다음주까지 만들겠다냥!"

- **라이캣의 상태창!**

```
#[In]

이름 = '캣'
설명 = '위니브 월드에서 가장 급성장하는 생선회사 대표 캣'
나이 = 19
오늘_잡은_물고기 = '100000'
직원수 = 4
키 = '45.9cm'
몸무게 = 1.6
잡식 = True
돈 = 1100000
훈장 = ['백상아리를 잡은 고양이', '성실한 납세자', '해골섬 낚시꾼', '청년 고용 착한 기업', '회사를 설립한 자']
기술 = ['고기잡이', '고기팔기', '낚시_Lv5', '통발_Lv5', '큰그물_Lv5']
```

Python

캣은 가용한 모든 자원, 알고 있는 모든 지식을 투입하기 시작했어요. 아래 코드를 봅시다.

```
#[In]

for x in (1,2,3,4,5):
  print(f'고등어 포장 {x}번째 입니다.')
```

Python

우선 직원들이 가장 힘들어하는 고등어 포장 로봇을 만들기 시작했어요. `for` 라는 강력한 프로그래밍 문법을 사용하였습니다.

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고등어 포장 1번째 입니다.
고등어 포장 2번째 입니다.
고등어 포장 3번째 입니다.
고등어 포장 4번째 입니다.
고등어 포장 5번째 입니다.
```

Python

위 코드처럼 **정해진 횟수를 반복하는 것을 for문**이라고 합니다. 코드를 조금 더 살펴보죠. `x`라는 변수를 선언하여 튜플 `(1,2,3,4,5)`의 길이만큼 for문 아래에 있는 문장을 반복하게 됩니다.

또, 변수 x 는 튜플 요소의 값을 하나씩 가지게 되죠. 조금 어렵죠? 더 자세한 설명은 다음 챕터에서 하도록 하겠습니다.

1.1 for문의 기본 구조

위에서 배운 for문의 기본 구조를 더 자세하게 정리하고 넘어가도록 하겠습니다. for문은 순서열을 순회하며 순서열의 끝에 도달하면 반복을 멈추게 됩니다. 또한 객체를 처음부터 끝까지 하나씩 **추출하며 순회**하기 때문에 그 사용법이 쉬워 가장 많이 사용되는 반복문 입니다.

다음 챕터에서 배울 while은 비교할 변수를 먼저 선언 해주어야 하기 때문에, 비교적 for를 더 많이 사용합니다. 하지만 각자의 용법이 있어, 어느 문법이 좋다고는 할 수 없습니다!

```python
#[In]

for x in (1,2,3):
  print(f'{x}번째 로봇을 뚝딱뚝딱')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

1번째 로봇을 뚝딱뚝딱
2번째 로봇을 뚝딱뚝딱
3번째 로봇을 뚝딱뚝딱
```

Python

위의 코드를 실행하면 변수 x에는 튜플(1, 2, 3)의 요소가 순서대로 출력됩니다. 위 예제에서 튜플의 길이는 3이므로 for문은 반복을 3번 수행합니다. 여기서 x는 다른 언어처럼 초기화 하지 않아도 작동합니다.

```python
#for 문의 구조
for (변수명) in (순회 가능한 객체) :
    수행할 문장1
    수행할 문장2
```

Python

for 문의 범위로 사용되는 것은 시퀀스 자료형 자료 또는 반복 가능한 자료형이어야 합니다.

- **반복문 순서도**(Flow Chart) - for 문

1. String(문자열)을 범위로 지정한 예시

```python
#[In]

왕국 = '위니브 월드'
for a in 왕국 :
    print(a)
```

Python

```
#[Out]

위
니
브

월
드
```

Python

2. List(리스트)를 범위로 지정한 예시

```
#[In]

기술 = ['고기잡기', '고기팔기', '낚시_Lv1', '통발_Lv1', '큰그물_Lv1']
for a in 기술:
    print(a)
```

Python

```
#[Out]

고기잡기
고기팔기
낚시_Lv1
통발_Lv1
큰그물_Lv1
```

Python

3. Dictionary(사전)을 범위로 지정한 예시

```python
#[In]

캣의_상태창 = {
    '이름': '캣',
    '설명':'위니브 월드에서 가장 급성장하는 생선회사 대표 캣',
    '나이':'19',
    '키': '45.9cm',
    '몸무게': 1.6
}

for a in 캣의_상태창:
    print(a)
```

Python

```
#[Out]

이름
설명
나이
키
몸무게
```

Python

위의 예제에서 Dictionary(사전)형 자료의 경우에는 key(키)만을 가져오게 됩니다.

key(키)에 해당하는 value(값) 또한 가져오고 싶다면 아래와 같이 튜플 언패킹을 사용할 수 있습니다.

```python
#[In]

for key,value in 캣의_상태창.items():
    print("{0} : {1}".format(key, value))
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

이름 : 캣
설명 : 위니브 월드에서 가장 급성장하는 생선회사 대표 캣
나이 : 19
키 : 45.9cm
몸무게 : 1.6
```

Python

for문에서 가장 많이 사용되는 range에 대해 알아봅시다. range는 특정 범위를 생성하기 위해 사용할 수 있습니다.

> 💡 range는 2.x에서는 선언하는 즉시 list가 되었으나 3.x으로 넘어오면서 range는 list로 변환해 주어야 list가 됩니다. 이는 2.x에서 썼었던 xrange와 같은 함수입니다.

range 함수는 연속하는 수열을 만듭니다. `range(시작_값, 종료_값, 연속하는_두_수의_차)`형식으로 선언되며 아래 예제를 보며 자세히 설명해 드리도록 하겠습니다.

> 💡 range(start, stop, step)으로 주요 표현합니다. 어디서 많이 보았던 형식이죠? 바로 슬라이싱에서 보았던 형식입니다.

시작 값 : 0, 종료 값 : 5, 연속하는 두 수의 차 : 1

```
#[In]

for a in range(0, 5, 1):
    print(a)
```

Python

```
#[Out]

0
1
2
3
4
```

Python

시작값 0과 멈춤값 5 사이의 연속하는 두 수의 차 1을 가지고 있는 요소들이 생성되었습니다. 생성된 range의 마지막 요소가 멈춤값보다 한 단계 작다는 사실에 주의하세요.

시작 값 : 0, 종료 값 : 10, 연속하는 두 수의 차 : 2

```python
#[In]

for a in range(0, 10, 2):
    print(a)
```

Python

```
#[Out]

0
2
4
6
8
```

Python

시작 값 0과 종료 값 10 사이의 연속하는 두 수의 차 2를 가지고 있는 요소들이 생성되었습니다.

생성된 range의 마지막 요소가 종료 값보다 한 단계 작다는 사실에 주의하세요. range()함수의 마지막 매개변수인 연속하는 두 수의 차는 생략할 수 있습니다. 생략할 시 연속하는 두수의 차는 1입니다.

시작 값 : 0, 종료 값 : 5, 연속하는 두수의 차 : 생략

```python
#[In]

for a in range(0, 5):
    print(a)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

0
1
2
3
4
```

Python

마지막 매개변수인 연속하는 두 수의 차를 생략하고 range()함수를 호출하였습니다. 이 경우 range() 함수의 마지막 매개변수인 연속하는 두 수의 차는 1이 됩니다.

시작 값 : 생략, 종료 값 : 5, 연속하는 두 수의 차 : 생략

```
#[In]

for a in range(5):
    print(a)
```

Python

```
#[Out]

0
1
2
3
4
```

Python

또한 range() 함수는 종료 값만을 입력하여 호출할 수 있습니다. 종료 값만 입력했을 시에는 시작값은 0이 되며 두 수의 차는 1이 됩니다.

시작 값 : 10, 종료 값 : 5, 연속하는 두 수의 차 : -1

```
#[In]

for a in range(10, 5, -1):
    print(a)
```

Python

```
#[Out]

10
9
8
7
6
```

Python

연속하는 두 수의 차를 마이너스로 둘 수도 있습니다. 이 경우 시작 값과 종료 값 보다 크게 주셔야 한다는 점도 기억해주세요.

좀 더 알아볼까요? 얕은물에서 할 내용은 아니니 가볍게 읽고 넘어가주세요. for 문에 사용될 수 있는 시퀀스형 자료는 문자열, 리스트, 튜플이 있습니다. 이 자료형은 메서드로 __iter__와 __next__를 가지고 있으며 이 두개의 메서드로 순회가 가능하게 합니다.

이는 아래와 같이 iter()함수와 next()함수로도 실행을 할 수 있습니다.

```python
#[In]

listx= [100,200,300,400]
strx= 'abcd'
listxlter = iter(listx)
strxlter= iter(strx)
print(next(listxlter),next(listxlter),next(listxlter),next(listxlter))
print(next(strxlter),next(strxlter),next(strxlter),next(strxlter))
```

Python

위 결과값은 아래와 같습니다.

```
#[Out]

100 200 300 400
a b c d
```

Python

1.2 for, else

```python
#[In]

for i in range(100):
    print(f'{i} 물고기를 잡았습니다.')
    if i == 5:
        print('만선입니다. 물고기를 다 잡았습니다.')
        break
else:
    print('아직 여유가 좀 있지만, 물고기가 더 없는 것 같으니 이정도로 만족하고 돌아갑시다.')
print('수고하셨습니다.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

0마리의 물고기를 잡았습니다.
1마리의 물고기를 잡았습니다.
2마리의 물고기를 잡았습니다.
3마리의 물고기를 잡았습니다.
4마리의 물고기를 잡았습니다.
5마리의 물고기를 잡았습니다.
만선입니다. 물고기를 다 잡았습니다.
```

Python

```
#[In]

for i in range(5):
    print(f'{i} 물고기를 잡았습니다.')
    if i == 5:
        print('만선입니다. 물고기를 다 잡았습니다.')
        break
else:
    print('아직 여유가 좀 있지만, 물고기가 더 없는 것 같으니 이정도로 만족하고 돌아갑시다.')
print('수고하셨습니다.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

0마리의 물고기를 잡았습니다.
1마리의 물고기를 잡았습니다.
2마리의 물고기를 잡았습니다.
3마리의 물고기를 잡았습니다.
4마리의 물고기를 잡았습니다.
아직 여유가 좀 있지만, 물고기가 더 없는 것 같으니 이정도로 만족하고 돌아갑시다.
수고하셨습니다.
```

Python

if문 뿐만 아니라 for에서도 else를 사용할 수 있습니다. else는 루프가 정상 종료되었을 때, 처음부터 자료형이 비어있었을 때 실행됩니다. break문을 만나면 else 문을 실행하지 않고 빠져나옵니다.

1.3 지능형 리스트(list comprehension)

얕은물에서는 가볍게 이런 것이 있다는 것만 알고 넘어가도록 하겠습니다. 가장 많이 사용되는 list 생성 기법입니다. 'list comprehension'은 한국어로 표현할 때 리스트 표현식, 지능형 리스트로 번역이 되곤 합니다.

```python
#[In]

#1
x= [i for i in range(1, 10)]
print(x)

#2
y=['{}X{}={}'.format(i, j, i*j) for i in range(2, 10) for j in range(1, 10)]
print(y)

#3
def sumthingFunction(i):
    if i % 100 ==0:
        return i
    else:
        return 0
기존리스트= [100,200,300,101,202,303]
새로운리스트= [sumthingFunction(i) for i in 기존리스트]
print(sum(새로운리스트))
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

#1
[1,2,3,4,5,6,7,8,9]

#2
구구단 리스트 출력

#3
600
```

Python

1.4 다중인자 리스트 순회

```python
#[In]

#다중 리스트 for문
skill = [
        ('고기잡이', 100),
        ('고기팔기', 120),
        ('낚시', 5),
        ('통발', 5),
        ('큰그물', 5)
]

for i,j in skill:
    print(i,j)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고기잡이 100
고기팔기 120
낚시 5
통발 5
큰그물 5
```

Python

```python
#[In]

#하나만 리스트여도 쌍이면 상관없이 잘 돌아감
skill = [
        ('고기잡이', 100, 'SS'),
        ('고기팔기', 120, 'SSS'),
        ('낚시', 5, 'C'),
        ('통발', 5, 'C'),
        ('큰그물', 5, 'C')
]

for skillName, skillLevel, skillGrade in skill:
    print(skillName, skillLevel, skillGrade)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고기잡이 100 SS
고기팔기 120 SSS
낚시 5 C
통발 5 C
큰그물 5 C
```

Python

1.5 enumerate

enumerate는 순서를 매길 때 사용합니다. 각각의 스킬들을 습득한 순서대로 출력하고 싶어요. 그럼 어떻게 출력을 해줄 수 있을까요? enumerate는 별도의 변수를 선언하지 않고 이것이 가능하게 해줍니다.

```python
#[In]

# enumerate

skill = [
        ('고기잡이', 100, 'SS'),
        ('고기팔기', 120, 'SSS'),
        ('낚시', 5, 'C'),
        ('통발', 5, 'C'),
        ('큰그물', 5, 'C')
]

for i, (skillName, skillLevel, skillGrade) in enumerate(skill):
    print(i, skillName, skillLevel, skillGrade)

for i, j in enumerate(skill, 100):
    print(i, j)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

0 고기잡이 100 SS
1 고기팔기 120 SSS
2 낚시 5 C
3 통발 5 C
4 큰그물 5 C
100 ('고기잡이', 100, 'SS')
101 ('고기팔기', 120, 'SSS')
102 ('낚시', 5, 'C')
103 ('통발', 5, 'C')
104 ('큰그물', 5, 'C')
```

Python

1.6 계산하는 로봇

캣은 로봇을 이용해서 '고등어 포장' 보다 조금 더 어려운 '카운터 보기'를 맡기기로 했어요. 손님께 생선 몇 개를 구매하실 건지 여쭤보고 그 갯수에 맞는 금액을 출력할 수 있는 로봇을 만들었어요.

```
#[In]

for i in range(5):
  num = int(input("고등어 몇 개를 구매하실 건가요?"))
  result = num * 5000
  print(result,"원 입니다.")

print("로봇이 종료되었습니다. 삐빅.")
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고등어 몇 개를 구매하실 건가요? 5
25000원 입니다.
고등어 몇 개를 구매하실 건가요? 3
15000원 입니다.
고등어 몇 개를 구매하실 건가요? 19
95000원 입니다.
고등어 몇 개를 구매하실 건가요? 1
5000원 입니다.
고등어 몇 개를 구매하실 건가요? 7
35000원 입니다.
로봇이 종료되었습니다. 삐빅.
```

Python

고등어를 몇 개 구매할 건지 input() 명령어를 통해 손님께 여쭤보고, 그 갯수에 가격을 곱해서 출력을 해주도록 만들었습니다.

---

2. 캣의 로봇(while)

금방 배웠던 for문을 이용해서 로봇을 만들면 정해진 숫자에서 종료되기 때문에, 생선이 더 있거나 손님이 더 있어도 로봇이 꺼지기 때문에 로봇을 다시 시작해줘야 한다는 단점이 있었어요.

그래서 캣은 while문을 활용하여 영업이 종료되거나 생선이 다 팔릴 때까지 작동을 하는 로봇을 만들기로 했습니다.

우선 간단하게 while 문을 살펴보도록 해요.

```
#[In]

a = 1
while a < 10 :
  print(a)
  a += 1
```

Python

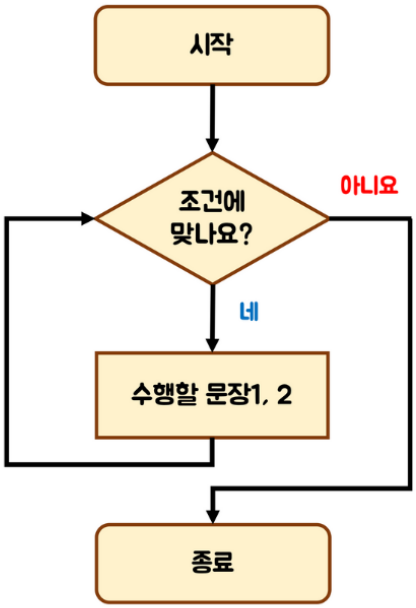
while문은 조건이 참인 동안에 명령을 반복해서 수행합니다. 반복할 명령은 들여쓰기로 구분되며 조건이 거짓이면 들여쓰기로 구분되어 있는 반복 구문을 탈출합니다.

위의 예제에서 9를 출력하고 마지막 a+=1이 연산(할당연산)되면 a=10이 되어 조건이 거짓이 되므로 루프를 탈출 합니다. 다음 a의 값이 10이 된다는 것 잊지 마세요.

```
while문의 구조

while 조건문 :
  수행할 문장1
  수행할 문장2
```

Python

- **반복문 순서도**(Flow Chart) - while 문



2.1 무한반복 while, break

그럼 이제 생선이 다 팔리면 로봇이 종료되도록 만들어볼까요? 현재 고등어는 5개가 남았고, 고등어가 0개가 된다면 다 팔았다는 말과 함께 while문을 종료하고 싶어요.

```python
#[In]

fish = 5
while True:
  print("고등어 ", fish, "개 남았습니다.")
  fish -= 1
  if fish == 0:
    print("고등어 다 팔렸습니다.")
    break
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```python
#[Out]

고등어  5 개 남았습니다.
고등어  4 개 남았습니다.
고등어  3 개 남았습니다.
고등어  2 개 남았습니다.
고등어  1 개 남았습니다.
고등어 다 팔렸습니다.
```

Python

while 문의 조건을 부분에 bool형인 True가 오게 하면 반복문은 무한 반복되게 됩니다. 무한 반복이 일어나면 프로그램이나 서비스가 죽는 사태가 발생되기 때문에 조건을 탈출할 수 있는 구문을 중간에 입력하거나 메모리 공간을 확보할 쉬는 시간을 주어야 합니다. 여기서는 break 문으로 조건을 탈출하도록 설계하였습니다.

2.2 while, else

위 예시와 똑같은 문장을 else문으로도 만들어 볼 수 있어요. 여기서 주석 처리된 부분을 풀어보시고 주석이 될 때와 되지 않을 때를 비교해보세요.

Type '/' for commands

```python
#[In]

fish = 5
while fish > 0:
    print("고등어 ", fish, "개 남았습니다.")
    # if fish < 3:
    #     break
    fish -= 1
else:
    print('고등어 다 팔렸습니다.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고등어  5 개 남았습니다.
고등어  4 개 남았습니다.
고등어  3 개 남았습니다.
고등어  2 개 남았습니다.
고등어  1 개 남았습니다.
고등어 다 팔렸습니다.
```

Python

else문은 다양하게 활용(특히 if문에서)됩니다. while문에서 else는 루프가 정상 종료되었을 때와 처음부터 while의 조건문이 False일 경우 실행됩니다. 위 두번째 예제에서 a==5인 경우 break문을 만나면 else 문을 실행하지 않고 빠져나옵니다.

3. break

break는 흐름을 끊어 중단할 때 사용합니다. 특정 코드를 반복하고자 for, while 문을 이용하여 반복문(loop)을 만들었다면 break 문을 사용하여 반복문을 중단할 수 있습니다. **여기서 주의하셔야 할 점은 이 break 구문은 바로 위의 for나 while문만 탈출한다는 것입니다!**

반복문(loop) 순회 중에 break 문을 만나면 반복문의 내부 블록을 벗어나게 됩니다. 그러나 예제 1번과 같이 어떠한 장치 없이 break를 넣으면 반복문이 반복하지 않으므로 예제 2번과 같이 주로 조건 안에 넣어 실행합니다.

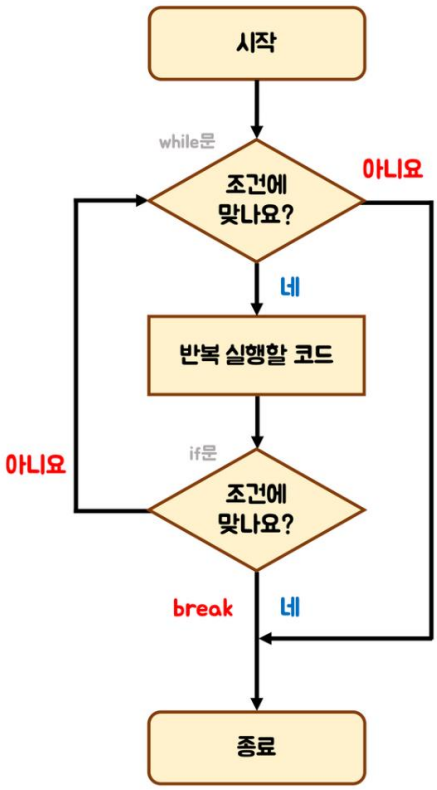
```python
# 예제1(while)
while 조건:
  반복 실행할 코드
  break # while 구문을 탈출

# 예제1(for)
for 변수 in 범위:
  break # for 반복문 탈출
```

Python

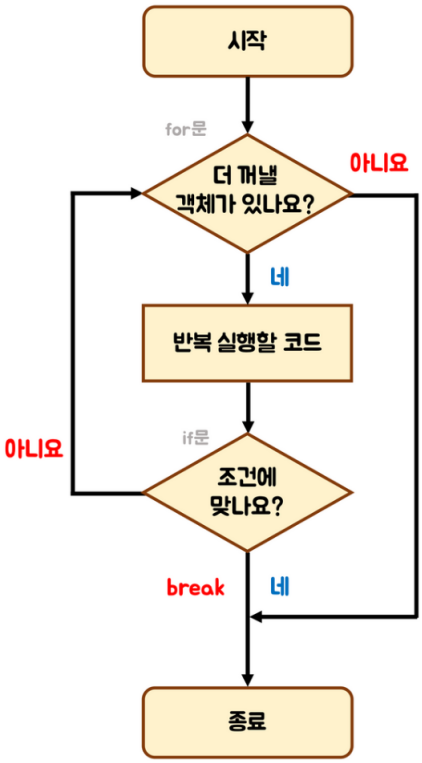
```python
# 예제2(while)
while 조건:
  반복 실행할 코드
  if 조건:
    break # while 구문을 탈출

# 예제2(for)
for 변수 in 범위:
  반복 실행할 코드
  if 조건 :
    break # for 반복문 탈출
```

Python

- 순서도(Flow Chart) - while, break 문



- 순서도(Flow Chart) - for , break 문



3.1 break 문

```python
#[In]

시간 = 0
현재시간 = int(input("시(hour)를 입력해주세요. "))
종료시간 = 18
if 현재시간 < 종료시간:
    print('영업시간이 끝났기 때문에 영업을 종료합니다.')
else:
    while True:
        시간 += 1
        현재시간 += 1
        print('{}시간이 지나 {}시가 되었습니다'.format(시간,현재시간))
        if 현재시간 == 종료시간:
            print('영업 종료시간이 되었기 때문에 영업을 종료합니다.')
            break
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

시(hour)를 입력해주세요. 10
1시간이 지나 11시가 되었습니다
2시간이 지나 12시가 되었습니다
3시간이 지나 13시가 되었습니다
4시간이 지나 14시가 되었습니다
5시간이 지나 15시가 되었습니다
6시간이 지나 16시가 되었습니다
7시간이 지나 17시가 되었습니다
8시간이 지나 18시가 되었습니다
영업 종료시간이 되었기 때문에 영업을 종료합니다.
```

Python

반복문이 실행되다 시간이 지나 현재 시간과 종료 시간이 같아지는 경우 영업을 종료한다는 메시지를 출력 후 break 문을 통하여 반복문을 빠져나갑니다.

반복문을 빠져나가자 반복문이 중단되어 더 이상 실행되지 않으므로 시간이 지나가지 않습니다.

앞서 말씀드린 것처럼, 중첩 반복문에서 바로 위에 반복문만 탈출하는 경우를 살펴보도록 하겠습니다. 아래의 경우 각 단의 `i X 5` 까지만 실행이 됩니다.

```
#[In]

for i in range(2, 10):
    for j in range(1, 10):
        if j > 5:
            break
        print(f'{i} X {j} = {i*j}')
```

Python

4. continue, pass

continue의 사전적 의미를 살펴보면 '**계속하다**'라는 뜻이 있습니다. 파이썬에서 continue문은 **반복문이 실행하는 코드 블록의 나머지 부분을 실행하지 않고 다음 반복으로 건너가게 흐름을 조정**합니다.

pass의 사전적 의미는 '**지나치다**'라는 뜻이며, 파이썬에서 pass문은 단순히 실행할 코드가 없다는 것을 의미하며 **아무런 동작을 하지 않고 다음 코드를 실행**합니다.

continue와 pass는 구분해서 사용을 해야 하니 차이점을 잘 정리해 두세요.

반복문 순회 도중 continue문을 만날 시 continue문은 이후의 나머지 반복문 내부 코드 블록을 실행하지 않고 다음 아이템을 선택하여 반복문의 내부 코드 블록 시작 부분으로 이동합니다.

break와 마찬가지로 예제1과 같이 조건이 없이 continue를 사용하는 것은 큰 의미가 없으므로 주로 예제2처럼 조건문을 달아 사용합니다.

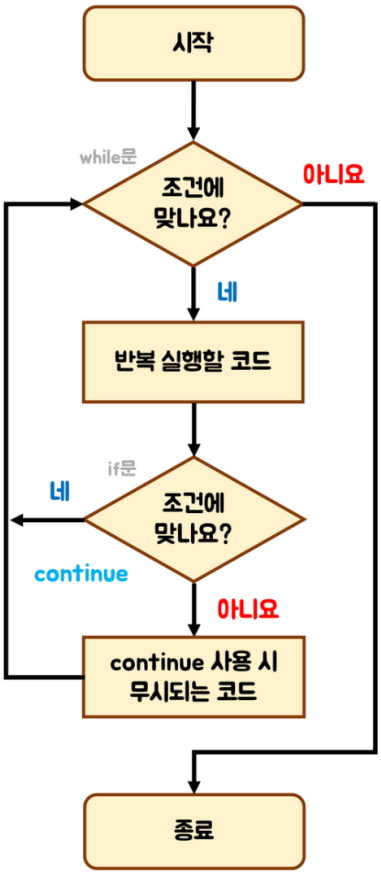
```python
# 예제1(while)
while 조건:
    반복 실행할 코드
    continue #while 다음 반복문 수행
    반복 실행할 코드 # continue 사용시 무시되는 코드

# 예제2(for)
for 변수 in 범위:
    반복 실행할 코드
    continue #for 다음 반복문 수행
    반복 실행할 코드 # continue 사용시 무시되는 코드
```

Python

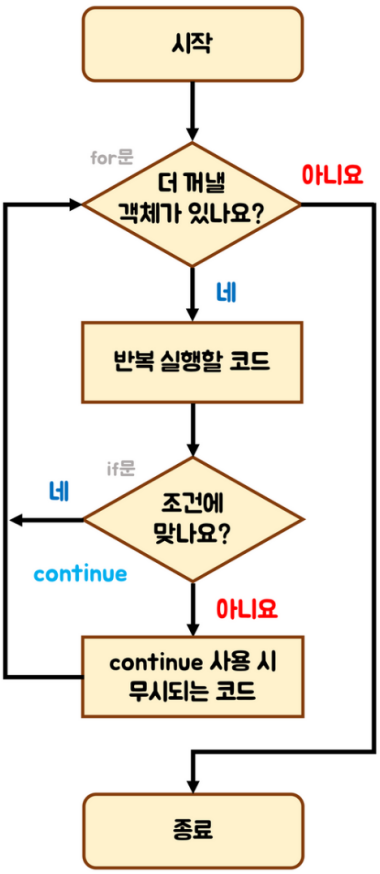
```python
# 예제1(while)
while 조건:
    반복 실행할 코드
    if 조건 :
        continue #while 다음 반복문 수행
    반복 실행할 코드 # continue 사용시 무시되는 코드

# 예제2(for)
for 변수 in 범위:
    반복 실행할 코드
    if 조건:
        continue # for 다음 반복문 수행
    반복 실행할 코드 # continue 사용시 무시되는 코드
```

Python

- 순서도(Flow Chart) - while, continue 문



- 순서도(Flow Chart) - for, continue 문




```python
#[In]

#continue
구매개수_총가격= [
    (3, 15000),
    (5, 25000),
    (1, 5000),
    (8, 40000),
    (0, 0),
    (2, 10000)
]

for 구매개수, 총가격 in 구매개수_총가격:
    if 구매개수 < 1:
        continue
    print('구매개수는 {}개이며 {}원입니다.'.format(구매개수, 총가격))
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

구매개수는 3개이며 15000원입니다.
구매개수는 5개이며 25000원입니다.
구매개수는 1개이며 5000원입니다.
구매개수는 8개이며 40000원입니다.
구매개수는 2개이며 10000원입니다.
```

Python

1개도 구매하지 않았을 경우 if의 조건이 참(True)이 되어 continue문을 실행합니다.

따라서 나머지 코드블록인 구매개수와 가격을 출력해주는 print문을 실행하지않고 다시 **for 반복문의 내부 코드블록 시작부분**으로 돌아갑니다.

여기서 continue라고 되어 있는 부분을 pass로 바꾸면 수험번호 1005번의 58점도 출력하는 것을 볼 수 있습니다. 반복 중간에 continue를 만나면 다음 반복으로 넘어가지만 pass를 만나면 그 라인만 지나치고 아래 문장을 그대로 실행합니다.

```
#[In]

#pass
구매개수_총가격= [
    (3, 15000),
    (5, 25000),
    (1, 5000),
    (8, 40000),
    (0, 0),
    (2, 10000)
    ]
for 구매개수, 총가격 in 구매개수_총가격:
    if 구매개수 < 1:
        pass
    print('구매개수는 {}개이며 {}원입니다.'.format(구매개수, 총가격))
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

구매개수는 3개이며 15000원입니다.
구매개수는 5개이며 25000원입니다.
구매개수는 1개이며 5000원입니다.
구매개수는 8개이며 40000원입니다.
구매개수는 0개이며 0원입니다.
구매개수는 2개이며 10000원입니다.
```

Python

5. else

앞서 살펴본 것처럼 Python에서는 while, for 문에서도 else문을 사용할 수 있습니다. 여기서의 else는 if에서의 else처럼 '그렇지 않으면'이라는 의미 보다는 '**그런 다음**'이라는 의미가 더 강하기 때문에 than으로 쓰여야 된다는 논의가 있기도 했습니다.

- **else문 순서도**(Flow Chart) - while문

- else문 순서도(Flow Chart) -for문

반복문이 break 등에 의해 중단없이 정상적으로 반복이 종료된 후 특정 코드를 실행하게 해야할 때 while~else, for~else를 사용할 수 있습니다.

```
while 조건:
  반복 실행할 코드
else:
  while 반복문이 모두 실행되어 종료되고 실행할 코드
```

Python

```
for 변수 in 범위:
  반복 실행할 코드
else:
  for 반복문이 모두 실행되어 종료되고 실행할 코드
```

Python

```
#[In]

#for, else문
for i in range(5, 0, -1):
    print("고등어 ", i, " 개 남았습니다.")
else:
    print('고등어 다 팔렸습니다.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

고등어 5개 남았습니다.
고등어 4개 남았습니다.
고등어 3개 남았습니다.
고등어 2개 남았습니다.
고등어 1개 남았습니다.
고등어 다 팔렸습니다.
```

Python

```python
#[In]

#break문이 있을 때
for i in range(5, 0, -1):
    print("고등어 ", i, " 개 남았습니다.")
    if i == 1:
        break
else:
    print('고등어 다 팔렸습니다.')
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```python
#[Out]

고등어 5개 남았습니다.
고등어 4개 남았습니다.
고등어 3개 남았습니다.
고등어 2개 남았습니다.
고등어 1개 남았습니다.
```

Python

위 코드에서 1번 예제처럼 break문 없이 정상 종료되었을 때에는 else문이 실행됩니다. 그러나 2번 예제처럼 break문이 있을 경우에는 else문을 실행하지 않게 됩니다.

if문에서 else는 '그렇지 않으면'이라는 의미로 조건이 거짓일 때 쓰입니다.

```python
if 조건:
  조건이 참일 경우 실행문
else:
  조건이 거짓 일 경우 실행문
```

Python

```python
#[In]

i = 2
if i < 4:
    print('i는 4보다 작습니다')
else:
    print("i는 4보다 크거나 같습니다")
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

i는 4보다 작습니다
```

Python

try~except~else문에서 else는 예외가 발생하지 않을 때 쓰입니다.

```
try:
  실행문
except:
  예외 발생 시 처리문
else:
  예외 발생하지 않을 경우 실행문
```

Python

```
#[In]

try:
    i = 1
    j = 1
    x = i/j
except:
    print("error")
else:
    print(x)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

1.0
```

Python

6. 중첩 반복문

반복문 내부에 또 다른 반복문이 있을 경우, **중첩되었다**고 얘기합니다. 중첩 반복문이란 반복문 안에 반복문이 들어가 중첩된 것을 얘기합니다. 중첩 반복문에서는 종료되는 값을 항상 확인하세요.

```python
#[In]

i = 2
while i < 10 :
    k = 1
    while k < 10:
        print(i, " * ", k, " = ", i * k)
        k += 1
    i += 1
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
3 * 1 = 3
  ...
  ...
  ...
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
```

Python

구구단은 2단부터 시작하기에 변수 i를 2로 설정해줍니다. 9단을 수행한 후 종료할 수 있도록 10미만이라는 조건을 설정해주었습니다. i는 구구단의 단을, k는 순차적으로 반복될 곱의 수이며 while문을 하나 더 이용하여 반복될 곱의 수를 1씩 순차적으로 더해주었습니다.

여기서 중요한 것은 **종료되는 값**입니다. k가 10이 된 상태로 안에 반복문이 종료되기 때문에 3단을 하기 위해서는 **다시 k를 1로 초기화 시켜주어야 합니다.**

for문과 range 함수를 이용하면 좀 더 간단하게 구구단을 출력할 수 있습니다.

```python
#[In]

for i in range(2,10):
    print("---{0}단---".format(i))
    for k in range(1,10):
        print(i,"*",k,"=",i*k)
```

Python

위 코드를 실행시키면 아래와 같이 출력됩니다.

```
#[Out]

---2단---
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
---3단---
3 * 1 = 3
3 * 2 = 6
  ...
  ...
  중략
```

Python

for문을 중첩 사용하면 좀 더 간결하고 직관적인 코드를 작성할 수 있습니다.

첫 번째 for문에서 range 함수를 이용하여 2단부터 10단까지 차례로 i에 대입됩니다. 두 번째 for문에서 곱해지는 수를 k에 넣고 구구단을 출력하고 있습니다.

while과 달리 k의 값이 for문 안에서 자동으로 초기화가 되기 때문에 while문처럼 k=1을 두 번째 for문 위에 넣을 필요는 없습니다.

▶ **퀴즈!**

생선의 종류는 고등어, 연어, 숭어가 있습니다. 고등어는 5천원, 연어는 8천원, 숭어는 3천원입니다. 모든 손님들은 3가지 종류의 생선을 모두 구매할 예정입니다.
5명의 손님께 구매할 생선들의 **갯수**를 각각 물어본 뒤, 맨 마지막엔 계산한 **총 금액**을 출력해주세요.
(문제가 어려우니 반드시 여러번 생각한 후 풀어보세요. - 힌트 : list를 사용해보세요)

```
#퀴즈 - 출력해야 하는 값 입니다.

고등어 몇 개를 구매하실 건가요?3
연어 몇 개를 구매하실 건가요?4
숭어 몇 개를 구매하실 건가요?2
53000 원 입니다.
고등어 몇 개를 구매하실 건가요?1
연어 몇 개를 구매하실 건가요?2
숭어 몇 개를 구매하실 건가요?3
83000 원 입니다.
...
중략
```

Python

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

사자탈을 쓴 캣

PYTHON BOOTCAMP

기업에 대한 노하우가 쌓이고 곳곳에 배치할 로봇까지 만들어, 직원들에게 높은 임금과 자유를 주게 된 캣은 고민에 빠졌어요.

"이렇게 벌어 병원을 세우려면 1억 3천 299년이 걸린다냥.."

평민들만 있는 곳에서는 벌 수 있는 돈이 한계가 있었기 때문이죠. **그래서 캣은 모든 지분을 사회에 환원**하고 사자들만 들어갈 수 있는 **라이언 타운**에 들어가기로 결심합니다.

그러나 문제가 있었어요. 라이언 타운은 왕족과 귀족인 사자들만 들어갈 수 있었으며 고양이는 사업은 커녕 라이언 타운에 들어갈 수 조차 없었어요.

한참을 고민하던 캣은 아주 기발한 생각을 하게 되었습니다.

"사자탈을 쓰고 들어가면 된다냥!"

그렇게 아주 간단한 방법으로 라이언 타운에 들어갈 수 있을 줄 알았지만 생각보다 관문을 지키는 문지기들은 그리 어리숙하지 않았어요. 꼼꼼하고 살벌한 검문검색에 캣은 입장 시도 조차 못하고 고민에 다시 빠졌습니다.

그때, 지분을 주었던 한 직원이 찾아와 고맙다며 정보를 주었어요!

"**라이언 타운의 왕족이 은밀하게 출입을 하던 비밀통로**가 있는데 그곳에 오랜 문지기가 우리 할아버지였다냥! 그곳만 통과하면, 라이언 타운 안에서는 검문 검색을 안하니 그 옷을 입고 다닐 수 있을것이다냥!"

캣은 은밀한 곳에 숨겨진 비밀통로를 찾아갔습니다. 약속대로 할아버지 문지기는 없었어요. 그런데! 생각지도 못한 난제를 만났어요. 비밀통로를 지키는 '말하는 문'을 만난거죠!

"이 문은 왕의 혈통만이 지나갈 수 있는 길! 혈통을 검증하기 위한 문제를 맞춰야 문을 열어줄 수 있다!! 크르릉!!"

라이언 타운을 지키는 '말하는 문'은 문제를 내기 시작했어요.

```
1. 왕의 충성스런 신하는 아래와 같다. 왕의 신하가 총 몇 명인지 구하라.
신하 = {
  '하이에나' : 1121,
  '코뿔소' : 122,
  '코끼리' : 88,
  '기린' : 119,
  '독수리' : 62,
  '고양이' : 31,
}

2. 이웃 왕국의 침략이 있어 출전 준비해야 한다. 100명 이상인 종은 출전을! 100명 이하인 종은 왕국을 수호한다. 출전 신하는 몇 명이고, 왕국을 수호하는 신하는 몇 명인가?

3. 침략을 방어하고, 출전 했던 신하들은 각각 80%라는 큰 손실을 입었습니다. 왕국을 수호하는 신하까지 총 얼마의 신하가 남아 있나요?
```

Python

과연 캣은 문제를 풀고 라이언 타운에 들어갈 수 있을까요?

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

스토리 : 쏘아진 화살

드디어 라이언 타운 잠입에 성공한 캣! 캣은 도와줄 친구도, 동료도, 가족도, 친척도 없었지만 그동안 쌓아왔던 경험으로 라이언 타운에서도 군계일학이었어요.

최대한 많은 물고기를 잡아 잠입한 통로를 통해 물고기를 들여왔습니다. 또 신뢰를 쌓고, 동료를 모으고, 회사를 성장시키기까지 이미 많은 경험을 통해 쌓은 캣의 기술로 마치 하늘에 쏘아진 화살처럼 회사를 성장시켰습니다.

"이제 병원을 세워야 겠다냥!"

캣은 처음 그의 뜻대로 병원을 세우기 위해 발품을 팔았습니다. 부지를 매입하고, 의사를 설득하고, 건축 설계, 법인 설립, 병원 등록까지 막히는 것이 없이 처리해 나갔어요. 그러나 문제는 전혀 생각치 못한 곳에서 터져나왔습니다.

"평민도 들어올 수 있는 병원이라니!? 그런 병원은 허락할 수 없흥! 크르릉!"

귀족들의 반발이 만만치 않았던 것이죠. 캣은 오랫동안 그들을 설득했습니다. 정치적 힘을 얻기 위해 어쩔 수 없이 정계에도 발을 들여놓았어요.

"차별은 투텁고도 무섭다냥, 파괴적 혁신이 필요하다냥."

캣은 캣의 뜻대로 병원을 설립하고, 그곳에서 평민을 받을 수 있게 되었지만, 그가 꿈꾸던 보육원, 교육 시설과 같은 차별없이 사는 사회는 멀게만 느껴졌습니다.

"방법을 찾아야 한다냥!"

그는 어느날, 그동안 쌓았던 모든 것을 내려놓고 홀연히 사라집니다.

6편 : 파이와 썬의 알고리즘 7원석을 찾아서

1. 라이캣의 모험
2. 클래스
    2.1 클래스 변수와 인스턴스 변수
    2.2 _init_함수
    2.3 상속
    2.4 다중 상속
    2.5 특별 메소드(magic method)
    2.6 캣의 준비물

1. 라이캣의 모험

위니브 월드에서는 '알고리즘 7원석'에 대한 이야기로 떠들썩합니다. 알고리즘의 제왕 '파이'와 '썬'이 어딘가에 숨겨둔 알고리즘 7원석을 찾으면 소원을 이뤄준다는 소문이었어요. 수행을 다니던 캣은 생각합니다.

"나는 왕이 되어야겠다냥!"

캣은 멀고 험한 길이 될 것임을 직감했습니다. 그동안 쌓았던 모든 지식이 무용지물이 될 수 있겠다는 생각을 하게 되었고, 새로운 준비를 결심했습니다.

"험한 길이 될거냥. 같으니 단단히 준비해야겠다냥!"

캣은 우선 자신이 지금까지 일구어 놓은 위니브 월드에 **회사들을 경영할 대리인**을 로봇으로 만들기로 합니다. 지금까지 했던 방식과는 다른 전혀 새로운 방식으로요.

- **라이캣의 상태창!**

```python
#[In]

이름 = '캣'
설명 = '위니브 월드에서 가장 성공한 사업가 라이캣, 현재 어디있는지 알 수 없음.'
나이 = 21
오늘_잡은_물고기 = '9999999'
직원수 = 124
키 = '46.1cm'
몸무게 = 1.8
잡식 = True
돈 = 9999999
정치 = 21
야망 = 2
훈장 = [
  '백상아리를 잡은 고양이',
  '성실한 납세자',
  '해골섬 낚시꾼',
  '청년 고용 착한 기업',
  '회사를 설립한 자',
  '혈통의 인정을 받은 자',
  '천민을 위한 병원을 세운 자'
]
기술 = {
  '회사운영': 99,
  '낚시' : 96,
  '통발' : 93,
  '큰그물' : 95,
  '재무' : 34,
}
```

Python

2. 클래스

클래스는 데이터와 기능을 가지고 있는 인스턴트 객체를 생성하기 위한 역할을 합니다.

파이썬은 대표적인 객체지향 프로그래밍 언어입니다. 클래스는 일종의 설계도면입니다. 파이썬은 이 설계도면을 보며 하나의 인스턴스 객체를 만들어 냅니다. 그리고 그 인스턴스 객체를 사용할 수 있게 됩니다.

어렵죠? 기억하세요. 지금 캣은 자신을 대리할 대리인을 만드는 것입니다. 따라서, **클래스**는 대리인을 찍어내는 **공장**! 그 공장에서 찍혀져 나오는 **인스턴스** 로봇이 바로 캣과 똑같은 생각을 하고 똑같이 행동하는 **대리인** 입니다.

```
#[In]

class 대리인(object):
  이름 = '라이캣의 대리인'
  훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')

  def 포장하기(self):
    print('포장이 완료되었습니다.')

대리인1 = 대리인()
대리인2 = 대리인()

대리인1.계산하기(5)
대리인1.포장하기()
대리인2.계산하기(10)
대리인2.포장하기()

print(대리인1.이름)
print(대리인1.훈장)

print(대리인2.이름)
print(대리인2.훈장)
```

```
#[Out]

25000 원 입니다.
포장이 완료되었습니다.
50000 원 입니다.
포장이 완료되었습니다.

라이캣의 대리인
['백상아리를 잡은 고양이', '성실한 납세자', '해골섬 낚시꾼', '청년 고용 착한 기업', '회사를 설립한 자', '혈통의 인정을 받은 자', '천민을 위한 병원을 세운 자']

라이캣의 대리인
['백상아리를 잡은 고양이', '성실한 납세자', '해골섬 낚시꾼', '청년 고용 착한 기업', '회사를 설립한 자', '혈통의 인정을 받은 자', '천민을 위한 병원을 세운 자']
```

Python

그런데 대리인이 이름도 같고 훈장도 같으니, 구분이 안되죠? 그래서 대리인 목록을 하나 만들도록 하겠습니다. 또한 이름이 모두 같으니 이름도 다르게 하겠습니다.

2.1 클래스 변수와 인스턴스 변수

```python
#[In]

class 대리인(object):
  # 클래스 변수 위치 (파이썬 규약에 따라 indent로 결정)
  이름 = '라이캣의 대리인'
  대리인_목록 = []
  훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')
```

```python
  def 포장하기(self):
    print('포장이 완료되었습니다.')

대리인1 = 대리인()
대리인.대리인_목록.append('대리인1')
대리인2 = 대리인()
대리인.대리인_목록.append('대리인2')


print(대리인.대리인_목록)
print(대리인1.대리인_목록)
print(대리인2.대리인_목록)
```

Python

```
#[Out]

['대리인1', '대리인2']
['대리인1', '대리인2']
['대리인1', '대리인2']
```

Python

클래스 변수는 다른 인스턴스들과 변수를 공유합니다. 이 변수는 메서드(클래스의 함수라고는 부르지 않습니다)의 위치와 동등한 들여쓰기 위치에 자리하고 있습니다.

이 클래스 변수는 위의 예시와 같이 인스턴스나 클래스 이름을 통해서 접근할 수 있습니다. 변수 이름으로 직접 접근을 하지 않는다는 사실을 주의해주세요!

이 클래스 변수는 해당 클래스를 통해 만들어진 모든 인스턴스 객체들이 공유하는 변수 값입니다.
각 인스턴스 객체들 각자가 관리하고 있는 변수는 인스턴스 변수라고 합니다.

위의 예시에서는 인스턴스를 생성할 때마다 어떤 인스턴스가 만들어졌는지 확인하기 위해 인스턴스를 생성할 때마다 `대리인.대리인_목록`에 접근하여 해당 리스트에 어떤 대리인이 추가되었는지를 표시하고 있습니다.

하지만 이렇게 처음 시작 시에 일일이 메서드를 호출해서 하는 것은 불편하고 실수가 생길 수 밖에 없는 작업입니다. 이런 불편을 해소하기 위해 `__init__` 메서드가 있습니다.

언더바가 두개인 이 메서드는 매직 메서드 또는 던더함수라고 합니다. 이것은 다음 장에서 알아보도록 하겠습니다.

이번 장에서는 인스턴스 변수를 통해, 이름을 변경해보죠!

```python
#[In]

class 대리인(object):
  이름 = '라이캣의 대리인'
  대리인_목록 = []
  훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def 이름_변경하기(self, name):
    self.이름 = name

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')

  def 포장하기(self):
    print('포장이 완료되었습니다.')

대리인1 = 대리인()
대리인.대리인_목록.append('대리인1')
대리인2 = 대리인()
대리인.대리인_목록.append('대리인2')

print(대리인1.이름)
print(대리인2.이름)

대리인1.이름_변경하기('라이캣하나')
대리인2.이름_변경하기('라이캣둘')

print(대리인1.이름)
print(대리인2.이름)
```

Python

```
#[Out]

라이캣의 대리인
라이캣의 대리인
라이캣하나
라이캣둘
```

Python

클래스 변수를 다룰 때는 클래스 자체(예시에서는 `대리인` 객체)를 이용하여 다루어야 하지만 위의 예시에서 이름을 변경할 때는 각 인스턴스 객체들이 각각 자신의 메서드를 통해 접근을 하고 있습니다. 여기서 self의 정체는 바로, 인스턴스의 영역을 얘기합니다.

앞서 `대리인 목록`을 모두가 공유했다면, **self의 영역은 다른 인스턴스에 해당 변수를 공유하지 않는 고유 영역**이라 볼 수 있습니다.

2.2 __init__ 함수

자, 이제 `__init__`을 사용해서, 이름 변경하기를 별도의 메서드가 아닌, 인스턴스가 생성될 때 호출이 되게 해주도록 하겠습니다.

```python
#[In]

class 대리인(object):
  대리인_목록 = []
    훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def __init__(self, name):
    self.대리인_목록.append(name)
    self.이름 = name

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')

  def 포장하기(self):
    print('포장이 완료되었습니다.')

대리인1 = 대리인('라이캣하나')
대리인2 = 대리인('라이캣둘')

print(대리인1.이름)
print(대리인2.이름)

print(대리인1.대리인_목록)
print(대리인2.대리인_목록)
```

Python

```
#[Out]

라이캣하나
라이캣둘
['라이캣하나', '라이캣둘']
['라이캣하나', '라이캣둘']
```

Python

이 메서드는 인스턴스 객체 생성 시 자동으로 실행합니다. 어떤가요? 코드가 좀 더 간결해졌습니다!

좀 더 알아보자면 `__init__` 메서드는 다른 프로그래밍 언어에서의 생성자(`constructor`) 역할을 하는 클래스 메서드입니다.

2.3 상속

```
#[In]

class 대리인(object):
  이름 = '라이캣의 대리인'
  대리인_목록 = []
  훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def __init__(self, name):
    self.대리인_목록.append(name)
    self.이름 = name

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')

  def 포장하기(self):
    print('포장이 완료되었습니다.')
```

```python
class 진화된_대리인(대리인):
  #이제 위니브 월드에 있는 대리인들 끼리는 서로 협업합니다.
  추가_서로협업하기 = True

  def 추가된_기능_청소기능(self):
    print("청소기능이 추가되었습니다.")

대리인1 = 대리인('라이캣하나')
대리인2 = 진화된_대리인('라이캣둘')

대리인2.추가된_기능_청소기능()
대리인2.이름
대리인2.훈장
```

Python

```
#[Out]

청소기능이 추가되었습니다.
None
라이캣둘
['백상아리를 잡은 고양이',
 '성실한 납세자',
 '해골섬 낚시꾼',
 '청년 고용 착한 기업',
 '회사를 설립한 자',
 '혈통의 인정을 받은 자',
 '천민을 위한 병원을 세운 자']
```

Python

클래스를 상속하는 방법은 위의 예시처럼 새로운 클래스의 `()` 안에 상속할 클래스명을 적어주는 것입니다.

`진화된_대리인` 클래스는 기존의 `대리인` 클래스가 사용하던 데이터와 기능들을 모두 사용할 수 있으며, 추가적인 기능을 만들었습니다.

또한 데이터(클래스 변수와 메서드를 재정의할 수 있습니다)를 새로 덮어씌울 수도 있으며 자기 자신만의 클래스 변수를 새로 정의할 수 있습니다.

여기에서 상속을 하는 `대리인`을 부모 클래스(슈퍼 클래스)라고 부르고 `진화된_대리인`을 자식 클래스(서브 클래스)라고 부릅니다.

2.4 다중 상속

여러 클래스를 상속받을 때는 아래와 같이 사용하면 됩니다. 다중 상속은 단일 상속 법과 같습니다. 여러개를 상속 받을 때에는 콤마를 사용합니다.

```python
#[In]

class 배터리_영구_증가(object):
  pass

class 효율_극대화(object):
  pass

class 대리인(object):
  이름 = '라이캣의 대리인'
  대리인_목록 = []
  훈장 = [
    '백상아리를 잡은 고양이',
    '성실한 납세자',
    '해골섬 낚시꾼',
    '청년 고용 착한 기업',
    '회사를 설립한 자',
    '혈통의 인정을 받은 자',
    '천민을 위한 병원을 세운 자'
  ]
  기술 = {
    '회사운영': 99,
    '낚시' : 96,
    '통발' : 93,
    '큰그물' : 95,
    '재무' : 34,
  }

  def __init__(self, name):
    self.대리인_목록.append(name)
    self.이름 = name

  def 계산하기(self, x):
    print(int(x)*5000, '원 입니다.')

  def 포장하기(self):
    print('포장이 완료되었습니다.')

class 진화된_대리인(대리인, 배터리_영구_증가, 효율_극대화):
  #이제 위니브 월드에 있는 대리인들 끼리는 서로 협업합니다.
  추가_서로협업하기 = True

  def 추가된_기능_청소기능(self):
    print("청소기능이 추가되었습니다.")
```

Python

2.5 특별 메소드(magic method)

앞서 말씀드린 것처럼 파이썬에서의 클래스에는 기본적으로 내장하고 있는 특별 메서드들이 있습니다. 파이썬에서는 이런 내장하고 있는 특별 메서드들을 쉽게 재정의하여 사용할 수 있게 되어있습니다. 매직 매서드라고 부릅니다. 혹은 더블 언더바(__)를 쓰고 있는 점에서 줄여서 던더(dunder) 메서드라고도 부릅니다.

```python
#[In]

class 신규_로봇(object):
  def __init__(self, name):
    self.name = name
  def __getattr__(self, item):
    print(item + '속성은 존재하지 않습니다.')

로봇1 = 신규_로봇('로봇1')
print(로봇1.name)
print(로봇1.청소하기)
```

Python

```
#[Out]

로봇1
청소하기속성은 존재하지 않습니다.
None
```

Python

위의 예시에서 `__init__`과 `__getattr__`이 특별 메서드입니다. 예시에서 보는 바와 같이 파이썬에 내장되어 있는 특별 메서드들을 쉽게 재정의하여 사용할 수 있습니다.

> 💡 **특별 메서드들의 종류를 자세하게 알고 싶다면 파이썬 공식 홈페이지를 참고바랍니다.**
> (https://docs.python.org/3/reference/datamodel.html)

2.6 캣의 준비물

자, 그럼 앞서 `type` 함수를 호출하면 나왔던 `class 'int'` 는 무엇일까요? 맞습니다. python은 모든 자료형이 class로 되어 있어요!
위니브 월드에 비유해보자면, 사실은 실제하는 캣도, class 였던 것이죠! 이제 대리인의 type과 dir을 찍어보세요. 보다 확연히 알 수 있을 겁니다.

모든 준비가 끝났으니 가볍게 준비물을 챙겨 떠나보도록 하죠. append, insert 등의 매서드를 사용해서 필요한 준비물들을 넣어보세요. 창의력을 발휘해보세요.

여러분들이라면 무엇을 챙겨가시겠나요?

```python
# 캣의 준비물
준비물 = [??, ??, ??, ??, ??]
```

Python

자, 이제 제대로 된 여행준비를 하고, 여행을 떠나봅시다.

해당 스토리는 자고일어나니 코딩테스트 시리즈와 이어집니다.

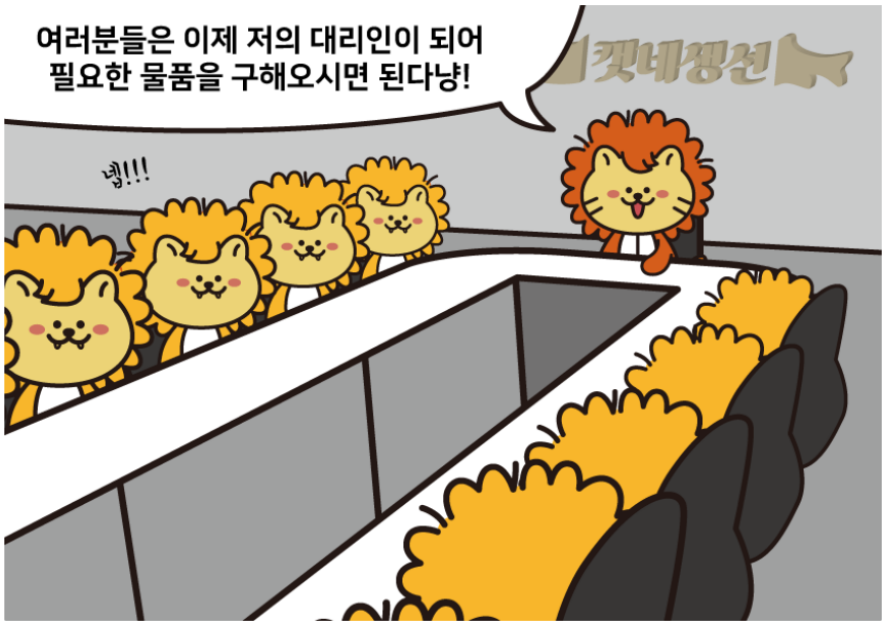
DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

캣의 모험 자금을 모아라!

PYTHON BOOTCAMP



라이캣 대리인 10명을 만들어 여행에 필요한 물품을 알아서 구해오도록 시킬 생각입니다. 물론 자신도 직접 뛰어 물품을 구할 생각이에요.

"물론 혼자서도 할 수 있지만 기회비용을 생각했을 때, 최선의 결과를 도출하기 위해 대리인을 만드는 것이 좋겠다냥!"

라이캣은 각각의 대리인들이 비교우위를 점하지 않기 때문에 일을 적절하게 분배하기로 했습니다.

1. **대리인 10명을 생성하세요!**

   아래 코드를 참고하여 물건을 구할 대리인을 설계하십시오.

```python
class 대리인(object):
    pass

대리인_그룹 = []
for name in range(10):
    대리인_그룹.append(대리인(str(name)))
```

Python

→ 대리인의 이름은 0, 1, 2, 3, 4...9로 10명입니다!

→ 경비를 구하는 기능이 있습니다. 아래와 같이 입력했을 때, 시간당 100만 노드를 버는 메서드를 만들어주세요.

```python
인스턴스이름.경비구하기('10시간')
```

Python

→ 물건을 구하는 기능이 있습니다. 아래와 같이 입력을 했을 때, 시간당 필요 물품 1개를 구해오는 메서드를 만들어주세요. 물품은 클래스 변수로, 모두가 공유합니다.

```
#[In]
인스턴스이름.물품구하기('3시간')
인스턴스이름.물품

#[Out]
['대리인이 구해온 물품 1', '대리인이 구해온 물품 2', '대리인이 구해온 물품 3']
```

Python

2. **대리인에게 스킬 추가하기!**

   아래와 같은 코드가 있었다고 하였을 때 이 대리인의 모든 스킬 합을 구하십시오. 또, 스킬을 추가해주는 메서드를 만들어주세요!

```
class 대리인(object):
    이름 = '라이캣의 대리인'
    대리인_목록 = []
    훈장 = [
          '백상아리를 잡은 고양이',
          '성실한 납세자',
          '해골섬 낚시꾼',
          '청년 고용 착한 기업',
          '회사를 설립한 자',
          '혈통의 인정을 받은 자',
          '천민을 위한 병원을 세운 자'
          ]
    기술 = {
        '회사운영': 99,
        '낚시' : 96,
        '통발' : 93,
        '큰그물' : 95,
        '재무' : 34
        }

    def __init__(self, name):
        self.대리인_목록.append(name)
        self.이름 = name

    def 계산하기(self, x):
        print(int(x)*5000, '원 입니다.')

    def 포장하기(self):
        print('포장이 완료되었습니다.')
```

Python

3. **진화된 대리인!**

   라이캣은 자신과 같이 일하는 좀 더 진화된 대리인을 원했습니다. 아래 기능을 도입하여 좀 더 진화된 대리인을 만들어 봅시다. 각각에 메서드는 자유롭게 구현해보세요.

   - 진화된_대리인1은 던전 탐험 추가, 희귀 아이템을 수집합니다.

   - 진화된_대리인2는 위험 요소를 미리 판단하여 유사 시 자금을 회피할 수 있는 기능을 도입합니다.

```python
진화된_대리인1.던전탐험('진홍의 목걸이')
진화된_대리인2.위험판단('위험')
진화된_대리인2.위험판단('평범')
진화된_대리인2.위험판단('긴장감 고조')
```

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

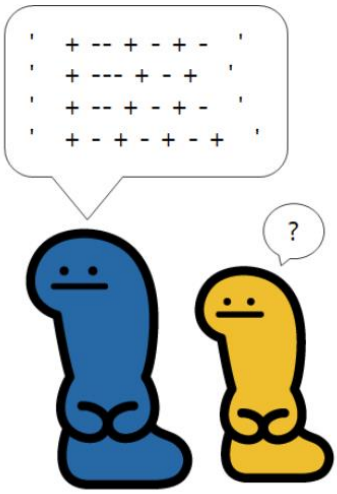
파이와 썬이 남긴 단 하나의 단서

PYTHON BOOTCAMP

모든 알고리즘을 해독할 수 있는 알고리즘 7 원석를 보유한 알고리즘 제왕 **파이**와 **썬**은 죽기 전, 이 보물에 '암호'를 걸어 세계 어딘가에 묻어놨다고 공표하였다. 그가 남긴 문자는 아래와 같다.

```
섬으로 향하라!

'   + -- + - + -   '
'   + --- + - +    '
'   + -- + - + -   '
'   + - + - + - +   '

해(1)와 달(0),
Code의 세상 안으로!(En-Coding)
```

Python



출력조건 : 문자열

해당 문제는 **눈떠보니 코딩 테스트 전날**로 이어집니다. Bootcamp 얕은물, 고생 많으셨어요! 😉

DATE.

비고						
	1					
	2					
	3					
	4					
	5					
	6					
	7					
	8					
	9					
	10					
	11					
	12					
	13					
	14					
	15					
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					

MEMO

7. CheatSheet

Tag Name	설명	구분	대표 속성
int	정수형	자료형 Built-in Functions	int
float	실수형	자료형 Built-in Functions	float
str OPEN	문자열	자료형 Built-in Functions	문자열
bool	부울형 불리언 자료형 참 거짓	자료형 Built-in Functions	bool
list	[값...] 순서가 있는 자료형	자료형 Built-in Functions	연산자 우선순위 1 리스트
tuple	(값...) 순서가 있는 자료형 값을 바꿀 수 없다	자료형 Built-in Functions	연산자 우선순위 1 튜플
dictionary	{키:값...} 순서가 없는 자료형	자료형 Built-in Functions	연산자 우선순위 1 딕셔너리
set	{값...} 순서가 없는 자료형 중복 허용 안함 인덱싱 지원 안함	자료형 Built-in Functions	연산자 우선순위 1 set
type(변수명)	입력값의 자료형이 무엇인지 알려 주는 함수	Built-in Functions	
print()	출력	Built-in Functions	
print(f'{변수}')	f프린트 용법, {}(중괄호)이용한 출력용법	출력 용법	
print({}.format(변수))	문자열 포매팅 방식	출력 용법 메서드	문자열 리스트
print(변수1,변수2)	,(콤마)로 연결하여 출력	출력 용법	
print('%s %d' %(변수1,변수2))	%(퍼센트) 문자열 포매팅	출력 용법	문자열
input()	입력 받을 때 사용하는 함수	Built-in Functions	
len()	길이 구하기	Built-in Functions	
sum()	입력받은 리스트나 튜플의 모든 요소의 합	Built-in Functions	
sorted()	입력값을 정렬한 후 그 결과를 리스트로 돌려준다	Built-in Functions	
round()	반올림	Built-in Functions	
pow()	제곱	Built-in Functions	
min()	최솟값	Built-in Functions	
max()	최댓값	Built-in Functions	
map(함수, 반복 가능한 자료형)	입력받은 자료형의 각 요소를 함수 결과를 묶어서 돌려주는 함수	Built-in Functions	
id()	객체의 고유 주소 값	Built-in Functions	
dir()	객체가 가지고 있는 변수나 함수	Built-in Functions	
chr()	아스키 코드 값을 입력받아 해당하는 문자 출력	Built-in Functions	
\n	문자열 안에서 줄을 바꿀 때 사용	이스케이프 코드	문자열
\t	문자열 사이에 탭 간격을 줄 때 사용	이스케이프 코드	문자열
\\	문자(\)를 그대로 표현할 때 사용	이스케이프 코드	문자열
\'	작은따옴표(') 그대로 표현할 때 사용	이스케이프 코드	문자열
\"	큰따옴표(") 그대로 표현할 때 사용	이스케이프 코드	문자열
\r	줄 바꿈문자 현재 커서를 가장 앞으로 이동	이스케이프 코드	문자열
\f	줄 바꿈 문자 현재 커서를 다음 줄로 이동	이스케이프 코드	문자열
\a	벨 소리	이스케이프 코드	문자열
\b	백 스페이스	이스케이프 코드	문자열
\000	널문자	이스케이프 코드	문자열
from 모듈이름 import 모듈함수	모듈 불러오기	keyword	
import 모듈이름	모듈 불러오기	keyword	
open	파일 객체 돌려주는 함수	파일 관련 Built-in Functions	
open(filename, 'w')	쓰기 모드로 파일 열기	파일 관련	
open(filename, 'r')	읽기 모드로 파일 열기	파일 관련	
open(filename, 'a')	추가 모드로 파일 열기	파일 관련	
open(filename, 'b')	바이너리 모드로 파일 열기	파일 관련	
write	파일에 내용 쓰기	파일 관련	
read	파일의 내용 전체를 문자열	파일 관련	
readline	파일의 내용을 한줄씩 읽음	파일 관련	
readlines	파일의 모든 줄을 읽어서 줄을 요소로 갖는 리스트	파일 관련	
변수명[번호]	인덱싱		문자열 리스트 튜플
변수명[시작 번호 : 끝 번호]	슬라이싱		문자열 리스트 튜플
변수명[시작 번호 :]	시작 번호부터 그 문자열의 끝		문자열 리스트 튜플
변수명[: 끝 번호]	문자열의 처음부터 끝 번호		문자열 리스트 튜플
변수명[:]	문자열의 처음부터 끝		문자열 리스트 튜플
변수명[-1]	마지막 요소		문자열 리스트 튜플

count	개수 세기	메서드	문자열 리스트
join	문자열 삽입	메서드	문자열
upper	소문자를 대문자로 바꾸기	메서드	문자열
lower	대문자를 소문자로 바꾸기	메서드	문자열
lstrip	왼쪽 공백 지우기	메서드	문자열
rstrip	오른쪽 공백 지우기	메서드	문자열
strip	양쪽 공백 지우기	메서드	문자열
replace	문자열 바꾸기	메서드	문자열
split	문자열 나누기	메서드	문자열
append	리스트에 요소 추가	메서드	리스트
sort	리스트 정렬	메서드	리스트
reverse	리스트 뒤집기	메서드	리스트
insert	요소 삽입	메서드	리스트
remove	요소 제거	메서드	리스트
pop	요소 끄집어 내고 삭제	메서드	리스트
extend	확장	메서드	리스트
변수명.keys()	변수의 키만 모아서 출력	메서드	딕셔너리
변수명.values()	values 리스트 만들기	메서드	딕셔너리
변수명.items()	key, value 쌍 얻기	메서드	딕셔너리
변수명.get('키')	key로 value 얻기	메서드	딕셔너리
변수명['키']='값'	딕셔너리 추가하기	메서드	딕셔너리
del 변수명[]	요소 삭제	keyword	리스트 튜플 딕셔너리
add	값 추가하기	메서드	집합
update	값 여러개 추가하기	메서드	집합
remove	특정 값 제거하기	메서드	집합
변수1.intersection(변수2)	교집합	메서드	집합
변수1..union(변수2)	합집합	메서드	집합
변수1.difference(변수2) OPEN	차집합	메서드	집합
%c	문자	포맷코드의 종류	문자열
%s	문자열	포맷코드의 종류	문자열
%d	정수	포맷코드의 종류	문자열
%f	실수	포맷코드의 종류	문자열
%b	2진수	포맷코드의 종류	문자열
%o	8진수	포맷코드의 종류	문자열
%x	16진수	포맷코드의 종류	문자열
%%	리터럴 %(문자 %자체)	포맷코드의 종류	문자열
{{ }}	중괄호 문자를 사용하고 싶은 경우	포맷코드의 종류	문자열
"{0:<자릿수}".format("")	왼쪽 정렬 자릿수를 맞출 수 있다	포맷코드의 종류 메서드	문자열
"{0:>자릿수}".format("")	오른쪽 정렬 자릿수를 맞출 수 있다	포맷코드의 종류 메서드	문자열
"{0:^자릿수}".format("")	가운데 정렬 자릿수를 맞출 수 있다	포맷코드의 종류 메서드	문자열
"{0:문자값^자릿수}".format("")	공백 대신 문자값으로 공백 채우기	포맷코드의 종류 메서드	문자열
+	덧셈	산술연산자	연산자 우선순위 7
-	뺄셈	산술연산자	연산자 우선순위 7
*	곱셈	산술연산자	연산자 우선순위 6
**	제곱	산술연산자	연산자 우선순위 4
/	나눗셈	산술연산자	연산자 우선순위 6
//	몫	산술연산자	연산자 우선순위 6
%	나머지	산술연산자	연산자 우선순위 6
@	행렬 곱셈	산술연산자	연산자 우선순위 6
=	대입	대입(할당)연산자	연산자 우선순위 16
+=	덧셈 대입	대입(할당)연산자	연산자 우선순위 16
-=	뺄셈 대입	대입(할당)연산자	연산자 우선순위 16
*=	곱셈 대입	대입(할당)연산자	연산자 우선순위 16
**=	제곱 대입	대입(할당)연산자	연산자 우선순위 16
/=	나눗셈 대입	대입(할당)연산자	연산자 우선순위 16
//=	몫 대입	대입(할당)연산자	연산자 우선순위 16

%=	나머지 연산 대입	대입(할당)연산자	연산자 우선순위 16
>	~보다 큰	비교연산자	연산자 우선순위 12
<	~보다 작은	비교연산자	연산자 우선순위 12
≥	~보다 크거나 같은	비교연산자	연산자 우선순위 12
≤	~보다 작거나 같은	비교연산자	연산자 우선순위 12
==	같은	비교연산자	연산자 우선순위 12
! =	다른	비교연산자	연산자 우선순위 12
and	그리고	논리연산자 keyword	연산자 우선순위 14
or	또는	논리연산자 keyword	연산자 우선순위 15
not	부정	논리연산자 keyword	연산자 우선순위 13
&	AND 연산	비트연산자	연산자 우선순위 9
|	OR 연산	비트연산자	연산자 우선순위 11
^	XOR 연산	비트연산자	연산자 우선순위 10
~	보수연산	비트연산자	연산자 우선순위 5
<<	왼쪽 쉬프트 연산	비트연산자	연산자 우선순위 8
>>	오른쪽 쉬프트 연산	비트연산자	연산자 우선순위 8
in	왼쪽 항목이 오른쪽 배열에 포함되면 True 그렇지 않으면 False	멤버 연산자 keyword	연산자 우선순위 12
not in	왼쪽 항목이 오른쪽 배열에 포함 안 되면 True 그렇지 않으면 False	멤버 연산자 keyword	연산자 우선순위 12
is	피연산자들의 위치(주소)가 같다면 True 그렇지 않으면 False	식별 연산자 keyword	연산자 우선순위 12
is not	피연산자들의 위치(주소)가 다르면 True 그렇지 않으면 False	식별 연산자 keyword	연산자 우선순위 12
await x	await 표현식		연산자 우선순위 3
lamda	람다 표현식		연산자 우선순위 18
if	조건이 참이면 조건에 맞는 것 실행	조건문 keyword	
elif	if문의 조건이 거짓이고 elif문의 조건이 참일 경우 실행	조건문 keyword	
else	if문의 조건이 거짓일 경우만 실행	조건문 keyword	
for (변수명) in (순회 가능한 객체)	순서열을 순회하며 끝에 도달하면 반복을 멈춤	반복문 keyword	
range(start, stop, step)	연속하는 수열 만듬	반복문 Built-in Functions	
while 조건문: 수행할 문장	조건이 참인 동안에 명령을 반복해서 수행	반복문 keyword	
break	반복문 중단	keyword	
continue	하위 코드를 실행하지 않고 다음 반복으로 넘어감	keyword	
pass	아무런 동작을 하지 않고 다음 코드를 실행	keyword	
enumerate(변수명)	순서 매길 때 사용	반복문 Built-in Functions	
try, except, else	예외처리	반복문 keyword	
def	함수 선언	함수 keyword	
return 변수명	함수값 반환	함수 keyword	
global	프로그램 어디에서나 접근이 가능한 변수	전역변수 keyword	
__init__	반복을 끝낼 값을 지정하면 특정 값이 나올 때 반복을 끝냄	매직메서드(던더함수)	
__next__	기본값을 지정하면 반복이 끝나더라도 기본값 출력	매직메서드(던더함수)	
class 클래스명(object):	데이터와 기능을 가지고 있는 인스턴트 객체를 생성하기 위한 역할	클래스	
class 클래스명1(클래스명2)	기존의 클래스가 사용하던 데이터와 기능들을 모두 사용	상속	
+ New			
find	문자가 처음 나온 위치 반환	메서드	문자열
index	위치 반환	메서드	문자열 리스트

Day5

Django 3.1

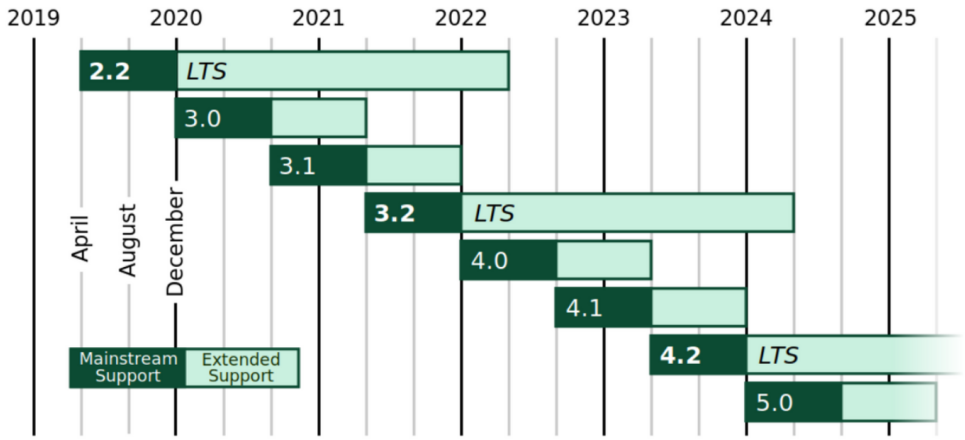
1. 장고 소개
2. 구름 IDE 환경설정
3. 장고 프로젝트 생성
4. 프로젝트 개요
5. 메인 페이지
6. cafelist 페이지
7. 모델 작성
8. cafedetails 페이지
9. 이미지 넣기
10. 댓글기능 추가하기
11. 게시글 작성일/수정일 추가
12. 카페 위치 추가
13. 모델에 정보 추가하기
14. 템플릿 적용 및 검색 기능 구현
15. 카페 등록 페이지
16. 서비스 배포
17. 구글 맵 API
18. 번회 튜토리얼
19. 요약정리

장고 소개

Django는 파이썬 Web Framework 중 가장 사랑받는 Full-Stack Framework로 인스타그램, NASA, 우리가 사용할 댓글 관리 서비스인 Disqus 등에서 사용하고 있습니다.

Full-Stack Framework는 웹 서비스 개발에 필요한 모든 요소들이 한곳에 모여있는 종합선물세트입니다. 보다 빠르고 편리하게 웹 서비스를 개발할 수 있도록 필요한 공구들을 모은 것이죠.

Django 업데이트 로드맵입니다. 현재 Django는 3.x를 쓰는 것이 일반적이며, 저희 책에서는 3.1 Version을 사용하여 기술하도록 하겠습니다.



Release Series	Latest Release	End of mainstream support[1]	End of extended support[2]
3.0	3.0.8	August, 2020	April, 2021
2.2 LTS	2.2.14	December 2, 2019	April 2022
2.1	2.1.15	April 1, 2019	December 2, 2019
2.0	2.0.13	August 1, 2018	April 1, 2019
1.11 LTS [3]	1.11.29	December 2, 2017	April 1, 2020
1.10	1.10.8	April 4, 2017	December 2, 2017
1.9	1.9.13	August 1, 2016	April 4, 2017
1.8 LTS	1.8.19	December 1, 2015	April 1, 2018
1.7	1.7.11	April 1, 2015	December 1, 2015
1.6	1.6.11	September 2, 2014	April 1, 2015
1.5	1.5.12	November 6, 2013	September 2, 2014
1.4 LTS	1.4.22	February 26, 2013	October 1, 2015
1.3	1.3.7	March 23, 2012	February 26, 2013

Release Series	Release Date	End of mainstream support[1]	End of extended support[2]
3.1	August 2020	April 2021	December 2021
3.2 LTS	April 2021	December 2021	April 2024
4.0	December 2021	August 2022	April 2023
4.1	August 2022	April 2023	December 2023
4.2 LTS	April 2023	December 2023	April 2026

https://www.djangoproject.com/download/

Django는 2005년 오픈소스로 시작되어 현재 2020년 8월 기준 Django 3.1이 나왔습니다. 앞서 말씀드린 것처럼 이 책은 3.1 기준으로 기술되었습니다.

책에는 상세한 내용을 다루기보다 전체적인 맥락을 잡기 위해 한 Circle을 도는 것에 초점이 맞춰져 있으니 보다 자세한 내용은 공식 홈페이지에 문서를 참고하시면 좋을 것 같아요.

Django

The Django Software Foundation deeply values the diversity of our developers, users, and community. We are distraught by the suffering, oppression, and systemic racism the Black community faces every day. We can no longer remain silent. In silence, we

https://www.djangoproject.com/

공식 홈페이지에는 django에 대한 소개와 다운로드, 지원 문서를 제공합니다. 이 지원문서에 django에 대한 내용이 다 담겨 있으니 개발할 때 꼭 참고하시면서 개발하세요.

앞에서도 한 번 말씀드린 공식 홈페이지와 공식 PDF문서입니다.

공식문서 :

Django documentation | Django documentation | Django

Everything you need to know about Django. Are you new to Django or to programming? This is the place to start! Having trouble? We'd like to help! Django has a lot of documentation. A high-level overview of how it's organized will help you know

https://docs.djangoproject.com/en/3.0/

PDF 파일 :

buildmedia.readthedocs.org

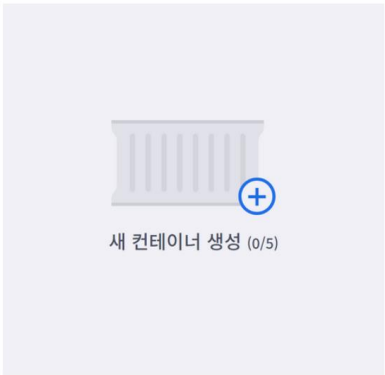
https://buildmedia.readthedocs.org/media/pdf/django/3.0.x/django.pdf

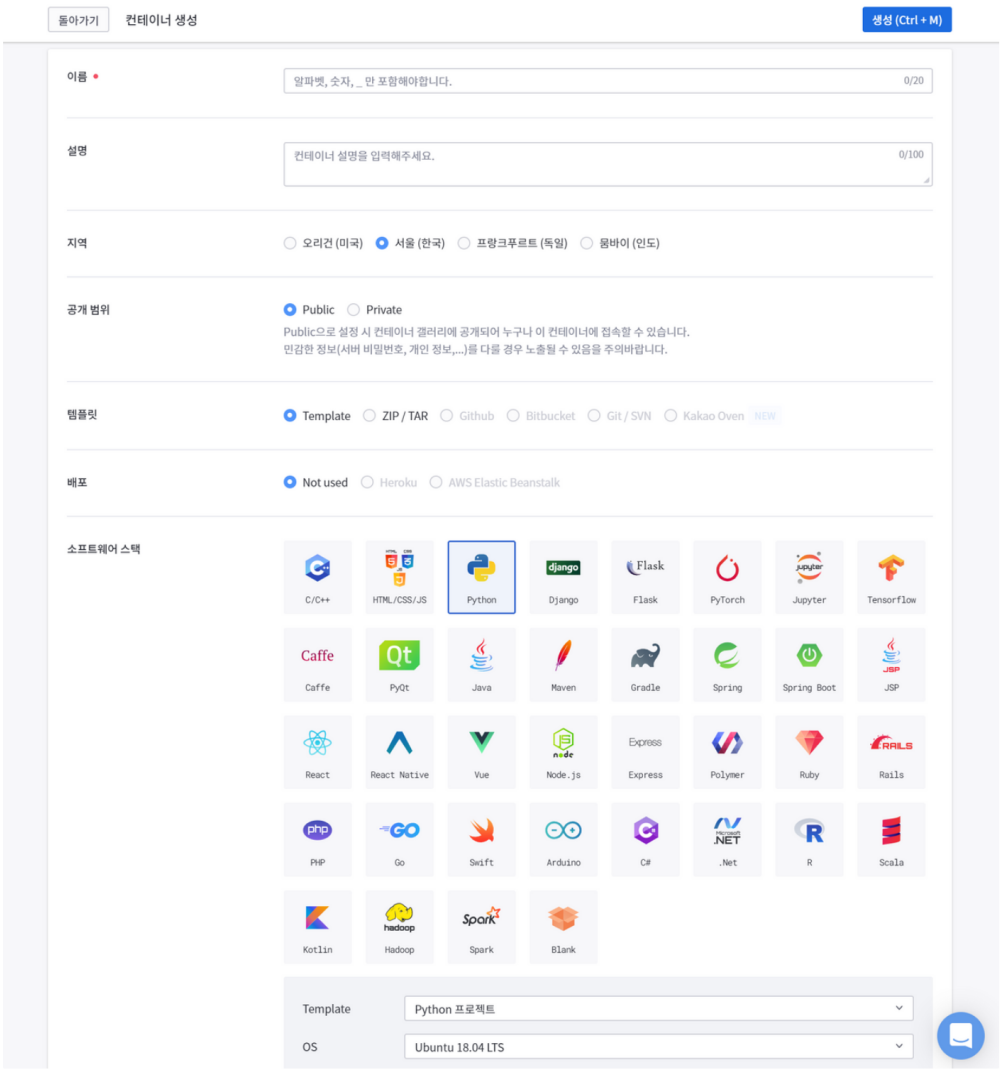
구름 IDE 환경설정

우리가 구름을 사용하는 이유는 3가지가 있습니다.

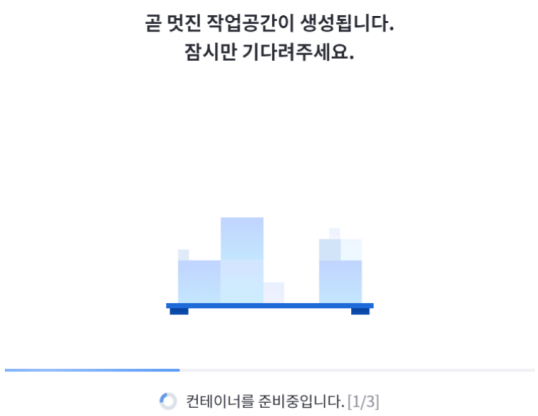
1. OS Version, Python Version, Django Version 별 에러를 잡는데 시간이 많이 소요됩니다. 또한 '개발은 배포 환경과 동일하게!'라는 말을 잊지 마세요. 우리의 리소스를 아껴줄 것입니다.
2. 집에 서버와 환경을 구축하는 것은 비용(시간 비용, 금전적 비용, 기회 비용)이 상대적으로 큽니다.
3. 배포를 명령어 한 번으로 진행할 수 있어요! 또는 구름에서도 서비스를 런칭할 수도 있답니다.

앞서 컨테이너를 만들었지만 여기서 새 컨테이너를 생성하여 진행하도록 하겠습니다. 새 컨테이너 생성을 누르셔서 컨테이너를 생성해 주세요.

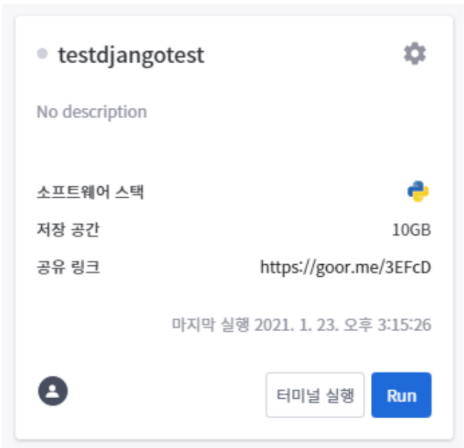




컨테이너가 만들어지고 있다는 화면이 나오고 곧 컨테이너 생성이 완료되었다고 뜹니다. 컨테이너 실행을 누르셔서 컨테이너 안으로 들어가주세요. 혹시 중간에 닫기 버튼을 누르셨다면 대시보드에서 해당 컨테이너 실행을 누르시면 됩니다.


곧 멋진 작업공간이 생성됩니다.
잠시만 기다려주세요.
컨테이너를 준비중입니다. [1/3]


goormide
컨테이너가 성공적으로 생성되었습니다.
컨테이너 실행
대시보드 이동


testdjangotest
No description
소프트웨어 스택
저장 공간
10GB
공유 링크
https://goor.me/3EFcD
마지막 실행 2021. 1. 23. 오후 3:15:26
터미널 실행
Run

장고 프로젝트 생성

A. 장고 프로젝트 생성

B. 장고 프로젝트 기본 세팅

A. 장고 프로젝트 생성

다음 명령어들을 차례로 입력하여 장고 프로젝트를 생성합니다.

여기서 주의하셔야 할 것이 `django-admin startproject tutorialdjango .` 명령어 마지막에 점(`.`)을 꼭 기입하셔야 합니다!

```
root@goorm:/workspace/tutorialdjango# pip install --upgrade pip

root@goorm:/workspace/tutorialdjango# mkdir mysite

root@goorm:/workspace/tutorialdjango# cd mysite

root@goorm:/workspace/tutorialdjango/mysite# python -m venv myvenv

root@goorm:/workspace/tutorialdjango/mysite# source myvenv/bin/activate

(myvenv)root@goorm:/workspace/tutorialdjango/mysite# pip install django==3.1

(myvenv)root@goorm:/workspace/tutorialdjango/mysite# django-admin startproject tutoriald
jango .

(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate
```

```
(myvenv) root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying sessions.0001_initial... OK
```

위와 같이 마지막 명령어에서 OK가 모두 뜨셨다면 성공한 것입니다! 자, 이제 지금까지 타이핑 하셨던 명령어를 설명해드리도록 하겠습니다.

최신의 pip으로 update하는 명령어 입니다.

```
root@goorm:/workspace/컨테이너명# pip install --upgrade pip
```

mkdir은 디렉토리를 만들어주는 명령어 입니다. mysite라는 폴더를 만들 것입니다.

```
root@goorm:/workspace/컨테이너명# mkdir mysite
```

cd명령어는 change directory 명령어 입니다. 우리가 앞서 만든 mysite라는 폴더로 이동합니다.

```
root@goorm:/workspace/컨테이너명# cd mysite
```

가상환경을 생성하는 명령어 입니다. 가상환경 설정은 프로젝트 관리를 편하게 해주고 버전별 충돌을 막아줍니다. venv 명령어가 실행이 되지 않는 환경일경우 **pip install virtualenv**로 virtualenv를 설치해주세요.

```
root@goorm:/workspace/컨테이너명/mysite# python -v venv myvenv
```

이 명령어를 실행하고 앞서 실행한 가상환경으로 들어가겠다는 명령어 입니다. 이 명령어는 구름IDE 컨테이너를 다시 실행할때마다 쳐주셔야 합니다.

```
root@goorm:/workspace/컨테이너명/mysite# source myvenv/bin/activate
```

패키지까지 설치하기 위한 명령어 입니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install django==3.1
```

현재 있는 폴더에 프로젝트를 '컨테이너이름'으로 만들어 시작하겠다는 명령어에요. 다시 한 번 강조합니다. `프로젝트이름` 뒤에 `.`이 있는데 생략하면 안됩니다. 꼭 점을 넣어주세요! 현재 폴더에 프로젝트를 생성하겠다는 명령어 입니다.

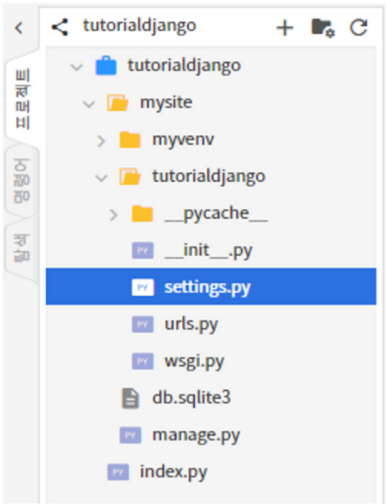
```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# django-admin startproject 프로젝트이름 .
```

migrate 명령어는 하편에서 설명해 드리도록 하겠습니다. 쉽게 말해 DB에 값을 넣는 작업입니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

B. 장고 프로젝트 기본 세팅

이제 `settings.py`에 28번째 줄에 가서서 `ALLOWED_HOSTS` 변수의 값을 `['*']`로 바꿔 주십시오. 모든 사용자의 접속을 허락하겠다는 것입니다. 그리고 106번째 줄의 `LANGUAGE_CODE`와 `TIME_ZONE`을 다음과 같이 수정해주세요. 언어와 지역을 한국 기준으로 바꿔줍니다.




```
ALLOWED_HOSTS=['*']
```

settins.py 28번째 줄

```
LANGUAGE_CODE = 'ko-kr'

TIME_ZONE = 'Asia/Seoul'
```

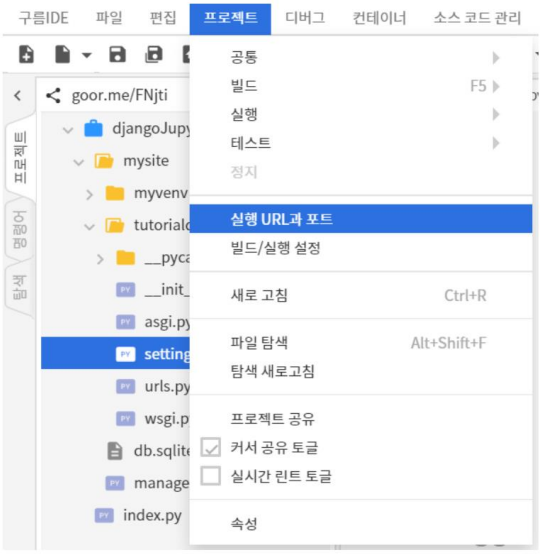
settings.py 106번째 줄

이제 다음 명령어로 서버를 실행시켜주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80
```

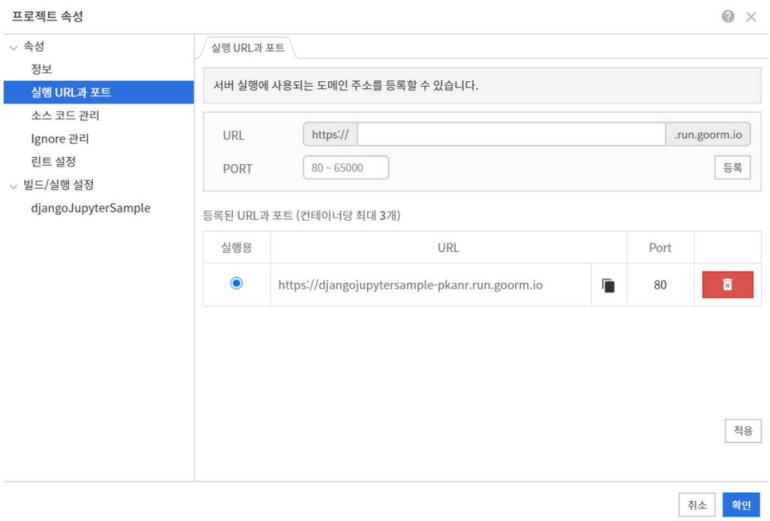
아래와 같은 텍스트가 나왔다면 정상적으로 실행되는 것입니다. 서버를 종료할 때에는 `Ctrl + C` 버튼을 눌러주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80

Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
July 27, 2020 - 05:47:50
Django version 3.0.8, using settings 'tutorialdjango.settings'
Starting development server at http://0:80/
Quit the server with CONTROL-C.
```

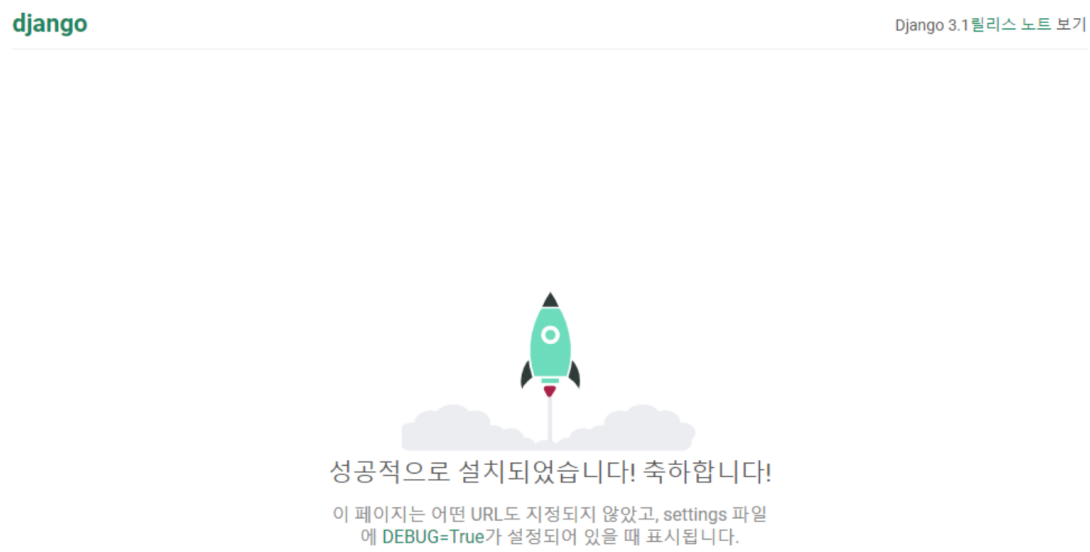


> 실시간 린트 토글을 체킹 해제해 주시면 빨간줄이 생기지 않습니다. 빨간줄이 있다고 실행이 안되는 것은 아니에요. 여러분의 코드를 잘못 체킹하고 있는 것이니, 실시간 린트 토글을 빼줍시다!

`등록된 URL과 포트(컨테이너당 최대 3개)` 아래 있는 `URL`을 클릭하시면 아래와 같이 서버와 연결된 URL이 열리면서 실행화면이 보입니다.



다음과 같이 작동한다면 성공한 것입니다!



04. 프로젝트 개요

완성된 페이지를 보면서 우리가 만들 페이지의 구조를 살펴보도록 하겠습니다. 여기서 유심히 보셔야 할 것은 **URL 구조**입니다.

아래 URL에서 실행되고 있는 서버를 확인하실 수 있습니다.

제주카페찾기

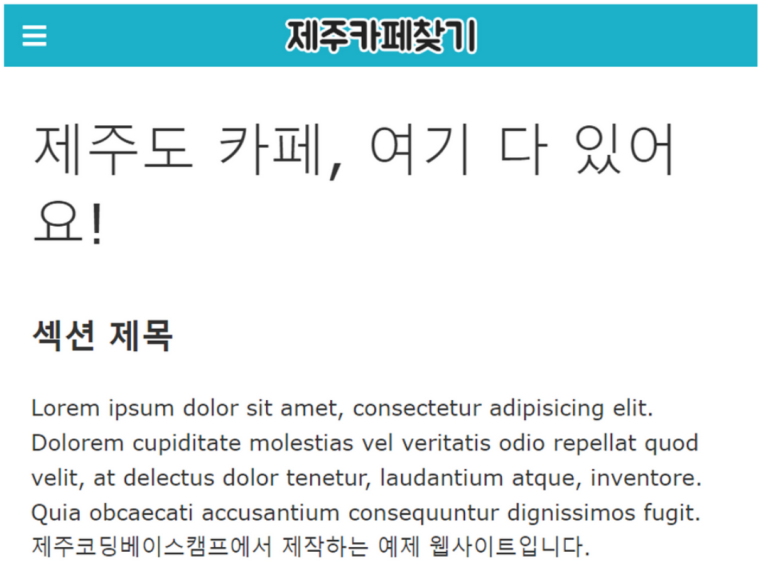
http://jejucafe.jejucodingcamp.com/

1. **https://사용자_지정_도메인.run.goorm.io/**
   - 메인페이지
   - Template : index.html

사용자가 처음 볼 main page 입니다. 여기서 사용자가 원하는 지역을 선택하고 카페 찾기를 누르면 해당 지역의 카페가 찾아지는 구조로 되어 있습니다.

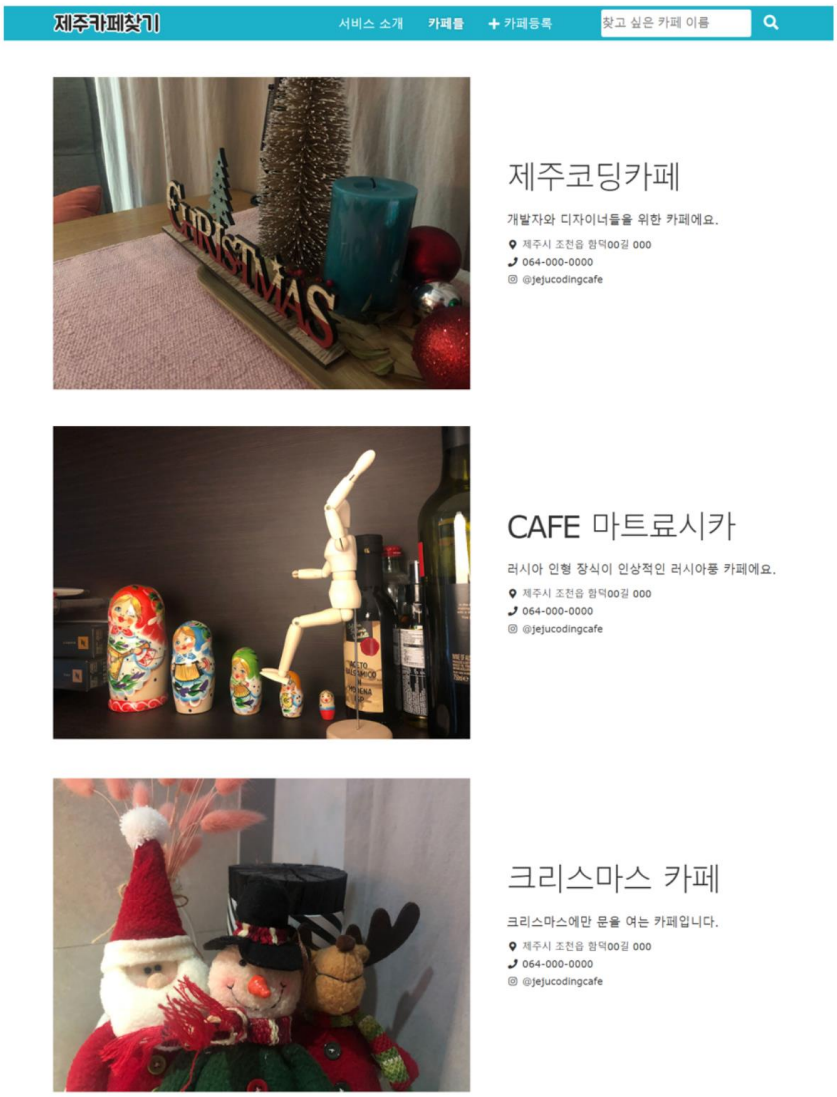
2. https://사용자_지정_도메인.run.goorm.io/about
  - 홈페이지 소개 페이지
  - Template : about.html

about page 입니다. 안에 글귀는 의미 없는 로렘 입숨으로 작성되어 있습니다.

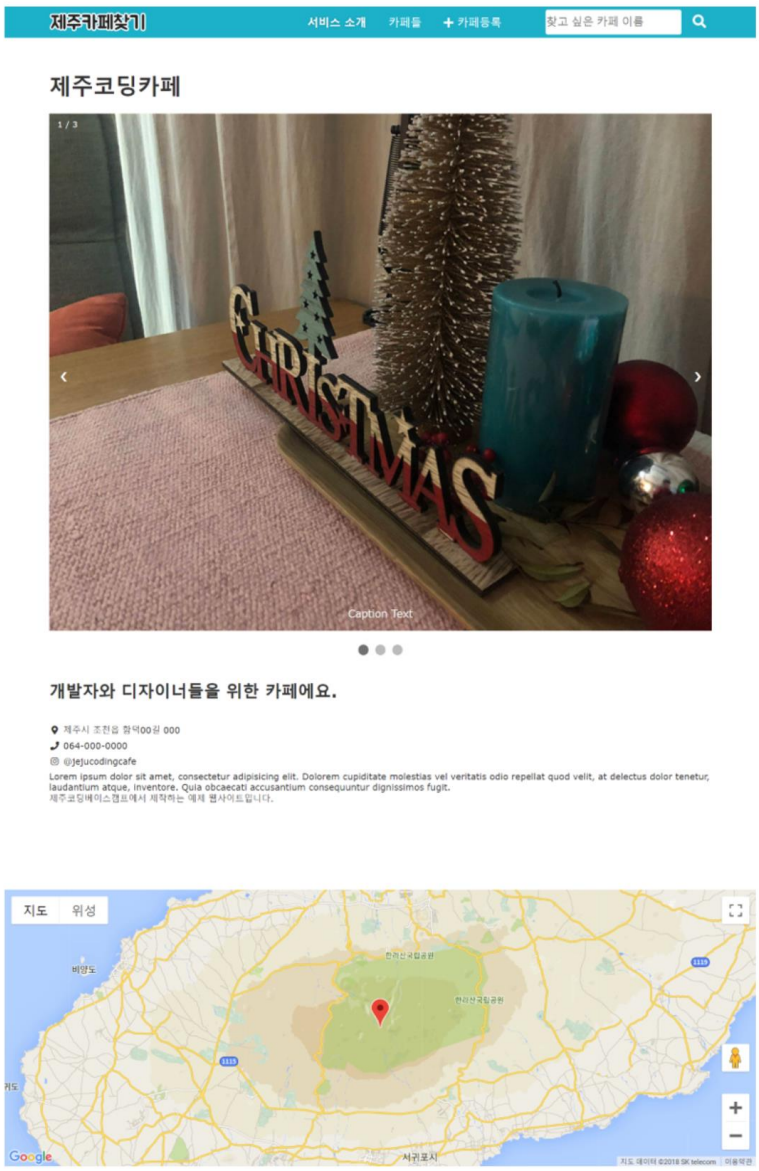


3. https://사용자_지정_도메인.run.goorm.io/cafelist
 - 메인페이지에서 카페들을 클릭했을 경우 이동
 - Template : cafelist.html

cafelist page 입니다. 해당 사진이나 글귀를 눌렀을 경우 상세 페이지로 이동합니다.



4. https://사용자_지정_도메인.run.goorm.io/cafelist/cafe번호
   - cafelist 페이지에서 카페 상세 정보 보기를 눌렀을 때 이동, google map API 사용
   - Template : cafedetails.html



cafedetails page 입니다. 카페의 사진 3장, 설명, 주소와 위치를 나타내고 google MAP API를 활용하여 지도에 위치를 나타내줍니다.

5. https://사용자_지정_도메인.run.goorm.io/cafeenroll
 - 메인페이지에서 카페등록을 클릭했을 경우 이동
 - Template : enroll.html

cafe 등록 페이지입니다. 해당 페이지는 admin 페이지에서 실행할 수 있기 때문에 없어도 되는 페이지이지만 form 연습 차원에서 튜토리얼 안에 넣어봤습니다.

[참고]

프로젝트 전체 소스 코드 : https://github.com/paullabkorea/jejucodingcamp

메인 페이지

메인 페이지 만들기
장고의 MTV 패턴

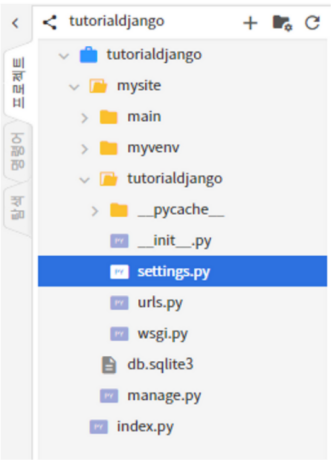
메인 페이지 만들기

Django는 여러 개의 앱을 동시에 만들어 조립하는 과정으로 보실 수 있어요. 우리는 간단한 실습이기 때문에 main이라는 앱만을 만들 것입니다. 우선 서버가 동작되고 있으므로 `Ctrl + C` 버튼을 누르셔서 서버를 종료시킨 다음 아래 명령어를 실행시켜 주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py startapp main
```

아무것도 뜨지 않은 채로 커서가 다음줄로 넘어가면 정상 동작 한 것입니다. 에러가 나오면 에러를 검색해서 에러명을 해결해주세요. 대부분 오타입니다.

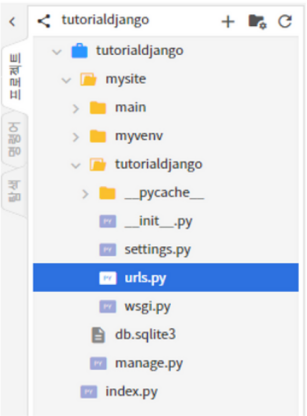
Settings에 가셔서 설치되어 있는 앱 목록에 main을 추가해주세요. 이 작업을 하지 않으면 앱이 구동하지 않습니다. 여러 개의 앱을 만들 경우 모두 여기 등록해 주셔야 합니다.




```
INSTALLED_APPS=[
    'main',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

💡 'main' 뒤에 쉼표가 있으니 꼭 체킹해주세요.

이제 urls.py를 작성할 것입니다. urls는 사용자가 어떤 url을 사용하여 들어오느냐에 따라 어떤 화면을 보여줄지를 결정하게 됩니다.



mysite > tutorialdjango > urls.py

```
from django.contrib import admin
from django.urls import path
from main.views import index

urlpatterns =[
    path('admin/', admin.site.urls),
    path('', index),
]
```

이 코드 중 `path('', index)`의 의미는 사용자가 뒤에 무언가 붙지 않은 있는 그대로의 url을 입력하고 들어왔을 경우 main/views.py파일 안에 index라는 함수를 연결시켜 주겠다는 뜻입니다. 여기서 불러온 index는 함수이며, 아직 만들어지지 않았기 때문에 오류가 뜹니다. 이 index함수는 `index.html`과 연결시켜 줄 것입니다.

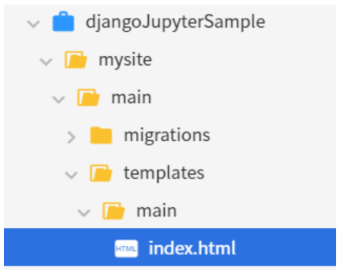
💡 다음과 같이 정의하면 나중에 url로 불러오기 편하지만, 코드 복잡도가 올라가기에, 초보자분들이 하기 쉽도록 나중에 처리해주도록 하겠습니다.
`path('', index, name='index')`

이제 `mysite > main> views.py`로 들어와 index라는 함수를 만듭니다. 여기서 사용자가 index.html을 볼 수 있게끔 연결을 해줍니다.

mysite > main > views.py

```
from django.shortcuts import render

def index(request):
    return render(request, 'main/index.html')
```

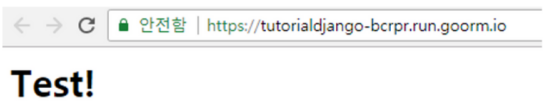
Template을 연결 시키는 과정으로 `mysite > main > templates > main > index.html` 을 만들어 주세요. 기본적인 테스트를 위하여 아래 내용을 붙여 넣어 주세요. 해당 앱이름 안에 templates라는 폴더 아래 다시 앱이름을 넣고 그 안에 `.html` 파일을 넣는 방식은 Django의 규칙입니다. 이 규칙은 `settings.py` 파일 안에 `TEMPLATES` 라는 속성에 `DIR` 값을 수정함으로 바꿀 수 있습니다.




```
<html>
<head>
  <title>Django!</title>
</head>
<body>
  <h1>Test!</h1>
</body>
<html>
```

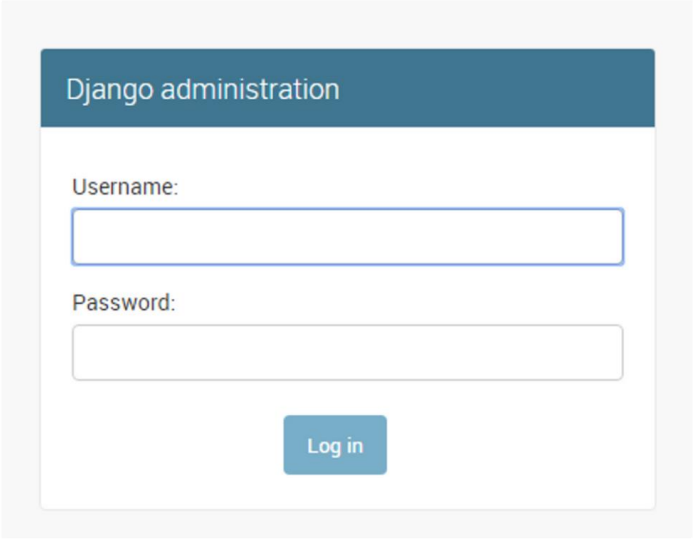
이제 다시 아래 명령어를 콘솔창에 입력하시고 프로젝트 > 실행URL과 포트를 클릭하신 다음 URL을 클릭하셔서 실행되는 화면을 보십시오.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80
```



이번에는 뒤에 /admin을 입력하셔서 admin 페이지가 열리는지 확인해주세요. 여기서는 게시물을 올리거나 수정하거나 삭제할 수 있으며 user에 대한 관리도 할 수 있습니다. 아직 superuser를 만들지 않았으므로 접속 할 수 없습니다.

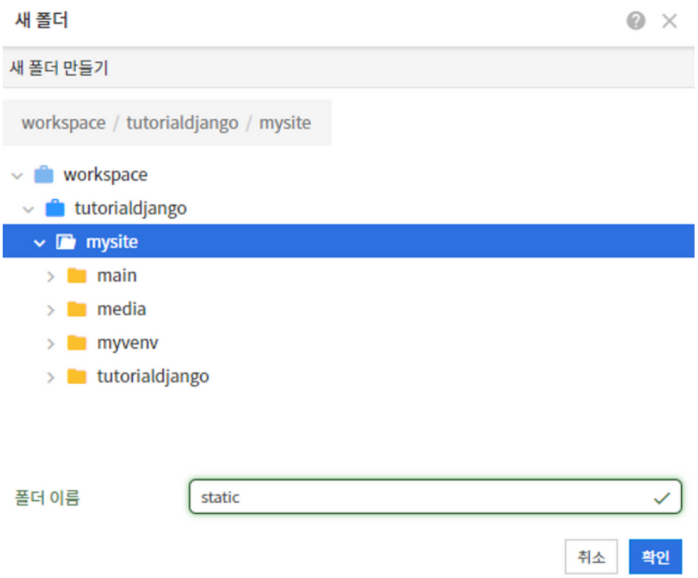
urls.py에서 보았던 admin/ 부분이 이 페이지를 뜻하는 것입니다. 보안상 실 서비스에서는 열어두지 않는 것이 좋습니다.



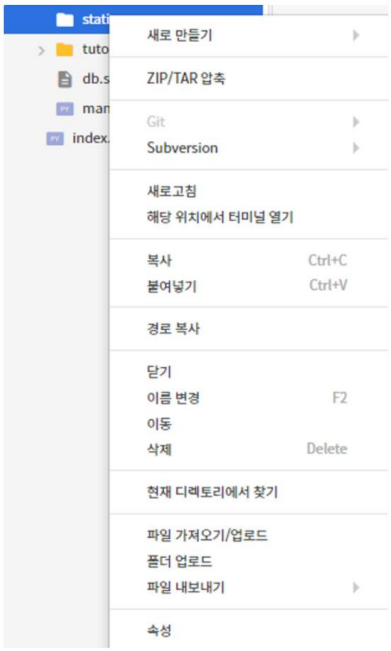
https://여러분URL/admin

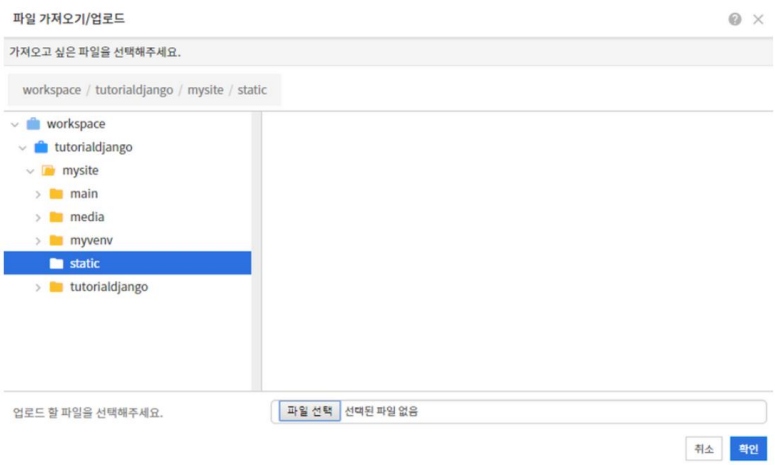
다음으로는 메인페이지에서 이미지를 불러오는 방법에 대해 소개하려 합니다. Django에서는 일반적인 HTML, CSS에서 지원하는 상대경로를 지원하지 않습니다. 우선 정적 파일들이 저장될 static이라는 폴더를 mysite 밑에 만들어 줍니다.

1. 새폴더 만들기



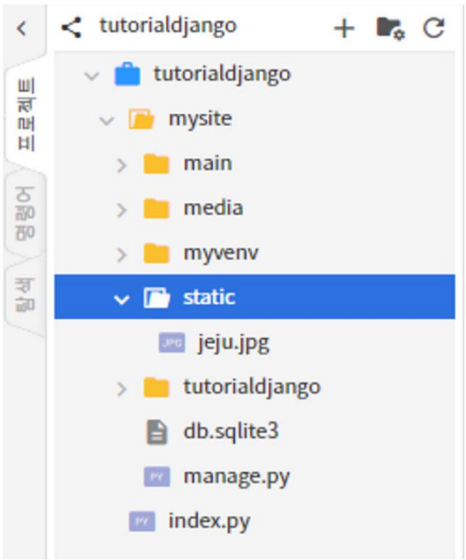
2. 파일업로드


새로 만들기
ZIP/TAR 압축
Git
Subversion
새로고침
해당 위치에서 터미널 열기
복사 Ctrl+C
붙여넣기 Ctrl+V
경로 복사
닫기
이름 변경 F2
이동
삭제 Delete
현재 디렉토리에서 찾기
파일 가져오기/업로드
폴더 업로드
파일 내보내기
속성


파일 가져오기/업로드
가져오고 싶은 파일을 선택해주세요.
workspace / tutorialdjango / mysite / static
workspace
tutorialdjango
mysite
main
media
myvenv
static
tutorialdjango
업로드 할 파일을 선택해주세요.
파일 선택
선택된 파일 없음
취소
확인

3. 파일 업로드 확인

   혹시 이미지가 안떴을 경우에는 새로고침을 눌러주세요.



다음은 static 폴더를 django와 연결시켜 주는 작업이 남았습니다. settings.py로 이동하셔서 아래와 같이 수정해주세요. 뒤에 콤마가 있으니 빼먹지 않도록 주의해주세요.

```
#...생략...

STATIC_URL = '/static/'
STATICFILES_DIRS = [
    BASE_DIR / 'static',
]
```

이제 index.html로 들어가 static 파일을 호출하는 code를 넣어주세요. `{% 문법 %}`의 구조는 template tag라고 불립니다. 자세한 내용은 Django 공식문서를 참고해주세요. 링크는 아래 달아두도록 하겠습니다.

Built-in template tags and filters | Django documentation | Django

This document describes Django's built-in template tags and filters. It is recommended that you use the automatic documentation , if available, as this will also include documentation for any custom tags or filters installed. The only exceptions

https://docs.djangoproject.com/en/3.0/ref/templates/builtins/

tutorialdjango/mysite/main/templates/main/index.html

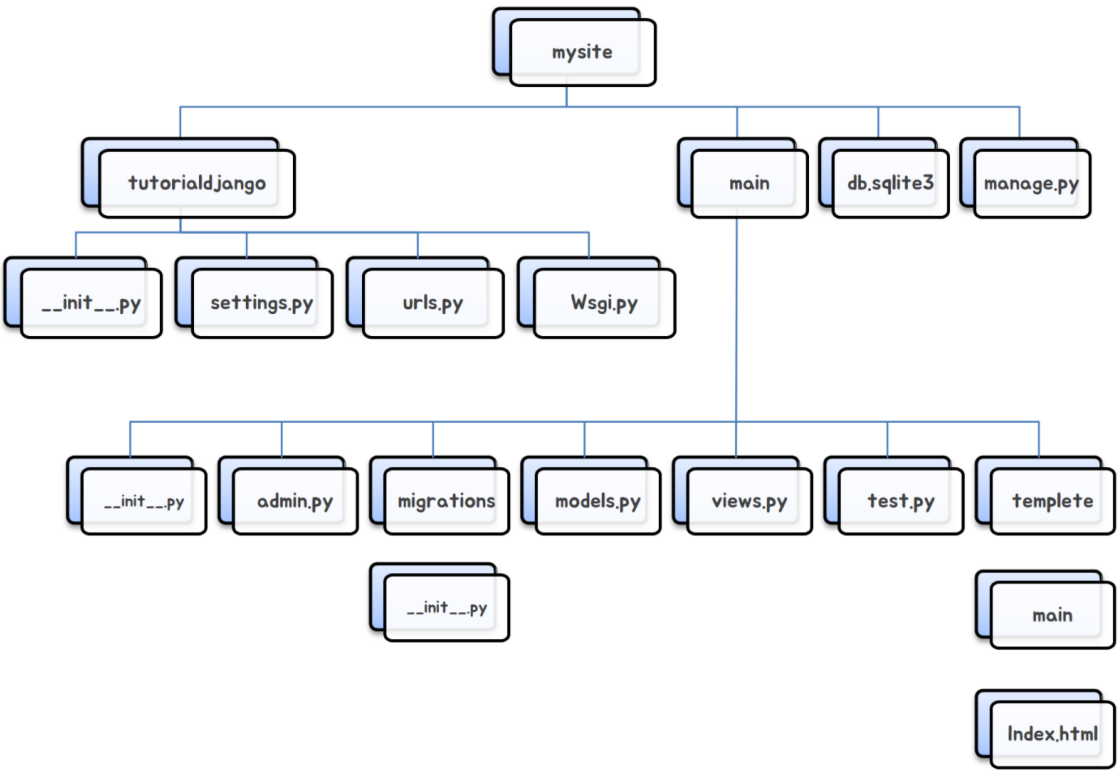
```
<html>
<head>
  <title>Django!</title>
</head>
<body>
  <h1>Test!</h1>
  {% load static %}
  <img src="{% static 'jeju.jpg' %}">
</body>
</html>
```

콘솔창에 python manage.py runserver 0:80이 입력이 안되어 있으시다면 입력하시고 프로젝트 > 실행URL과 포트를 누르신 다음 URL을 클릭하시면 아래와 같은 화면이 나옵니다.

Test!

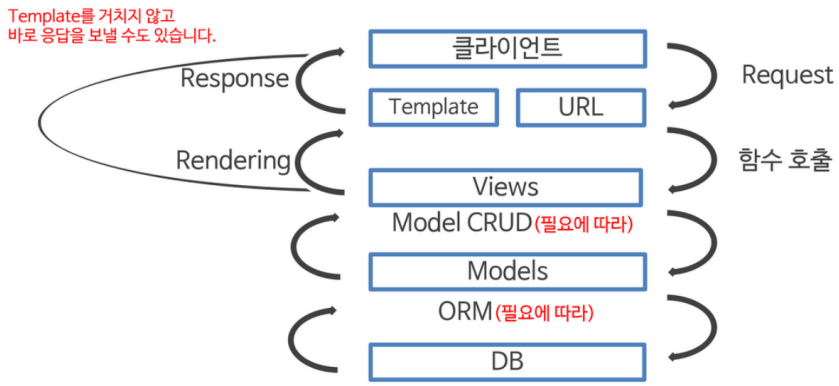
장고의 MTV 패턴

현재 폴더 구성 트리입니다. 우리가 수정한 순서는 urls.py > views.py > index.html 순입니다.



Django는 MVC모델을 MTV라고 부릅니다. 용어가 바뀌었을 뿐이지 기본적인 개념은 같습니다. 아래 표는 Django 작동원리를 도식화 한 것입니다.

MTV(빨간색으로 표시된 부분은 설계에 따라 작동하지 않을 수도 있습니다.)



Model – DB와 연결된 Python Class

Template – 사용자에게 response될 Client View

View – Django에서 처리한 데이터를 Template에게 전달

각 문자는 다음과 같이 표준 SQL 문으로 대응 가능합니다.

이름	조작	SQL
Create	생성	INSERT
Read(또는 Retrieve)	읽기(또는 인출)	SELECT
Update	갱신	UPDATE
Delete(또는 Destroy)	삭제(또는 파괴)	DELETE

https://ko.wikipedia.org/wiki/CRUD

CRUD는 대부분의 컴퓨터 소프트웨어가 가지는 기본적인 데이터 처리 기능인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 묶어서 일컫는 말입니다.

cafelist 페이지

이번에는 포스팅이 모아져 있는 cafelist 페이지를 만들어 보도록 하겠습니다. 우선 urls.py 파일을 수정하고 views.py파일을 수정합니다. 앞서 말씀드린 것처럼, 각각에 path 안에 name을 지정하면 url로 보다 쉽게 접근할 수 있습니다. 다만 코드의 복잡도가 올라가지 않도록, 마지막에 수정해줄게요.

- tutorialdjango/mysite/tutorialdjango/urls.py

```
from django.contrib import admin
from django.urls import path
from main.views import index, cafelist

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index),
    path('cafelist/', cafelist),
]
```

- tutorialdjango/mysite/main/views.py

```
from django.shortcuts import render

def index(request):
    return render(request, 'main/index.html')

def cafelist(request):
    return render(request, 'main/cafelist.html')
```

이제 html 파일을 만들 차례입니다. `cafelist.html` 파일을 아래 폴더에 만들어주세요.

- mysite/main/templates/main/cafelist.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
</body>
</html>
```

만약 중간에 한번 나갔다 오셨다면 아래 명령어처럼 쳐주세요. 또는 서버가 재실행 되어 맨 앞에 `(myvenv)`가 생략되어 있을 경우 아래 명령어를 꼭 쳐주세요.

```
root@goorm:/workspace/tutorialdjango# cd mysite

root@goorm:/workspace/tutorialdjango/mysite# source myvenv/bin/activate

(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py runserver 0:80
```

계속해서 진행하고 계셨다면 아래 코드만 입력하시면 됩니다.

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py runserver 0:80
```

`프로젝트` > `실행 URL과 포트`를 누르시고 URL을 클릭하신다음 뒤에 /cafelist라고 치시면 cafelist페이지가 나옵니다.

실행화면(https://tutorialdjango-bcrpr.run.goorm.io/cafelist/)

모델 작성

모델 클래스 작성

admin 페이지를 통한 모델요소 추가

모델 클래스 작성

이제 각각의 포스팅에 저장될 공간을 만들어보도록 하겠습니다. 우선 name과 content를 만듭니다. 사진을 저장할 공간은 다른 챕터에서 만들도록 하겠습니다.

여기서 `models.CharField()` 와 같은 값들을 Django model field라고 하는데요. 아래 문서에서 다양한 필드와 그 필드에 따른 옵션을 보실 수 있습니다. 하지만 기억하세요. 초보자에게 너무 많은 내용은 오히려 방해가 됩니다!

- Django model field

  Model field reference | Django documentation | Django

  A file-upload field. Note The primary_key argument isn't supported and will raise an error if used. Has two optional arguments: This attribute provides a way of setting the upload directory and file name, and can be set in two ways. In both

  https://docs.djangoproject.com/en/3.0/ref/models/fields/

- tutorialdjango/mysite/main/models.py

```
from django.db import models

class Cafe(models.Model):
    name = models.CharField(max_length=50)
    content = models.TextField()
```

터미널 창에서 `Ctrl + C`를 누르시고 빠져나오신 다음 아래 명령어를 입력해주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations main
Migrations for 'main':
    main/migrations/0001_initial.py
        -Create model Post
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
...
Running migrations:
    Applying main.0001_initial... OK
```

위처럼 나왔다면 성공한 것입니다. DB에 반영된 내용을 Admin page에도 보이도록 등록할 것입니다. 아래 파일에서 admin 사이트 설정을 할 수 있습니다.

- tutorialdjango/mysite/main/admin.py

```
from django.contrib import admin
from .models import Cafe

admin.site.register(Cafe)
```

하지만 조금더 어려운 코드를 사용하여 아래와 같이 사용하는 것도 가능합니다. **아래 코드는 실행시키지 마시고 참고**만 삼아 주세요.

admin page도 커스터마이징이 가능하다는 것을 보여드리기 위한 예제입니다.

```
#참고 코드 - 실행은 하지 마십시오.

from django.contrib import admin
from .models import Cafe

# 방법1
# admin.site.register(Cafe)

# 방법2 (create_at은 구분을 위해 위에 models.py에서 생성, 본 예제에서만 사용함.)
@admin.register(Cafe)
class CafeAdmin(admin.ModelAdmin):
    list_display = ['id', 'name', 'content', 'create_at']
    list_display_links = ['id', 'name']
```

참고 코드 실행페이지

admin 페이지를 통한 모델요소 추가

이제 Superuser를 만듭니다. Superuser는 게시물을 삭제, 수정, 저장할 수 있으며 다른 유저를 관리할 수 있습니다. 비밀번호가 `*` 모양으로 안뜨니 주의해주세요. 또한 비밀번호 길이가 짧거나, 이름이나 이메일과 유사하게 비밀번호를 설정하면 재입력 메시지가 나오게 되니 비밀번호는 특수문자 포함하여 8자 이상으로 입력해주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py createsuperuser
Username (leave blank to user 'root'): leehojun
Email address: leehojun@gmail.com
Password:
Password (again):
Superuser created successfully.
```

자, 이제 서버를 구동시키는 명령어를 입력해주세요. 이제 손에 많이 익으실겁니다.

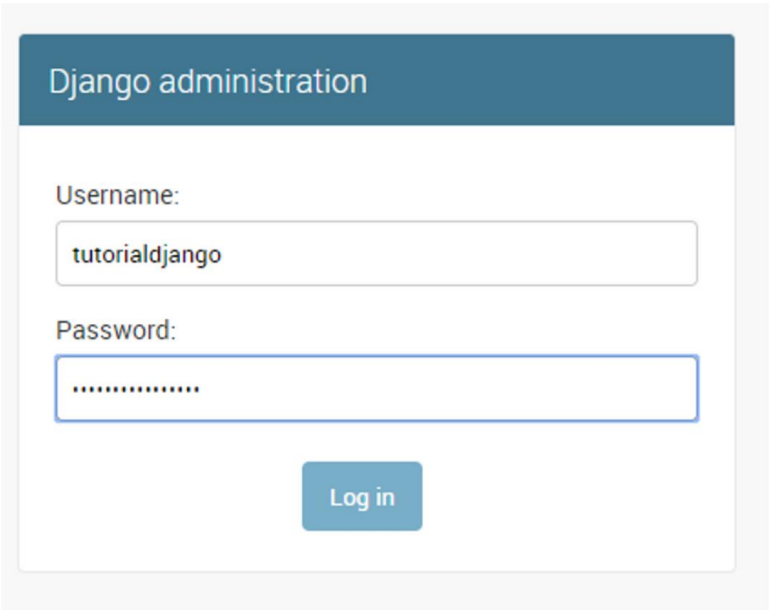
```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80
```

`프로젝트` > `실행 URL과 포트`를 누르시고 URL을 클릭하신다음 뒤에 /admin라고 치시면 admin페이지가 나옵니다. 아래 형식과 같겠죠.

URL : https://tutorialdjango-bcrpr.run.goorm.io/**admin/**

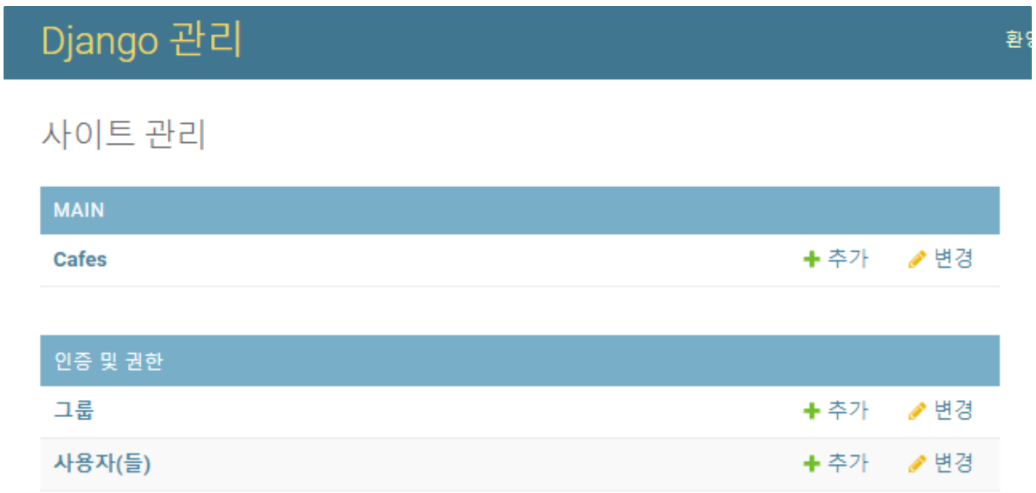
Admin 사이트 안으로 들어오셨다면 카페를 추가해보겠습니다.

1. 만든 superuser로 로그인 합니다.

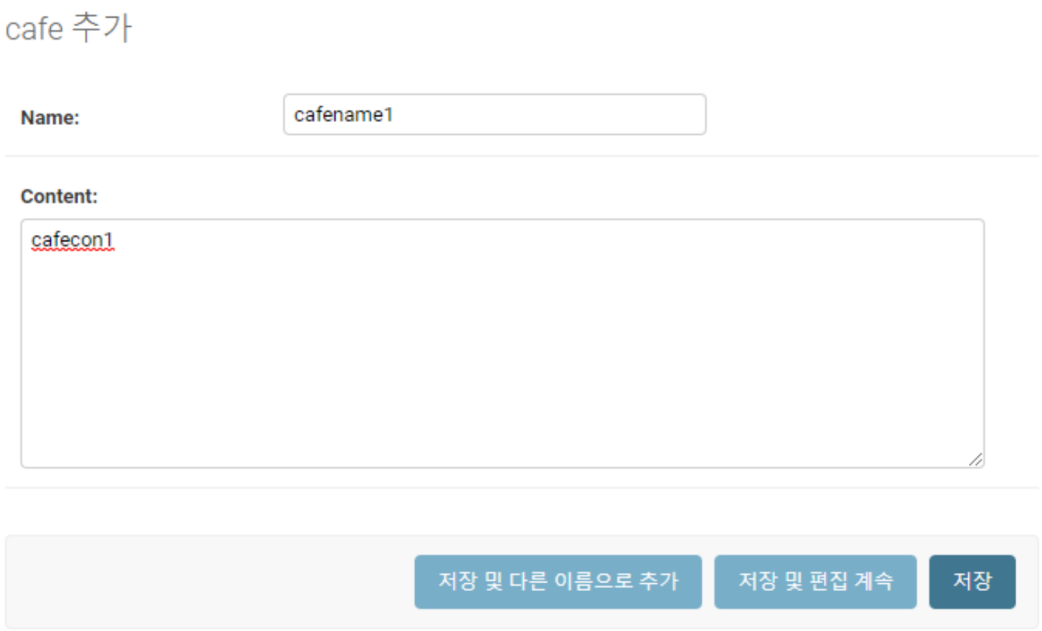


superuser의 아이디와 패스워드를 입력해주세요.

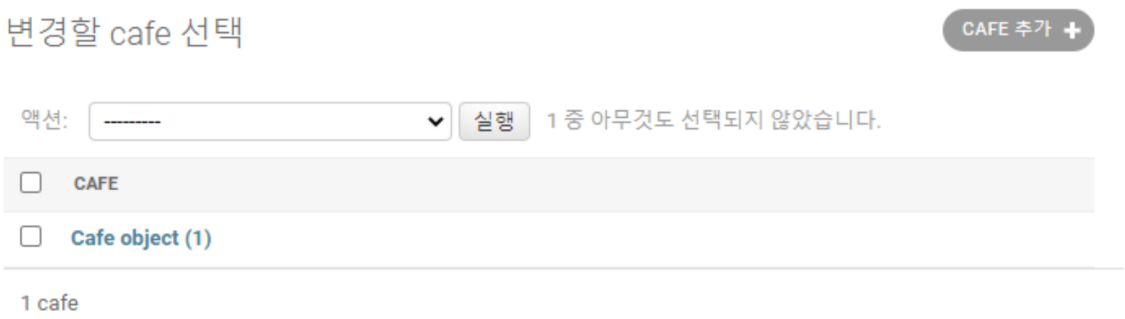
2. `Cafes`의 '추가' 버튼을 누릅니다.



3. 카페 이름과 내용을 작성합니다.



저장을 하면 우리가 입력한 `name`이 표시되는 것이 아니라 `Cafe object (1)`이러한 형식으로 보입니다.

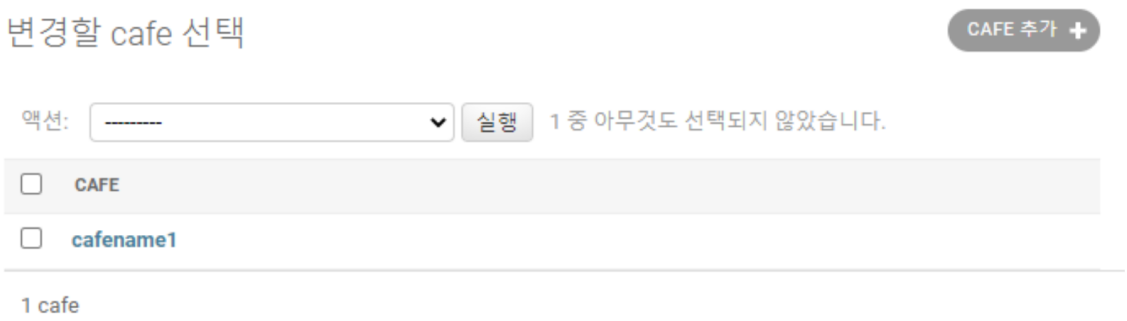


제목을 name으로 출력하라는 명령이 없었기 때문입니다. `models.py`로 들어가서 다시 제목이 무엇인지 설정해줄게요.

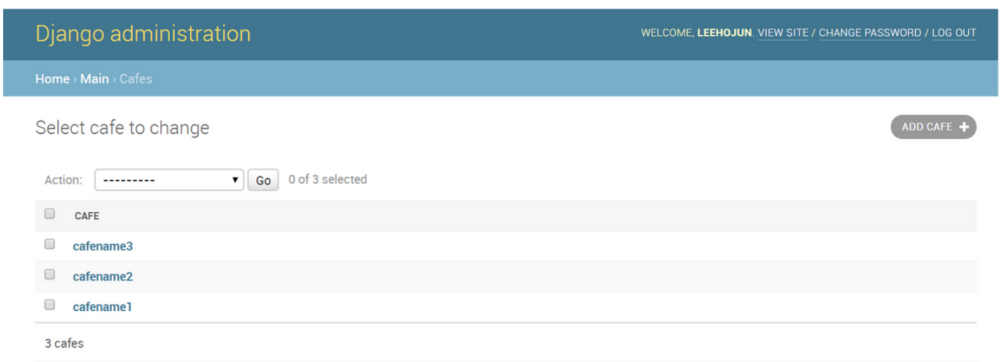
- mysite/main/models.py

```
from django.db import models

class Cafe(models.Model):
    name = models.CharField(max_length=50)
    content = models.TextField()

    def __str__(self):
        return self.name
```

브라우저 창에서 새로고침을 해보시면 이름이 바뀐 것을 볼 수 있습니다.



Test를 위해 3개의 게시물을 작성하였습니다. 이번에는 이 게시물을 cafelist 페이지에 띄어 보도록 하겠습니다. views 파일을 아래와 같이 수정해주세요.



tutorialdjango/mysite/main/views.py

```
from django.shortcuts import render
from .models import Cafe

def index(request):
    return render(request, 'main/index.html')


def cafelist(request):
    cafes = Cafe.objects.all()

    context = {
        'cafes': cafes
    }
    return render(request, 'main/cafelist.html', context)
```

Template tag를 사용하여 DB에 입력된 내용을 모두 가져옵니다. 템플릿 그와 랭귀지에 관련된 자세한 내용은 아래 링크를 참고해주세요. 위에도 언급이 되어 있지만, 다시 찾기가 불편하실 것 같아 아래 링크를 달아드립니다.

- tamplates language :

  The Django template language | Django documentation | Django

  This document explains the language syntax of the Django template system. If you're looking for a more technical perspective on how it works and how to extend it, see The Django template language: for Python programmers . Django's

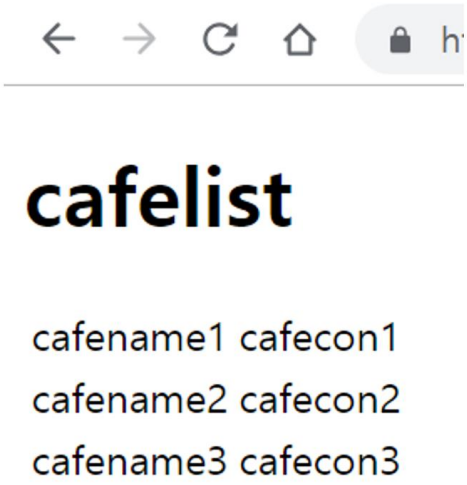
  https://docs.djangoproject.com/en/3.0/ref/templates/language/

- template tag :

  Built-in template tags and filters | Django documentation | Django

  This document describes Django's built-in template tags and filters. It is recommended that you use the automatic documentation , if available, as this will also include documentation for any custom tags or filters installed. The only

  https://docs.djangoproject.com/en/3.0/ref/templates/builtins/




```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
  <table>
    {% for cafe in cafes %}
    <tr>
      <td>{{ cafe.name }}</td>
      <td>{{ cafe.content }}</td>
    </tr>
    {% endfor %}
  </table>
</body>
</html>
```

소스코드보기를 하면 우리가 작성했던 템플릿 랭귀지가 보이지 않습니다. 이렇게 사용자에게 보여질 때에는 html 형식으로 넘겨주게 됩니다.

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
  <table>
    <tr>
      <td>cafename1</td>
      <td>cafecon1</td>
    </tr>
    <tr>
      <td>cafename2</td>
      <td>cafecon2</td>
    </tr>
    <tr>
      <td>cafename3</td>
      <td>cafecon3</td>
    </tr>
  </table>
</body>
</html>
```

cafedetails 페이지

[참고]

아래 명령어는 구름IDE에서 다시 접속하실 때마다 실행해 주셔야 하는 명령어 입니다.

```
root@goorm:/workspace/컨테이너명# cd mysite

root@goorm:/workspace/컨테이너명/mysite# source myvenv/bin/activate

(myvenv)root@goorm:/workspace/컨테이너명/mysite#
```

myvenv가 붙지 않은 상태에서 그동안 명령어를 치셨다면 **지금이라도 컨테이너를 삭제해버리시고 처음부터 다시 하시길 권장**해 드립니다. (myvenv)가 붙지 않은 환경, 붙은 환경은 완전히 다른 환경이기 때문입니다. 물론, 애러를 잡거나 폴더 몇 개 지우는 것으로도 끝날 수 있지만, 초급자일 때에는 반복학습도 중요할 뿐더러 에러를 잡는 것이 어려울 수 있으니까요.

cafedetails 페이지 만들기

이제 cafelist 페이지에서 카페를 눌렀을 때 나오는 상세 화면, cafedetails 페이지를 만들어 보도록 하겠습니다.

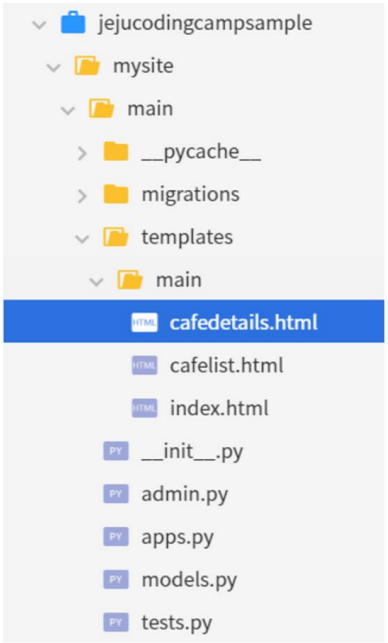
- tutorialdjango/mysite/tutorialdjango/urls.py

```
from django.contrib import admin
from django.urls import path
from main.views import index, cafelist, cafedetails

urlpatterns = [
    path('admin/',admin.site.urls),
    path('',index),
    path('cafelist/',cafelist),
    path('cafelist/<int:pk>',cafedetails),
]
```

- tutorialdjango/mysite/main/views.py

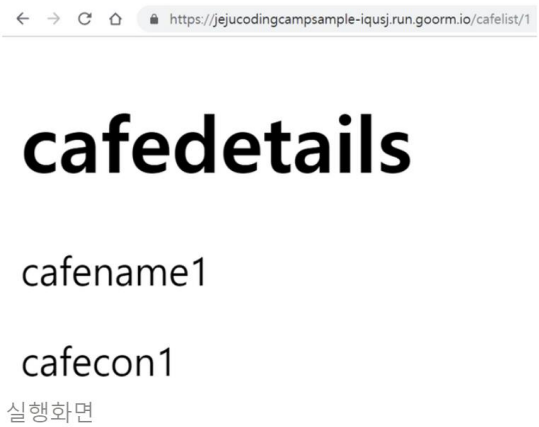
```
from django.shortcuts import render
from .models import Cafe

def index(request):
    return render(request, 'main/index.html')

def cafelist(request):
    cafelistobj = Cafe.objects.all()

    context = {
        'cafes':cafes
    }

    return render(request, 'main/cafelist.html', context)

def cafedetails(request, pk):
    cafe = Cafe.objects.get(pk=pk)

    context = {
        'cafe': cafe,
    }

    return render(request, 'main/cafedetails.html', context)
```

이제 아래 화면처럼 main 안에 템플릿은 3개가 됩니다. 마지막 템플릿입니다.



아래 URL 패턴처럼 blog/{post번호}를 입력하시면 아래 페이지처럼 출력이 됩니다.

URL : https://tutorialdjango-bcrpr.run.goorm.io/cafelist/1/



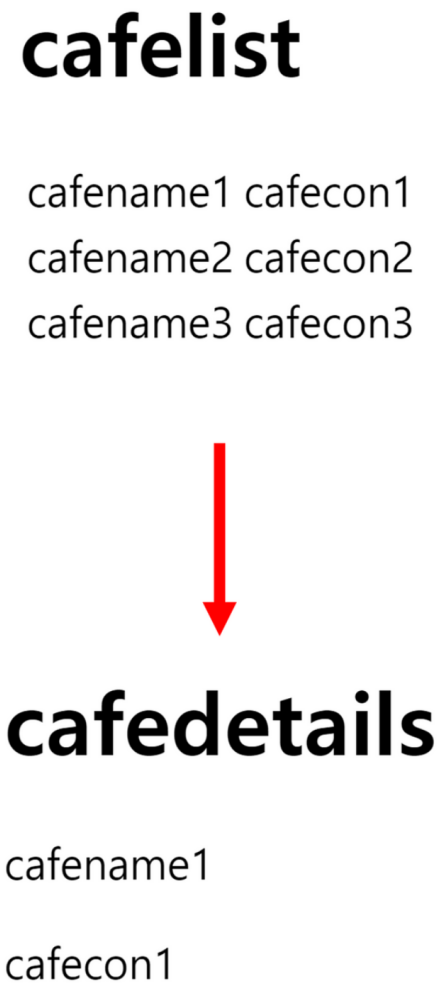
실행화면

간단하게 cafelist화면에서 cafe 제목과 내용을 클릭하였을 때 해당 cafedetails로 이동하도록 만들어 보도록 하겠습니다.

- tutorialdjango/mysite/main/templates/main/cafelist.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
  <table>
    {% for cafe in cafes %}
    <tr onclick="location.href='/cafedetails/{{ cafe.pk }}/'">
      <td>{{ cafe.name }}</td>
      <td>{{ cafe.content }}</td>
    </tr>
    {% endfor %}
  </table>
</body>
</html>
```

연결된 화면입니다. 해당 post 제목과 내용을 클릭하면 해당 포스팅으로 이동합니다.



이번에는 반대로 posting 상세 내용에서 목록으로 넘어가는 링크 만들어 보도록 하겠습니다.

- 컨테이너명/mysite/main/templates/main/cafedetails.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafedetails</h1>
  <p>{{cafe.name}}</p>
  <p>{{cafe.content}}</p>
  <a href="/cafelist/">목록</a>
</body>
</html>
```

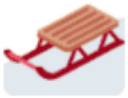
자, 이제 좀 더 고급진 방법을 알아보도록 하겠습니다. 이해를 돕기 위해 이 부분은 개인의 습득 속도에 따라 선택적으로 반영하도록 하겠습니다. (따라하지 않으셔도 괜찮습니다.)

1. urls.py 수정

```
from django.contrib import admin
from django.urls import path
from main.views import index, cafelist, cafedetails

urlpatterns = [
    path('admin/',admin.site.urls),
    path('',index),
    path('cafelist/',cafelist),
    path('cafelist/<int:pk>/',cafedetails, name='cafedetails'),
]
```

2. 각각의 html 파일 수정

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
  <table>
    {% for cafe in cafes %}
    <tr onclick="{% url 'cafedetails' cafe.id %}">
      <td>{{cafe.name}}</td>
      <td>{{cafe.content}}</td>
    <tr>
    {% endfor %}
</body>
</html>
```



이미지 넣기

모델에 이미지 필드 추가

Media 설정

모델에 이미지 필드 추가

cafe에 이미지를 넣어보겠습니다.

tutorialdjango > mysite > main > models.py로 이동하겠습니다. 여기서 null True를 주게 되면 기존에 있는 게시판 게시물은 null 값으로 들어가게 됩니다.

- tutorialdjango/mysite/main/models.py

```
from django.db import models

class Cafe(models.Model):
    name = models.CharField(max_length = 50)
    content = models.TextField()
    mainphoto = models.ImageField(blank=True, null=True)

    def __str__(self):
        return self.name
```

위와 같이 수정하셨다면 이제 사진을 처리할 수 있는 라이브러리를 설치 해야 합니다. `Pillow`라는 라이브러리이며 아래와 같이 설치할 수 있습니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip3 install pillow
```

모델의 변경사항을 DB에 반영합니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

서버는 다시 구동시켜주세요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80
```

Media 설정

이제 사진이 저장될 공간을 설정해주도록 하겠습니다. 이 설정을 하지 않으면 사진을 업로드 할 경우 mysite 바로 아래로 파일이 들어가게 됩니다. 아래 코드의 위치는 `settings.py`에 맨 마지막입니다.

- tutorialdjango/mysite/tutorialdjango/settings.py

```
from pathlib import Path

BASE_DIR = Path(__file__).resolve().parent.parent

#...중략...

MEDIA_ROOT = BASE_DIR / 'media'
MEDIA_URL = '/media/'
```

💡 django 3.1에서는 경로를 표현할 때 os.Path 보다 pathlib.Path를 사용합니다. pathlib.Path를 사용하면 경로를 더 간편하게 표현할 수 있습니다.
os.Path 사용시 : `MEDIA_ROOT = os.path.join(BASE_DIR, 'media')`
pathlib.Path 사용시 : `MEDIA_ROOT = BASE_DIR / 'media'`

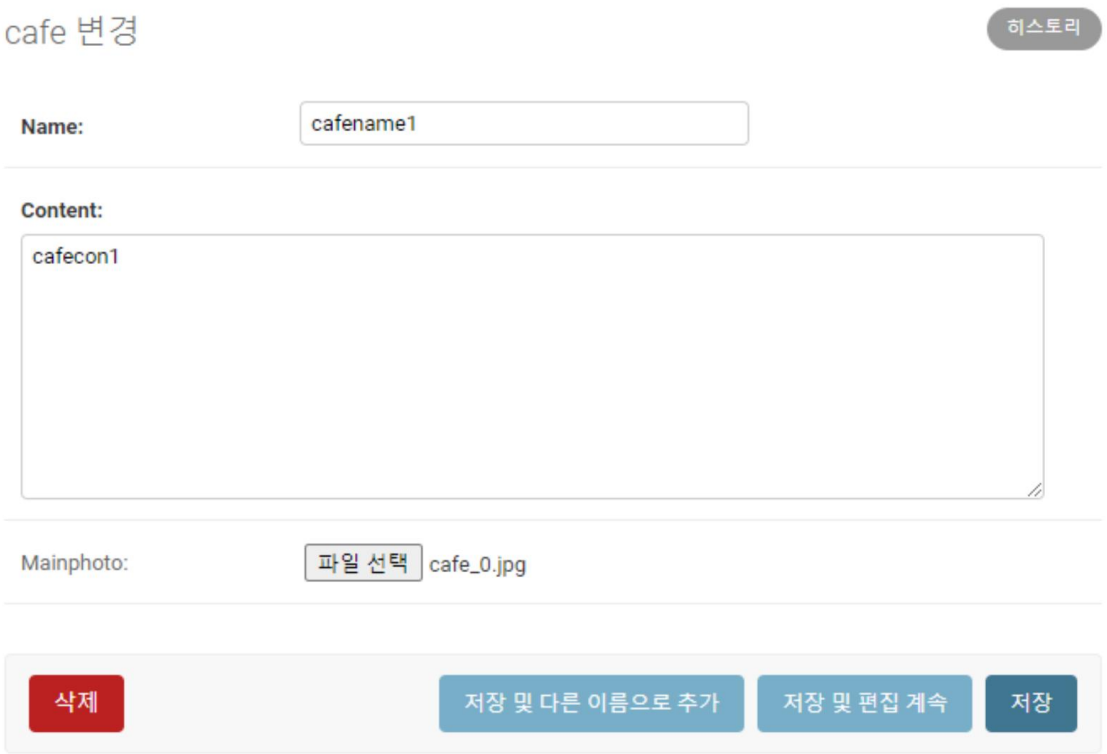
이미지 파일은 총 3개를 올려보도록 하겠습니다.

cafe_0

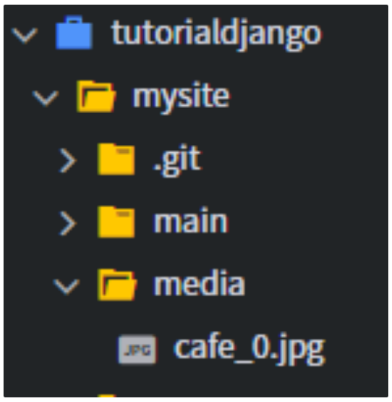
cafe_1

cafe_2

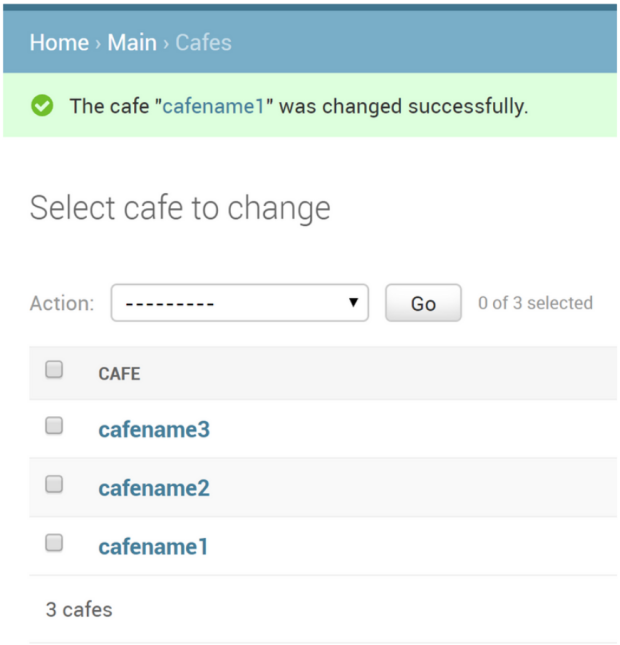
순서대로 cafe_0부터 올리겠습니다.



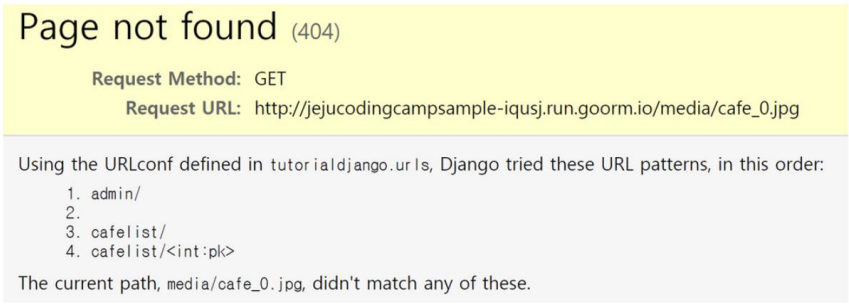
사진을 등록하면 `media` 라는 폴더 안에 이미지 파일이 업로드 된 것을 확인할 수 있습니다.



업로드한 파일을 확인하기 위해 해당 cafe를 클릭합니다.

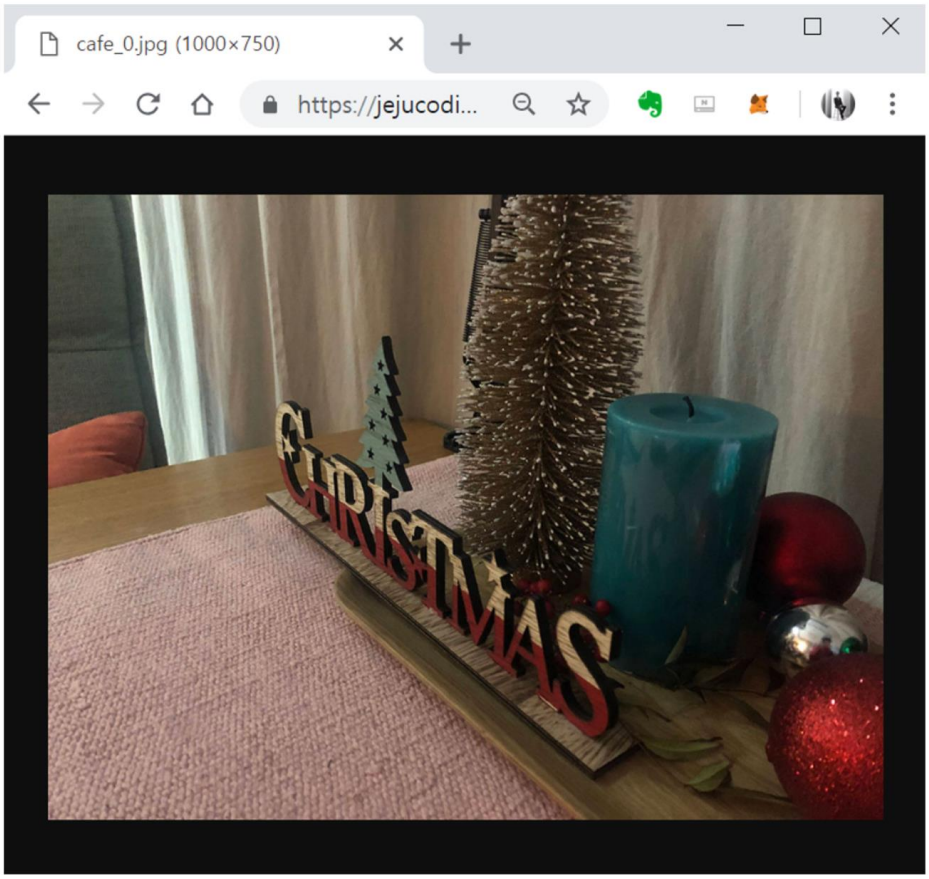


아래 `Mainphoto`에서 `cafe_0.jpg`를 클릭하니 page를 찾을 수 없다고 뜹니다. 우리가 `urls.py`에서 URL을 설정해주었었죠? 이러한 이미지도 URL이 필요합니다.



이번에는 이미지 URL을 설정하러 가보도록 하겠습니다. `urls.py`에서 아래와 같이 urlpatterns를 넣어주세요. 그러면 이미지가 정상적으로 보입니다.

```
from django.contrib import admin
from django.urls import path
from main.views import index, cafelist, cafedetails
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/',admin.site.urls),
    path('',index),
    path('cafelist/',cafelist),
    path('cafelist/<int:pk>/',cafedetails),
]

urlpatterns +=static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT
```



이제 나머지 두 사진도 넣은 뒤, 이번에는 cafedetails 페이지에서 해당 이미지를 불러와 보도록 하겠습니다. 파일 접근에 대한 자세한 내용은 홈페이지 링크를 참고해주세요.

Managing files | Django documentation | Django

This document describes Django's file access APIs for files such as those uploaded by a user. The lower level APIs are general enough that you could use them for other purposes. If you want to handle "static files" (JS, CSS, etc.), see Managing static files

https://docs.djangoproject.com/en/3.0/topics/files/

tutorialdjango/mysite/main/templates/main/postdetails.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafedetails</h1>
  <p>{{cafe.name}}</p>
  <p>{{cafe.content}}</p>
    {% if cafe.mainphoto %}
      <img src="{{ cafe.mainphoto.url }}">
    {% endif %}
  <a href="/cafelist/">목록</a>
</body>
</html>
```

cafedetails

cafename1

cafecon1

목록

댓글기능 추가하기

disqus

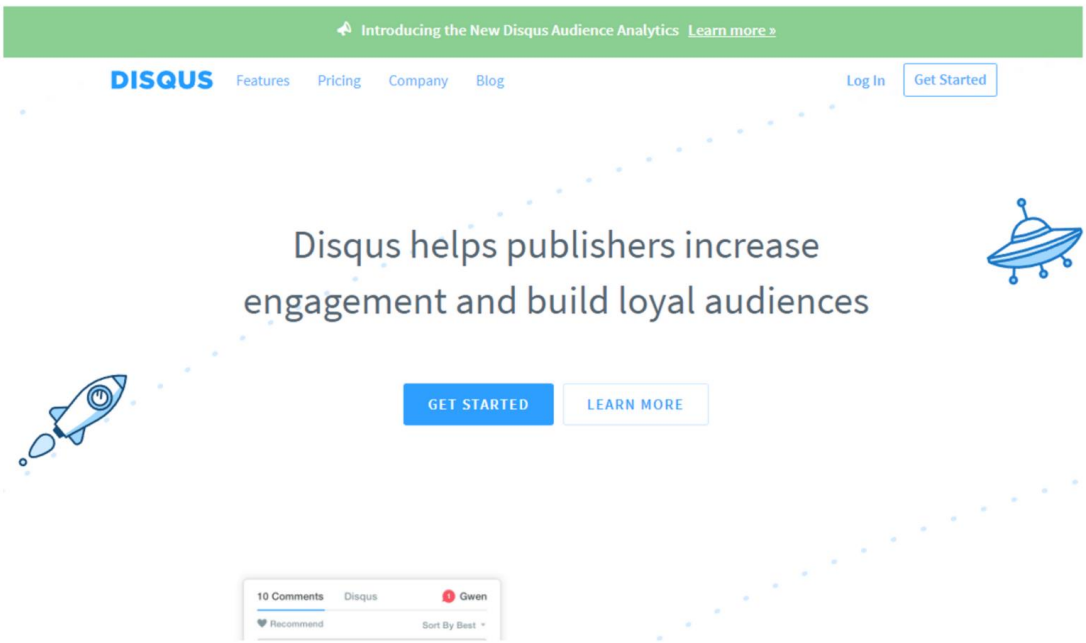
이제 댓글을 달 수 있도록 설정해 보도록 하겠습니다. 아래 명령어를 사용하여 disqus를 Django내에 설치할 수도 있지만 우리는 좀 더 편한 방법을 사용해 보도록 하겠습니다.

disqus 설치하기

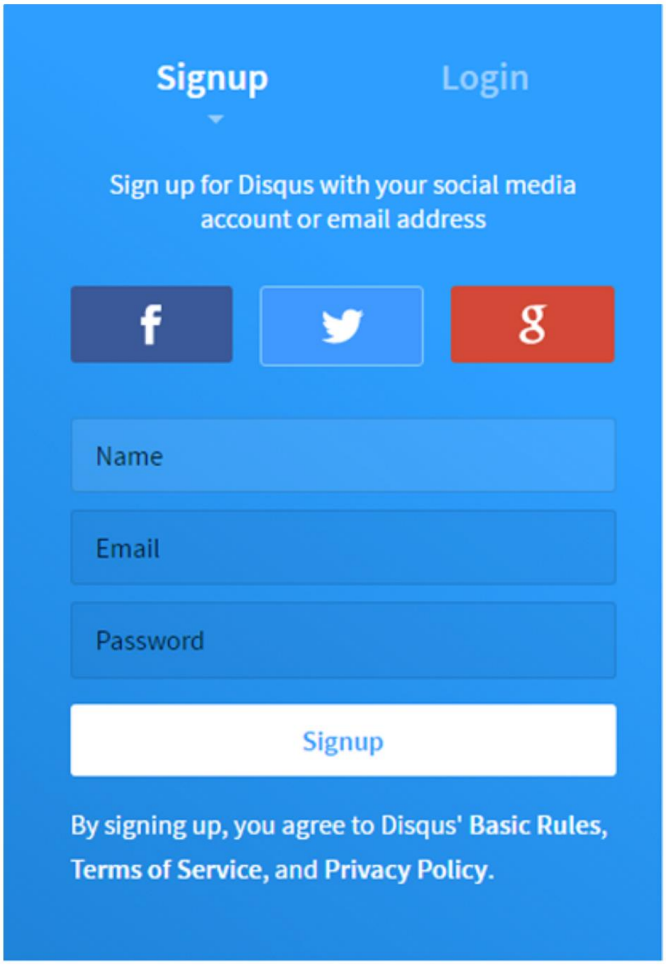
```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install django-disqus
```

* 우리는 설치하지 않는 방법을 사용하도록 하겠습니다. 따라서 위 명령어를 사용하지는 않습니다. 그러나 설치하셔서 사용하시는 분도 계시기 때문에 명시해 두도록 하겠습니다.

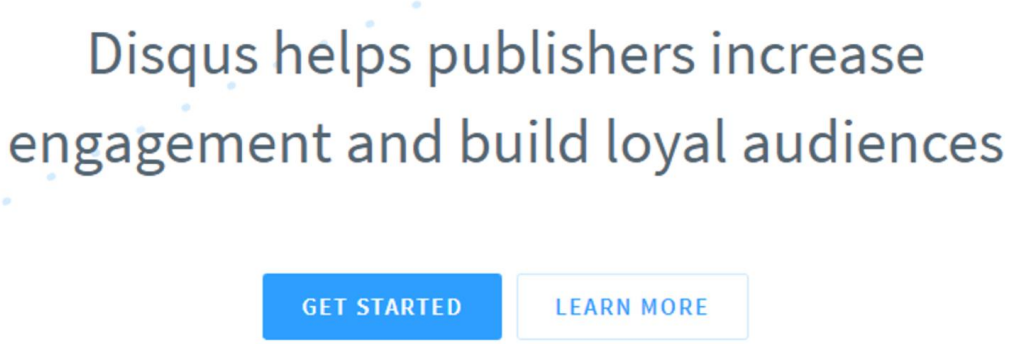
1. disqus 사이트에 접속합니다. https://disqus.com/



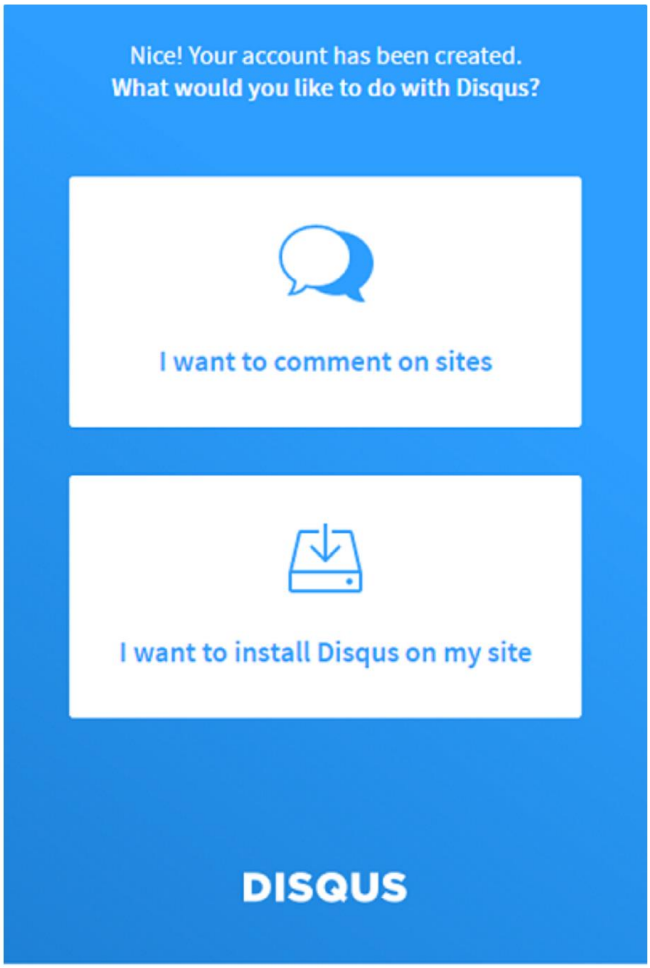
2. login버튼을 누르시고 회원 가입을 해주세요. Name, E-mail 등은 관리하기 쉽도록 구름 IDE ID, PW를 사용하시는 것을 권장합니다.



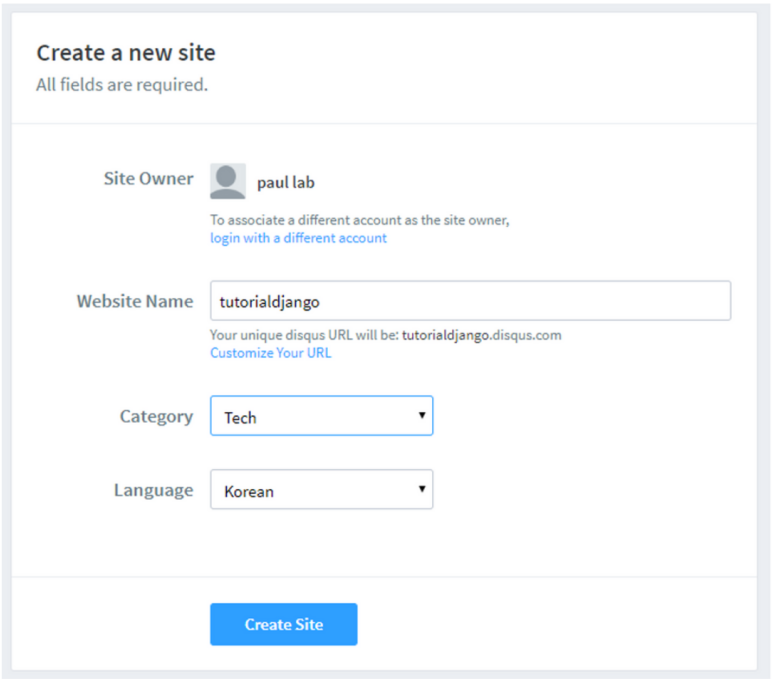
3. 자, 이제 `GET STARTED`를 눌러주세요.

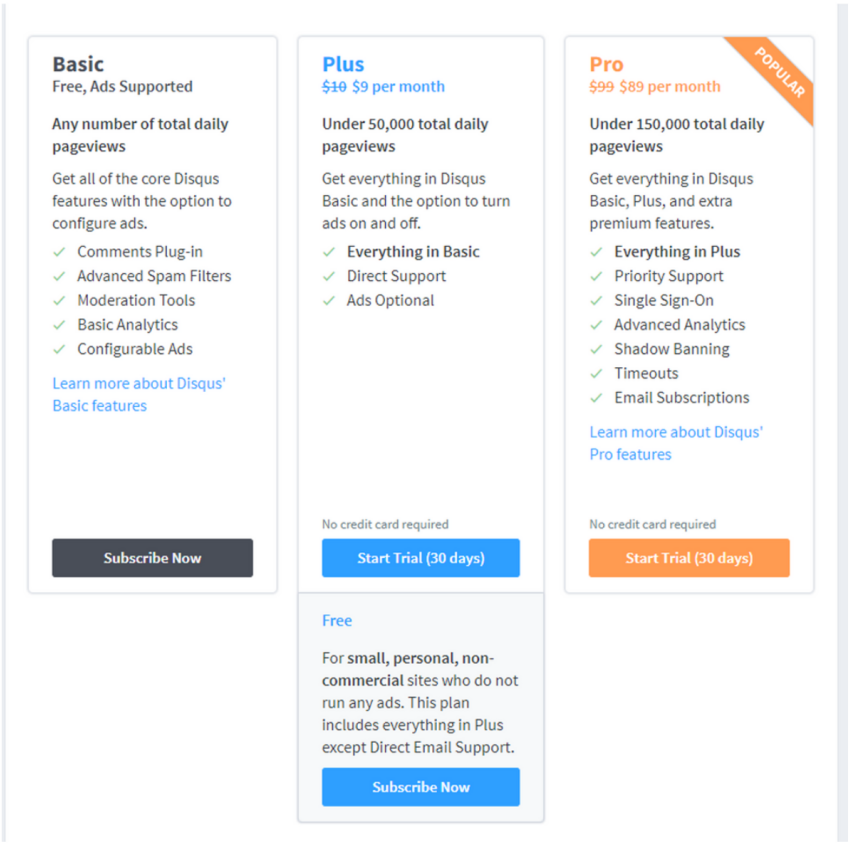


4. `I want to install Disqus on my site`를 클릭해주세요.

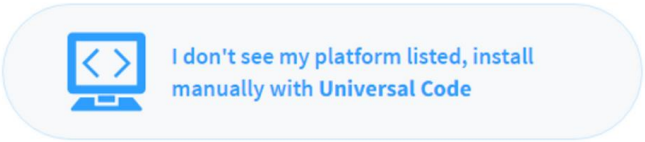


5. 아래 내용을 작성하신 다음(작성하는 내용이 중요하지는 않습니다.) Create Site를 눌러주세요. 우리는 BASIC으로 설치를 하도록 하겠습니다.



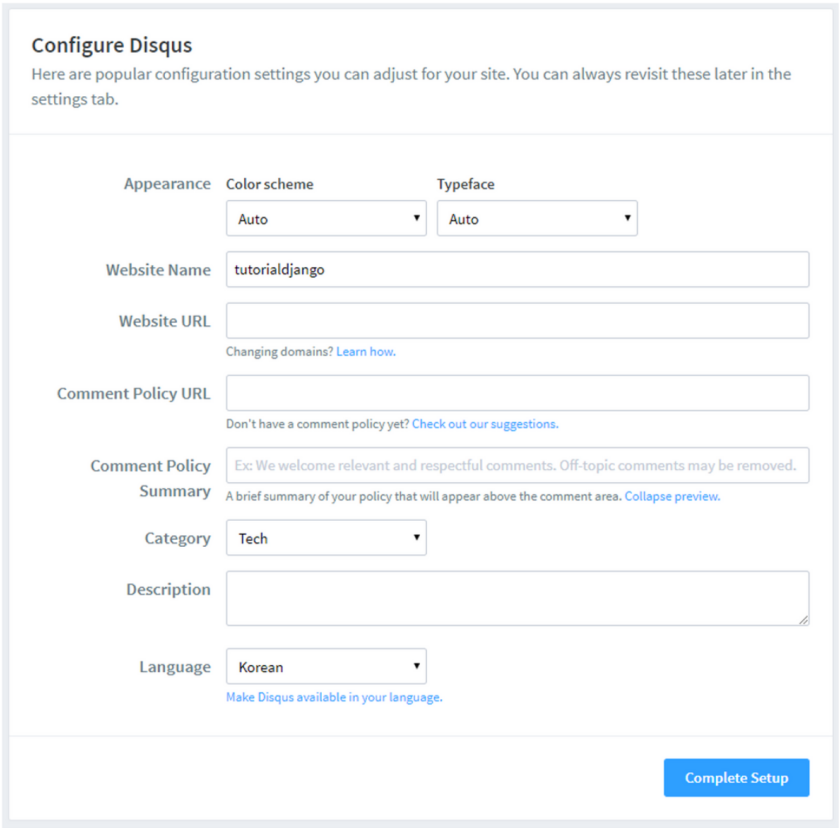


6. 다음을 누르시면 여러 플랫폼이 나오는데 맨 아래 있는 것을 클릭하겠습니다.



7. 그런 다음 1번이 나와있는 모든 소스를 복사해두세요. 이 스크립트를 HTML 파일에 붙여넣으면 원하는 곳에 댓글창이 뜨게 됩니다! (참 쉽죠? 😉)

8. 마지막에 작성하라고 나오는 내용은 작성하지 않으시고 `Complete Setup`을 눌러주세요.

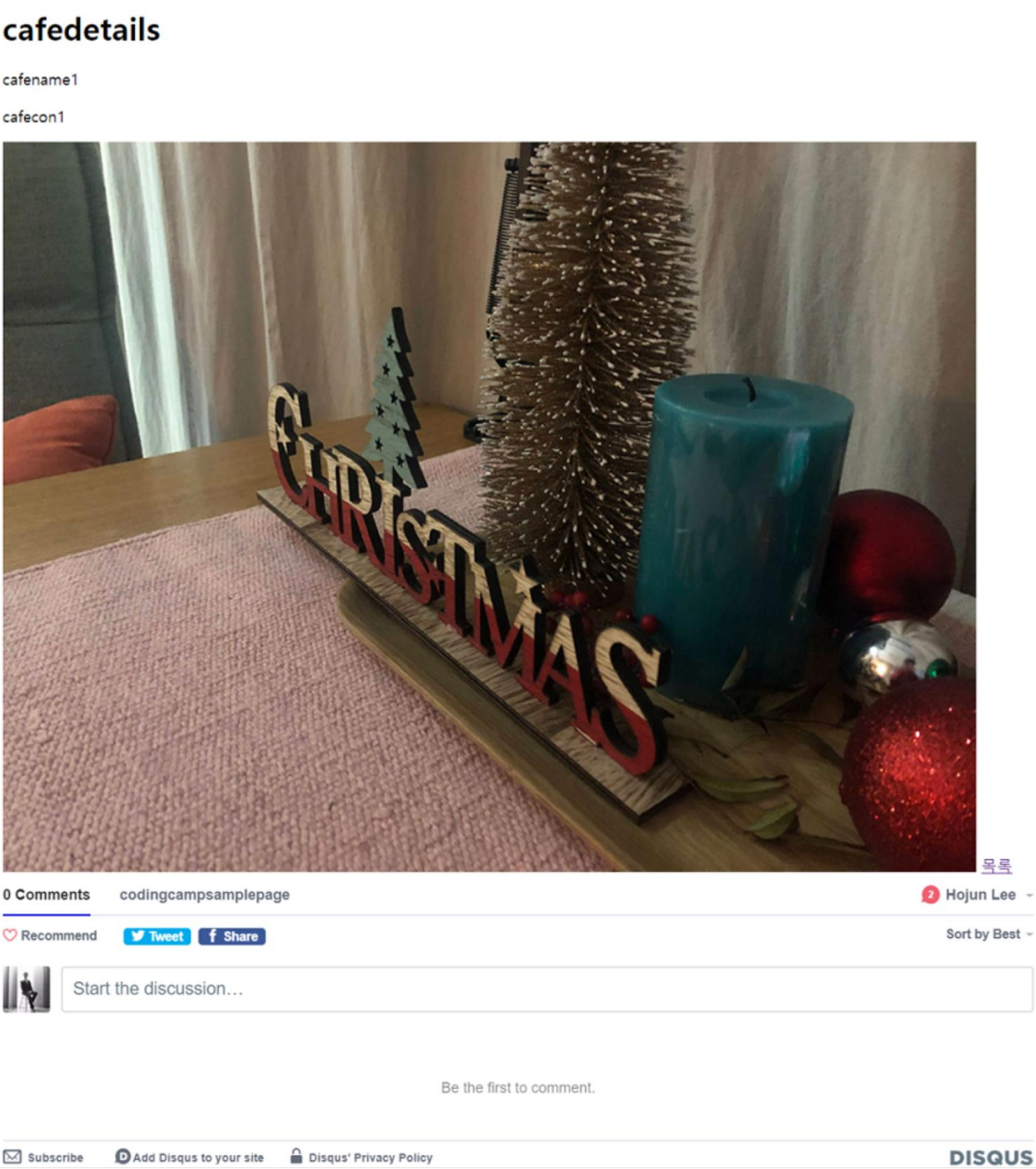


9. `cafedetails.html`로 오셔서 body 닫기 태그 위에 해당 소스를 붙여주세요. 주석은 보기 편하게 지웠습니다.

   tutorialdjango/mysite/main/templates/main/cafedetails.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafedetails</h1>
  <p>{{cafeobj.name}}</p>
  <p>{{cafeobj.content|linebreaks}}</p>
  {% if cafeobj.mainphoto %}
    <img src="{{ cafeobj.mainphoto.url }}">
  {% endif %}
  <a href="https://jejucodingcampsample-iqusj.run.goorm.io/cafelist>목록</ㅁ>
  <div id="disqus_thread"></div>
<script>
(function(){
var d= document, s = d.createElenment('script');
s.src = 'https://codingcampsamplepage.disqus.com/embed.js';
s.setAttribute('data-timestamp', + new Date());
(d.head || d.body).appendChild(s);
})();
</script>
<noscript>Plaese enable JavaScript to view the <a href="https://disqus.com/?ref_nosc
ript">
comments powered by Disqus.</a></noscript>
</body>
</html>
```

실행된 화면입니다. 목록 아래 댓글창이 달립니다.



게시글 작성일/수정일 추가

모델 수정

이번에는 게시된 날짜와 수정된 날짜를 추가해 보도록 하겠습니다.

Cafe 모델에 `published_date` 와 `modified_date` 필드를 추가합니다.

- tutorialdjango/mysite/main/models.py

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)
    mainphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()

    def __str__(self):
        return self.name
```

`DateTimeField`에서 `auto_now_add=True` 옵션을 주면 모델 개체 생성 시간이 자동으로 저장되고, `auto_now=True` 옵션은 모델 개체 생성 뿐 아니라 수정 시에도 시간이 자동으로 저장됩니다.

다음의 `makemigrations` 명령어 수행 시 사진과 같이 두가지 선택 사항이 나옵니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations
```

```
(myvenv) root@goorm:/workspace/tutorialdjango/mysite# python manage.py makemigrations
You are trying to add the field 'published_date' with 'auto_now_add=True' to cafe without a default;
the database needs something to populate existing rows.

 1) Provide a one-off default now (will be set on all existing rows)
 2) Quit, and let me add a default in models.py
```

원래 저장되어있던 1~3번 카페의 published_date 값을 정해야 하기 때문입니다.

- 1번은 현재 저장되어 있는 요소들의 `published_date` 값을 정하는 것
- 2번은 `models.py`에 돌아가서 `published_date`의 기본값을 설정하는 것

우리는 1번 사항을 선택합니다.

```
Please enter the default value now, as valid Python
You can accept the default 'timezone.now' by pressing 'Enter' or you can provide another value.
The datetime and django.utils.timezone modules are available, so you can do e.g. timezone.now
Type 'exit' to exit this prompt
[default: timezone.now] >>> timezone.now
```

그러면 넣을 값을 정하라고 나오는데 default가 `timezone.now` 즉 현재시간입니다. 그냥 엔터를 치거나 `timezone.now`를 입력하고 엔터를 칩니다. 그러면 현재 저장되어 있는 카페들의 `published_date` 값은 현재 시간으로 저장됩니다.

```
Migrations for 'main':
  main/migrations/0006_auto_20210115_1551.py
    - Add field modified_date to cafe
    - Add field published_date to cafe
```

💡 `modified_date`의 경우 신경쓰지 않아도 됩니다. 수정 때마다 자동으로 현재 시간이 저장되기 때문입니다.

이제 `migrate` 명령까지 마칩니다.

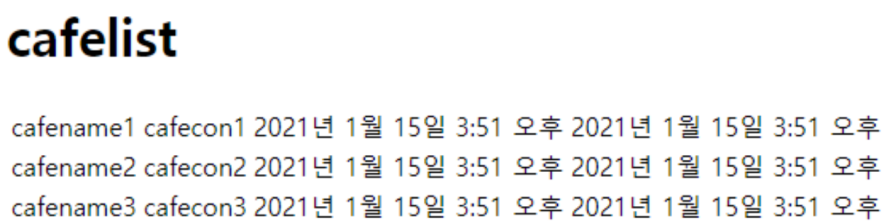
```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
...
Running migrations:
  Applying_main.0003_auto_20200720_0707.py... OK
```

날짜 출력

cafelist 페이지에서 cafe들의 생성일, 수정일을 표시해 보겠습니다.

- tutorialdjango/mysite/main/templates/main/cafelist.html

```
<!DOCTYPE html>
<html>
<head>
  <title>cafelist</title>
</head>
<body>
  <h1>cafelist</h1>
  <table>
    {% for cafe in cafes %}
    <tr onclick="location.href='/cafelist/{{ cafe.pk }}'">
      <td>{{ cafe.name }}</td>
      <td>{{ cafe.content }}</td>
      <td>{{ cafe.published_date }}</td>
      <td>{{ cafe.modified_date }}</td>
    </tr>
    {% endfor %}
  </table>
</body>
</html>
```



카페 위치 추가

카페 위치이름 추가

```
['한경면', '한림읍', '애월읍', '제주시', '조천읍', '구좌읍', '우도면', '성산읍', '표선면',
'남원읍', '서귀포시', '안덕면', '대정읍']
```

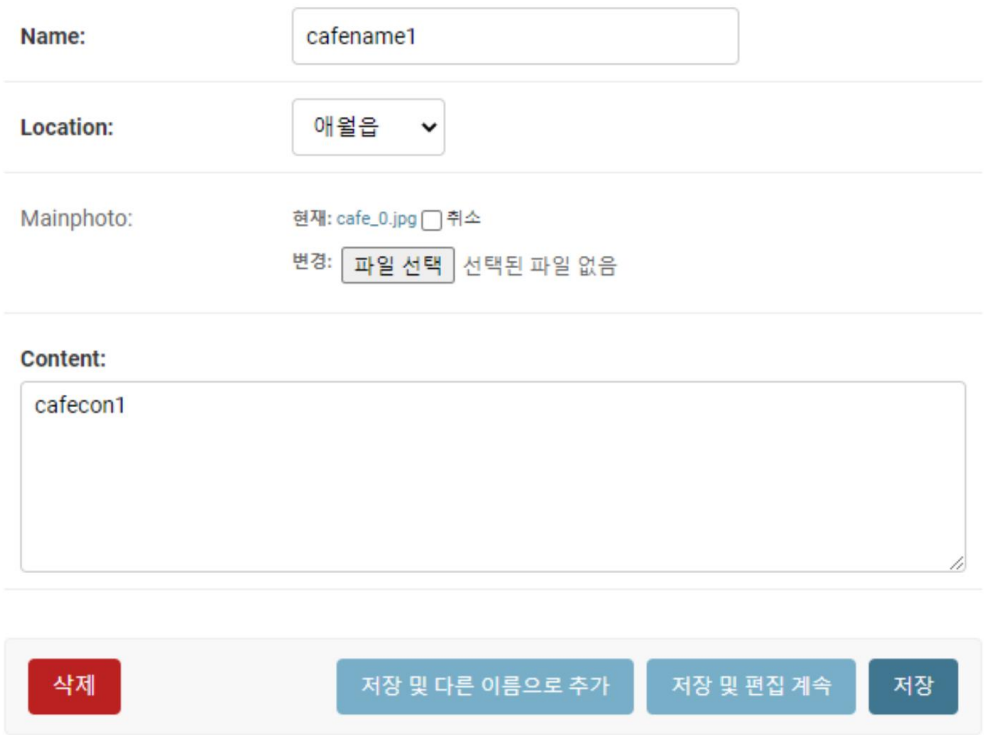
각 카페마다 카페 위치에 맞는 지명을 추가하려고 합니다.

Cafe 모델을 다음과 같이 수정합니다.

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)

    locations = [
        ('Hangyeong-myeon', '한경면'),
        ('Hallim-eup', '한림읍'),
        ('Aewol-eup', '애월읍'),
        ('Jeju-si', '제주시'),
        ('Jocheon-eup', '조천읍'),
        ('Gujwa-eup', '구좌읍'),
        ('Daejeong-eup', '대정읍'),
        ('Andeok-myeon', '안덕면'),
        ('Seogwipo-si', '서귀포시'),
        ('Namwon-eup', '남원읍'),
        ('Pyoseon-myeon', '표선면'),
        ('Seongsan-eup', '성산읍'),
        ('Udo-myeon', '우도면'),
    ]

    location = models.CharField(max_length=50, choices=locations)
    mainphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()

    def __str__(self):
        return self.name
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

이후 admin 페이지를 이용해 cafename1 ~ cafename3 까지 차례로 애월읍, 제주시, 조천읍으로 수정하겠습니다.



모델에 정보 추가하기

subphoto 추가

이번에는 사진 한장을 더 추가할 수 있도록 models.py 파일을 수정하겠습니다. 마찬가지로 `makemigrtions`와 `migrate`를 입력해주세요.

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)

    locations = [
        ('Hangyeong-myeon', '한경면'),
        ('Hallim-eup', '한림읍'),
        ('Aewol-eup', '애월읍'),
        ('Jeju-si', '제주시'),
        ('Jocheon-eup', '조천읍'),
        ('Gujwa-eup', '구좌읍'),
        ('Udo-myeon', '우도면'),
        ('Seongsan-eup', '성산읍'),
        ('Pyoseon-myeon', '표선면'),
        ('Namwon-eup', '남원읍'),
        ('Seogwipo-si', '서귀포시'),
        ('Andeok-myeon', '안덕면'),
        ('Daejeong-eup', '대정읍'),
    ]

    location = models.CharField(max_length=50, choices=locations)
    lat = models.FloatField(null =True)
    lng = models.FloatField(null=True)
    mainphoto = models.ImageField(blank=True, null=True)
    subphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()

    def __str__(self):
        return self.name
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

세 카페에 모두 subphoto로 다음 사진을 넣었습니다.

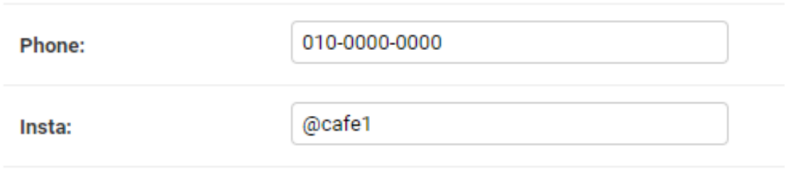
전화번호 & 인스타 계정

이번에는 cafelist에서 필요한 항목들을 추가해 보도록 하겠습니다. 마찬가지로 makemigrtions와 migrate를 입력해주세요.

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)

    locations = [
        ('Hangyeong-myeon', '한경면'),
        ('Hallim-eup', '한림읍'),
        ('Aewol-eup', '애월읍'),
        ('Jeju-si', '제주시'),
        ('Jocheon-eup', '조천읍'),
        ('Gujwa-eup', '구좌읍'),
        ('Udo-myeon', '우도면'),
        ('Seongsan-eup', '성산읍'),
        ('Pyoseon-myeon', '표선면'),
        ('Namwon-eup', '남원읍'),
        ('Seogwipo-si', '서귀포시'),
        ('Andeok-myeon', '안덕면'),
        ('Daejeong-eup', '대정읍'),
    ]

    location = models.CharField(max_length=50, choices=locations)
    lat = models.FloatField(null =True)
    lng = models.FloatField(null=True)
    mainphoto = models.ImageField(blank=True, null=True)
    subphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()
    phone = models.CharField(max_length=20)
    insta = models.CharField(max_length=20)

    def __str__(self):
        return self.name
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

makemigrations 시에 기존 카페들의 값은 임시로 아무 문자열을 넣고, 완료된 후 Admin 창에서 원하는 값들을 넣습니다.



공식문서에 보시면 blank와 null에 관련된 내용이 있습니다. blank는 사용자가 입력하더라도 비어도 괜찮다는 뜻이고, null은 DB가 비어있어도 된다는 옵션입니다.

Model field reference | Django documentation | Django

A file-upload field. Note The primary_key argument isn't supported and will raise an error if used. Has two optional arguments: This attribute provides a way of setting the upload directory and file name, and can be set in two ways. In both cases, the value

https://docs.djangoproject.com/en/3.0/ref/models/fields/

null

Field.null

If **True**, Django will store empty values as **NULL** in the database. Default is **False**.

Avoid using **null** on string-based fields such as **CharField** and **TextField**. If a string-based field has **null=True**, that means it has two possible values for "no data": **NULL**, and the empty string. In most cases, it's redundant to have two possible values for "no data;" the Django convention is to use the empty string, not **NULL**. One exception is when a **CharField** has both **unique=True** and **blank=True** set. In this situation, **null=True** is required to avoid unique constraint violations when saving multiple objects with blank values.

For both string-based and non-string-based fields, you will also need to set **blank=True** if you wish to permit empty values in forms, as the **null** parameter only affects database storage (see **blank**).

Note

When using the Oracle database backend, the value **NULL** will be stored to denote the empty string regardless of this attribute.

blank

Field.blank

If **True**, the field is allowed to be blank. Default is **False**.

Note that this is different than **null**. **null** is purely database-related, whereas **blank** is validation-related. If a field has **blank=True**, form validation will allow entry of an empty value. If a field has **blank=False**, the field will be required.

템플릿 적용 및 검색 기능 구현

템플릿 가져오기

템플릿 상속

about 페이지

선택 지역 카페 리스트 보여주기

카페 이름 검색기능 구현

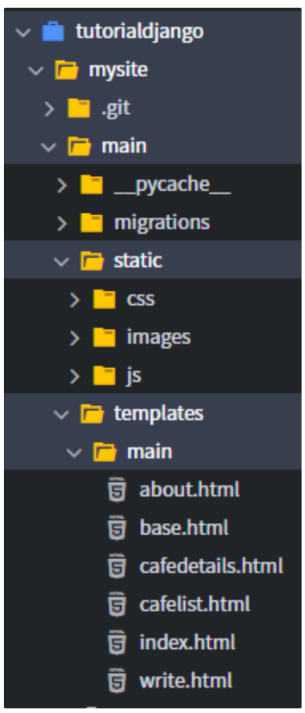
템플릿 가져오기

이제 전에 들었던 수업의 UI를 Django에 입히는 작업이 남아 있습니다. 우선 아래 링크에서 파일을 다운로드 받아 아래처럼 수행해주세요.

* 모든 templates 소스는 http://www.paullab.co.kr/django_3.1(ver.2021).zip

파일을 다운 받으시면 `static` 폴더와 `templates` 폴더가 있습니다. 이를 `main` 폴더 아래에 넣어줍니다.

* 우클릭 하여 '파일 가져오기/업로드' 버튼을 누르신 다음 파일을 선택하셔서 업로드하시면 됩니다.



템플릿 상속

이번에는 템플릿 상속에 대해 알아보도록 하겠습니다. 예를 들어 메뉴바가 바뀌었다 가정한다면, 그동안 편집해두었던 `.html` 파일을 모두 수정해야 하는 상황이 발생됩니다.

이러한 상황을 막기 위해 Django에서는 템플릿 상속을 지원합니다. 하나의 템플릿을 만들어 여러 개의 템플릿에서 상속받을 수 있도록 하는 것이죠. 위에서 들었던 예로, 메뉴를 하나의 템플릿으로 만들어 다른 파일이 상속할 수 있게 한다면 단 1개의 파일, 메뉴 파일만 수정하면 됩니다.

```
내용

{% block content %}
{% endblock %}

내용
```

상속할 템플릿

```
{% extends "main/base.html "%}

{% block content %}
내용
{% endblock content %}
```

상속받을 템플릿

about 페이지

이번에는 about page를 만들어 보도록 하겠습니다.

1. about.html 파일 URL 연결
2. write.html 파일 URL 연결

이제 좀 익숙하시죠? 😉 익숙하시니! 지난번에 언급해드린 것처럼, 다른 코드를 조금 수정하도록 하겠습니다.

views.py

```
from django.contrib import admin
from django.urls import path
from main.views import index, about, cafelist, cafedetails
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index, name='index'),
    path('about/', about, name='about'),
    path('cafelist/', cafelist, name='cafelist'),
    path('cafelist/<int:pk>', cafedetails, name='cafedetails'),
]

urlpatterns += static(setings.MECIA_URL, document_root=settings.MEDIA_ROOT)
```

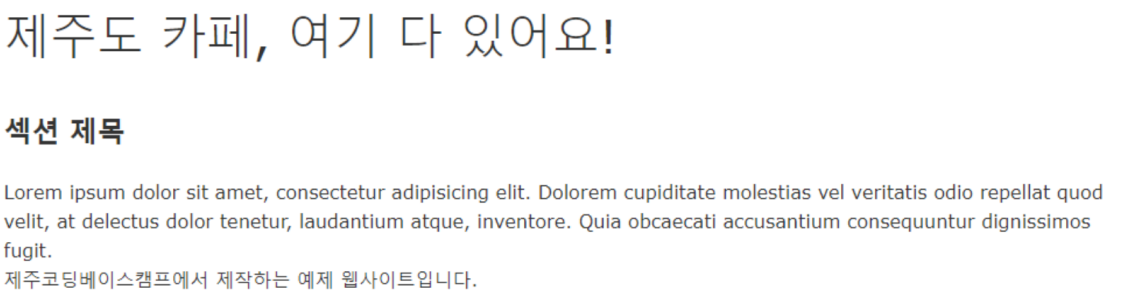
추가된 path 말고도, 다른 path들에 name 속성이 붙었죠? 이렇게 하면 지난번에 한 것처럼 URL을 하드코딩하는 것이 아니라, URL이 변경되어도 아래처럼 유동적으로 사용할 수 있답니다.

```
<a href="{% url 'index' %}">클릭하세요!</a>
```

어때요? 훨씬 편하죠? 하지만 처음에는 익숙하지 않으실 테니, 천천히 바꿔 가십시오.

urls.py

```
from django.shortcuts import render
from .models import Cafe

def index(request):
    context = {
        'locations' : Cafe.locations
    }
    return render(request, 'main/index.html', context)

def about(request):
    return render(request, 'main/about.html')
```



선택 지역 카페 리스트 보여주기

index 페이지에서 다음 사진과 같이 지역을 선택해서 카페 리스트를 볼 수 있도록 화면을 구성하려고 합니다.

그러려면 index 페이지에 제주 지역들의 리스트의 정보를 전달해 주어야 합니다.

```
from django.shortcuts import render
from .models import Cafe

def index(request):
    context = {
        'locations' : Cafe.locations
    }
    return render(request, 'main/index.html', context)
```

이렇게 지역 정보를 `locations` 라는 이름으로 index.html에 넘겨주면 index.html의 다음 for 태그를 이용해 지역 이름들이 렌더링 됩니다.

```
{% for location in locations %}
  <div class="jejumap-item" id="jejumap-item-hangyeong">
      <input type="checkbox" class="jejumap-check te-elem" id="check-area-{{ forloop.cou
nter }}" data-city="notJcity" name='locations' value='{{ location.0 }}'>
      <div class="mapshape"></div>
      <label class="jejumap-item-title te-elem" for="check-area-{{ forloop.counter }}">
{{ location.1 }}</label>
  </div>
{% endfor %}
```

이제 index 페이지에서 지역을 선택하면 선택된 지역의 카페들만 보여지도록 cafelist 함수를 다음과 같이 수정합니다.

```
def cafelist(request):
    selected_locations = request.GET.getlist('locations')

    if selected_locations:
        cafes = Cafe.objects.filter(location__in=selected_locations)
    else:
        cafes = Cafe.objects.all()

    context = {
        'cafes': cafes
    }

    return render(request, 'main/cafelist.html', context)
```

- `Cafe.objects.filter(location__in=selected_locations)`
  이 부분의 의미는 `location` 값이 `selected_location`에 포함되는 카페들만 필터링하겠다는 의미입니다.

이제 `제주시`만 선택한 후 카페찾기를 누르면 `제주시` 카페만 나옵니다.

카페찾기!

cafename2

개발자와 디자이너들을 위한 카페에요.

Jeju-si
010-0000-0000
@cafe2

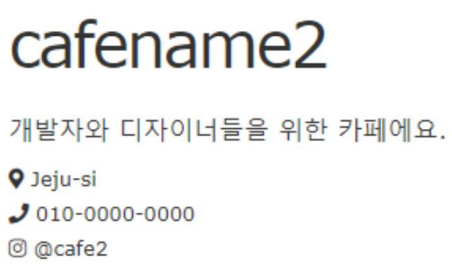
카페 이름 검색기능 구현

메뉴 바의 검색창에 카페 이름을 검색하면 해당 이름과 관련된 카페들의 리스트를 볼 수 있도록 합니다. `cafelist` 함수를 수정합니다.

```
def cafelist(request):

    selected_locations = request.GET.getlist('locations')
    search = request.GET.get('search')

    if selected_locations:
        cafes = Cafe.objects.filter(location__in=selected_locations)
    elif search:
        cafes = Cafe.objects.filter(name__icontains=search)
    else:
        cafes = Cafe.objects.all()

    context = {
        'cafes': cafes
    }

    return render(request, 'main/cafelist.html', context)
```

- `Cafe.objects.filter(name__icontains=search)`
  이 부분의 의미는 카페의 `name`이 검색어 `search`를 포함하고 있는 카페들만 필터링하겠다는 의미입니다.

이제 `2` 를 검색하면, `2`와 연관된 이름을 갖는 `cafename2`만 검색됩니다.



15. 카페 등록 페이지

admin 페이지를 이용해서 카페를 등록할 수 있지만, 이는 관리자만 접속할 수 있으므로 일반 사용자는 사용이 불가능 합니다. 그래서 이번에는 유저가 카페를 등록할 수 있는 페이지를 만들어보겠습니다.

`/write` URL에 `write.html` 페이지가 렌더링 되도록 하겠습니다.

```
from django.contrib import admin
from django.urls import path
from main.views import index, about, write, cafelist, cafedetails
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index, name='index'),
    path('about/', about, name='about'),
    path('write/', write, name='write'),
    path('cafelist/', cafelist, name='cafelist'),
    path('cafelist/<int:pk>/', cafedetails, name='cafedetails'),
]

urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

urls.py

```
from django.shortcuts import render, redirect
from .models import Cafe

(...중략...)

def write(request):
    if request.method == 'POST':

        data = {
            'name': request.POST.get('name'),
            'location': request.POST.get('location'),
            'phone': request.POST.get('phone'),
            'content': request.POST.get('content'),
            'mainphoto': request.FILES.get('mainphoto'),
            'subphoto': request.FILES.get('subphoto'),
        }

        cafe = Cafe.objects.create(**data)

        return redirect(f'/cafelist/{cafe.pk}/')

    context = {
        'locations': Cafe.locations,
    }

    return render(request, 'main/write.html', context)
```

views.py

카페등록

카페위치 ○ 한경면 ○ 한림읍 ○ 애월읍 ○ 제주시 ○ 조천읍 ○ 구좌읍 ○ 대정읍 ○ 안덕면 ○ 서귀포시 ○ 남원읍 ○ 표선면 ○ 성산읍 ○ 우도면

카페이름

전화번호

인스타계정

카페소개

사진1 파일 선택 선택된 파일 없음

사진2 파일 선택 선택된 파일 없음

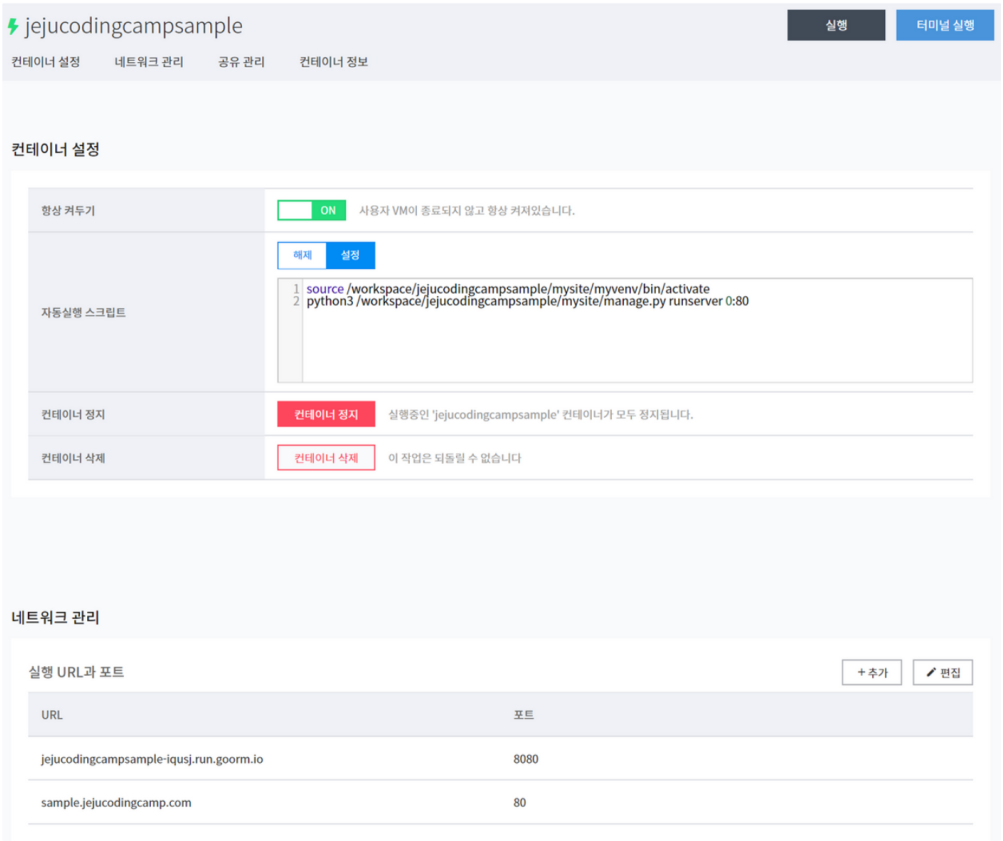
등록하기

서비스 배포

구름 IDE에서는 항상 켜두기 기능을 **프리미엄 계정에서만 제공**되고 있습니다. 클릭만 하시면 항상 켜두기가 활성화 됩니다. 더보기를 클릭하셔서 다음 장으로 넘어가세요.

실행 URL과 포트에서 추가를 누르시고 구매하신 URL을 입력합니다. 자동실행 스크립트는 Django가 실행되기 위한 최소한의 스크립트를 적어놓는 것이 좋습니다.

```
source /workspace/컨테이너이름/mysite/myvenv/bin/activate
python3 /workspace/컨테이너이름/mysite/manage.py runserver 0:80
```



URL이 제대로 연결되고 작동하는지 확인해보세요.

구글 맵 API

구글 맵 API 사용 준비

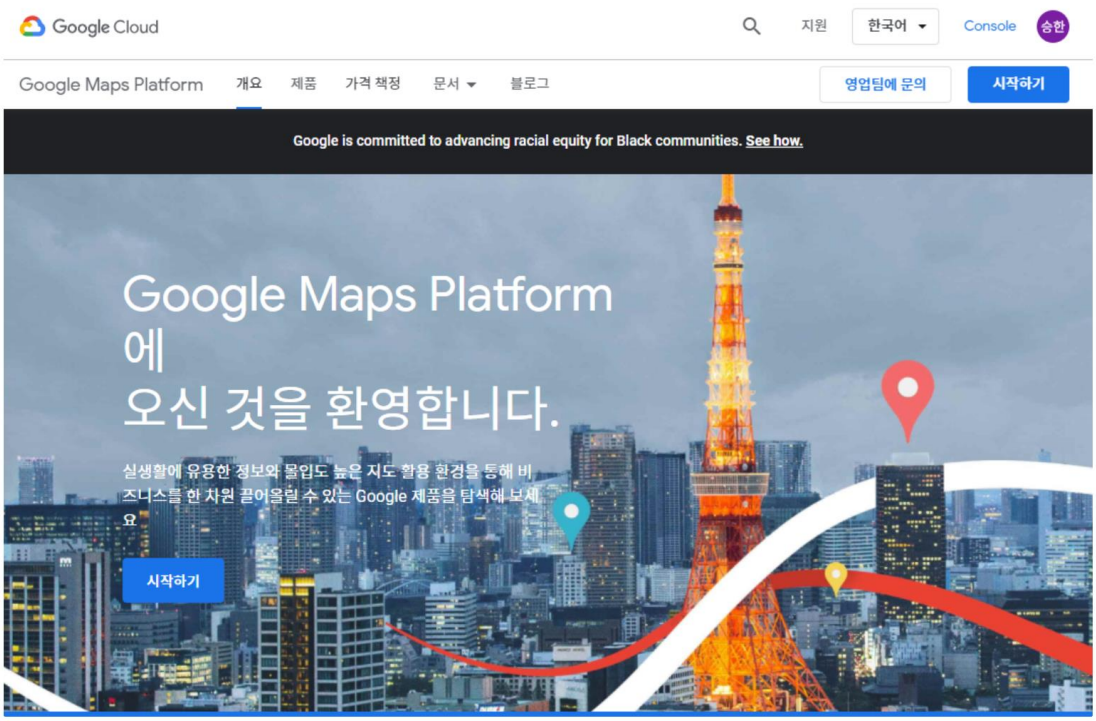
cafedetail에 구글맵 추가하기

구글맵에 마커 추가하기

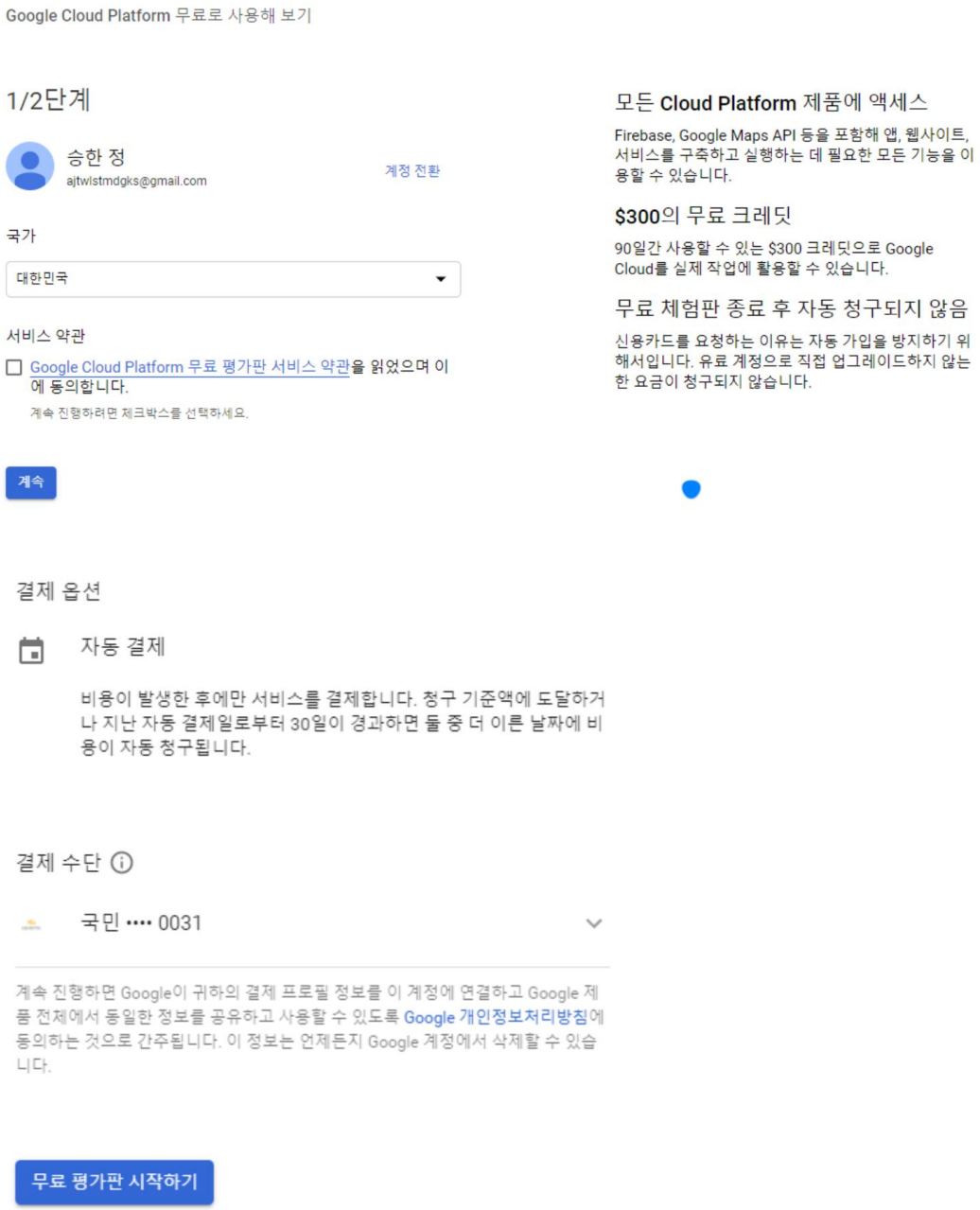
API 키 보안 설정

구글 맵 API 사용 준비

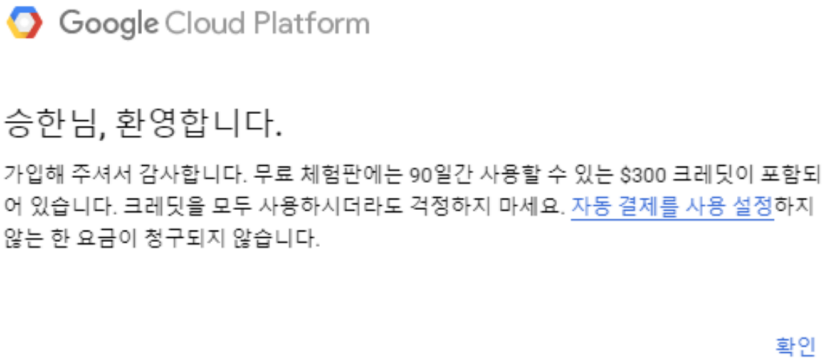
1. 구글에 로그인하고 구글 맵 플랫폼 사이트로 이동합니다.
   https://cloud.google.com/maps-platform/



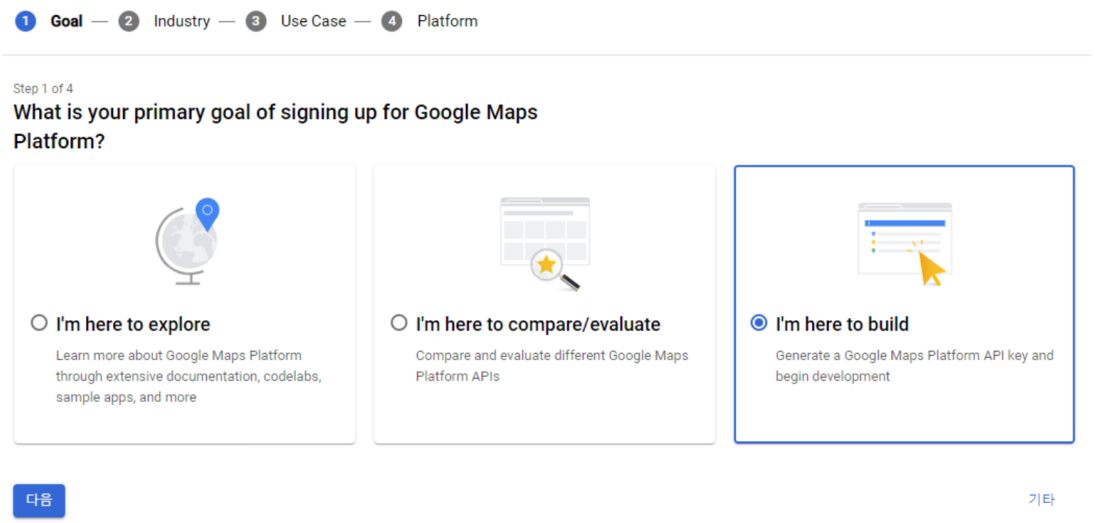
2. 사이트 내의 `시작하기` 버튼을 누르세요. 그러면 GCP(Google Cloude Platform) 무료 평가판 이용을 위한 절차가 진행됩니다.

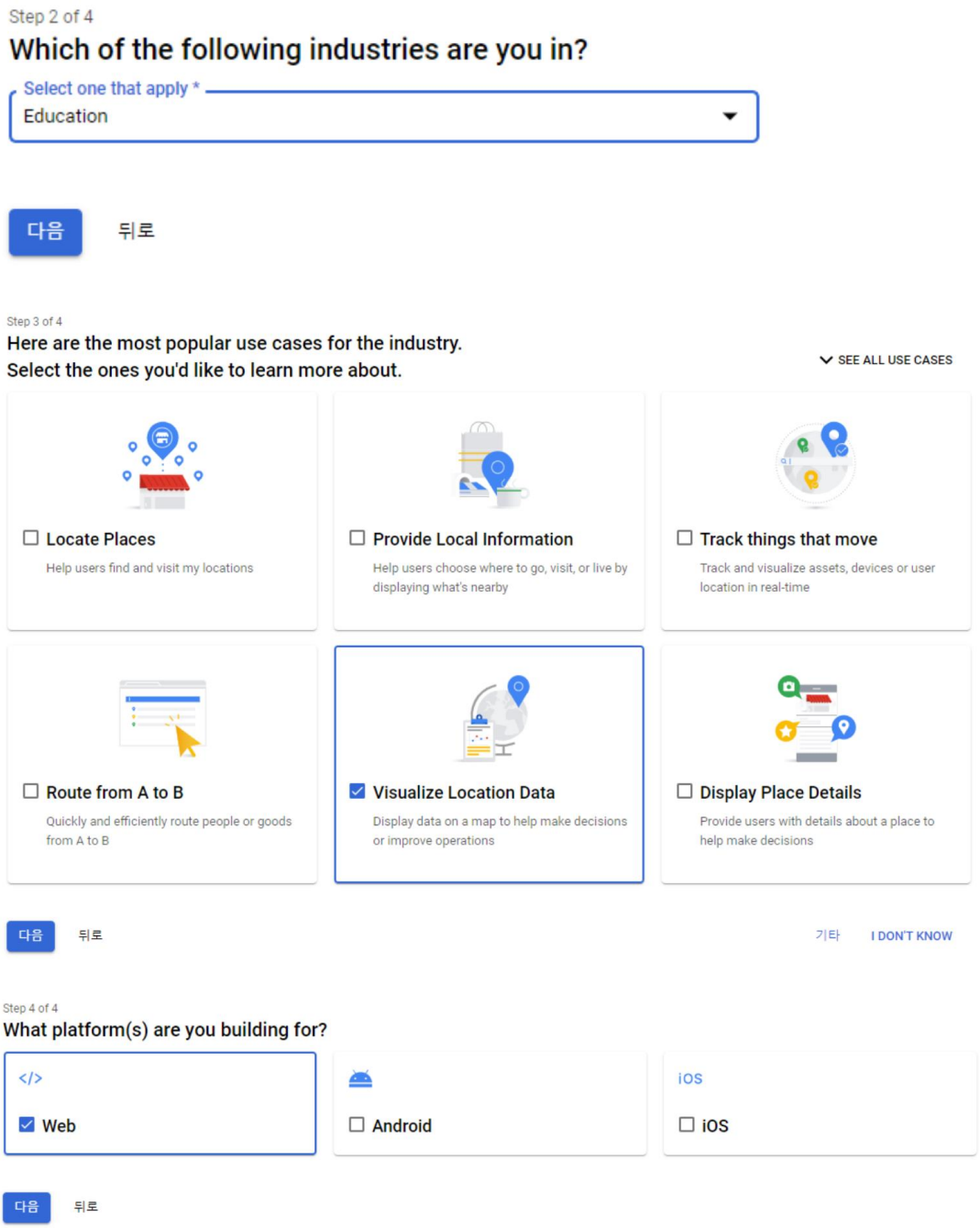


3. (2단계) 결제수단 등록까지 완료하면 90일 동안 Google Cloud Platform에서 제공하는 기능들을 300$ 내에 무료로 이용할 수 있습니다. 구글 안내에 따르면 평가판이 지나도 자동으로 결제되지는 않는다고 합니다.

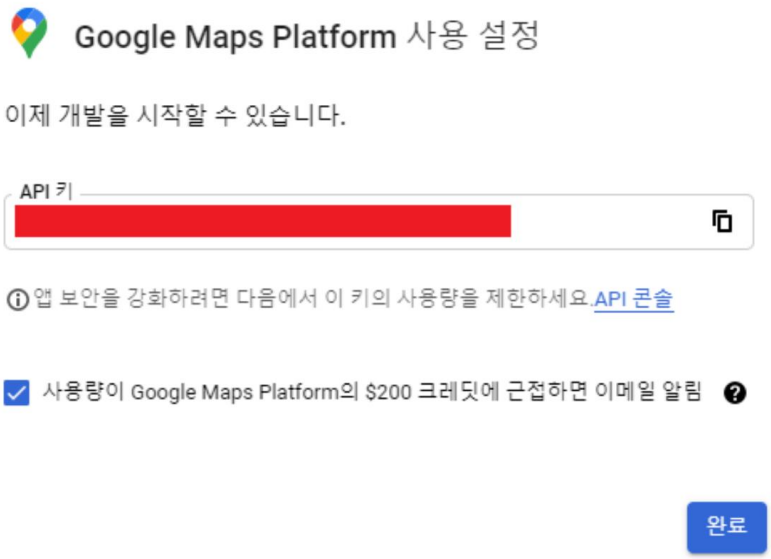


4. 간단한 설문이 진행됩니다. 본인이 사용하고자하는 목적에 맞게 설문하시면 됩니다. 설문 내용에 따라 기능을 추천해준다고 합니다.

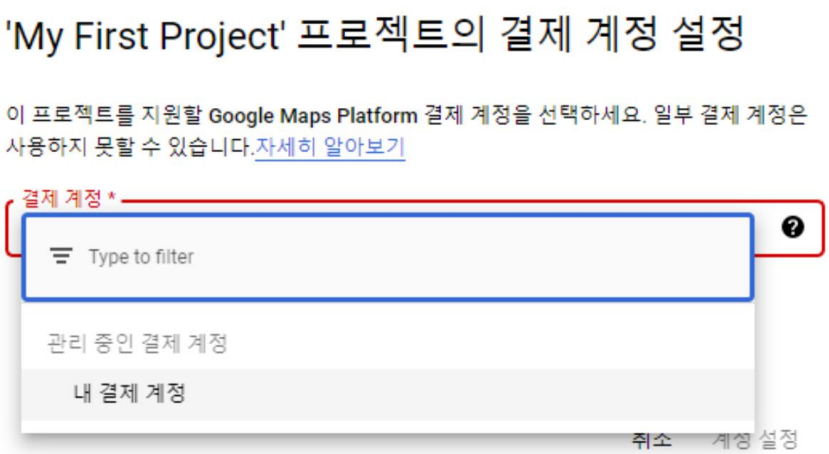



Step 2 of 4
Which of the following industries are you in?
Select one that apply *
Education
다음
뒤로
Step 3 of 4
Here are the most popular use cases for the industry.
Select the ones you'd like to learn more about.
SEE ALL USE CASES
Locate Places
Help users find and visit my locations
Provide Local Information
Help users choose where to go, visit, or live by displaying what's nearby
Track things that move
Track and visualize assets, devices or user location in real-time
Route from A to B
Quickly and efficiently route people or goods from A to B
Visualize Location Data
Display data on a map to help make decisions or improve operations
Display Place Details
Provide users with details about a place to help make decisions
다음
뒤로
기타
I DON'T KNOW
Step 4 of 4
What platform(s) are you building for?
</>
Web
Android
iOS
iOS
다음
뒤로

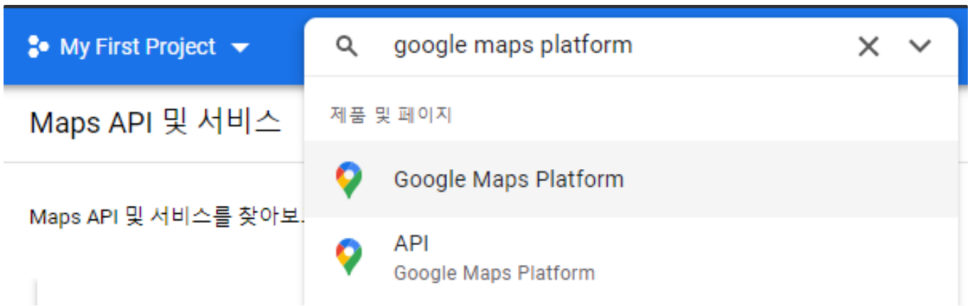
5. 설문이 끝나면 API 키를 받습니다.



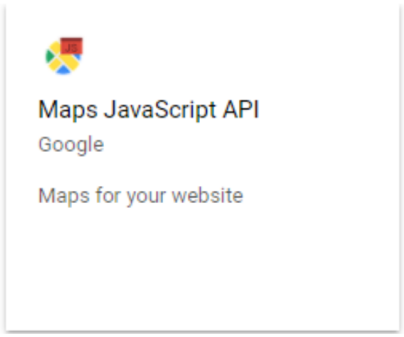
6. `My First Project`라는 이름으로 첫 번째 프로젝트가 생성되었을 겁니다. 해당 프로젝트의 결제 계정을 처음 만들었던 결제 계정으로 설정합니다. 결제 계정을 설정하지 않으면 API를 사용할 수 없습니다.

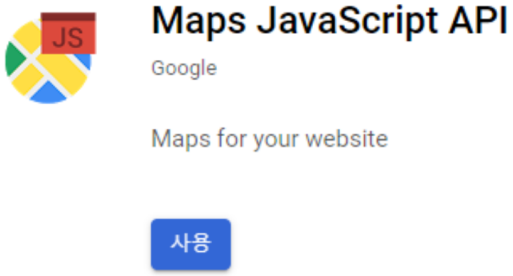


7. `제품 및 리소스 검색`에서 `google maps platform`을 검색합니다.



8. `Maps JavaScript API`를 사용합니다.





cafedetail에 구글맵 추가하기

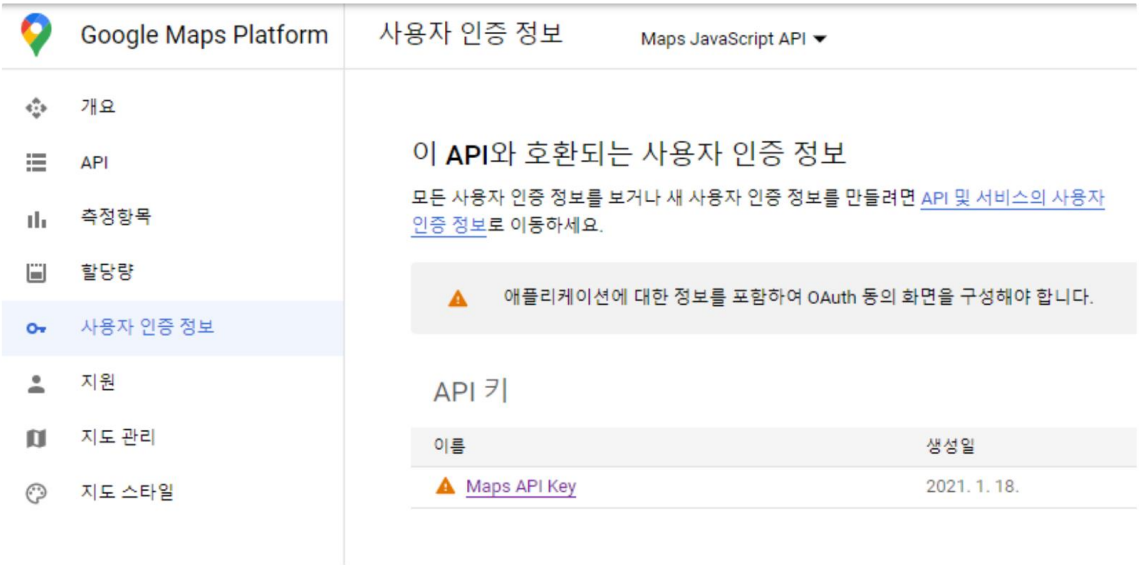
1. 구글 맵을 보여줄 cafedetail.html 에 다음 코드를 추가합니다. `[사용자 키]` 부분에 본인이 발급받은 API 키를 추가합니다.

```
<script>
  function initMap() {
    var jeju = {lat: 33.3616658, lng: 126.5204118};
    var map = new google.maps.Map(document.getElementById('map'), {
      zoom: 10,
      center: jeju
    });

    var marker = new google.maps.Marker({
      position: {lat: {{cafe.lat}}, lng: {{cafe.lng}}},
      map: map
    });

  }
</script>
```

`<body>` 태그 마지막 부분에 다음 코드를 추가합니다.

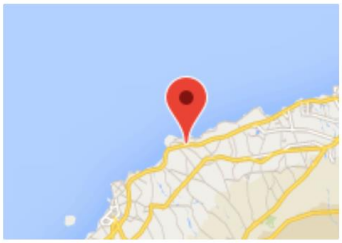
```
<script async defer
    src="https://maps.googleapis.com/maps/api/js?key=[사용자 키]&callback=initMap">
</script>
```

2. 해당 페이지로 가면 이제 구글맵을 볼 수 있습니다.



구글맵에 마커 추가하기

구글 맵에는 개발자가 원하는 위치를 가리키는 마커를 삽입할 수 있습니다. 해당 카페의 위치를 마커로 표시해보겠습니다.



마커를 삽입하기 위해서는 해당 위치의 위도와 경도를 알아야 합니다. Cafe 모델에 카페의 위도와 경도를 저장하는 `lat`, `lng` 필드를 정의합니다.

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)

    locations = [
        ('Hangyeong-myeon', '한경면'),
        ('Hallim-eup', '한림읍'),
        ('Aewol-eup', '애월읍'),
        ('Jeju-si', '제주시'),
        ('Jocheon-eup', '조천읍'),
        ('Gujwa-eup', '구좌읍'),
        ('Daejeong-eup', '대정읍'),
        ('Andeok-myeon', '안덕면'),
        ('Seogwipo-si', '서귀포시'),
        ('Namwon-eup', '남원읍'),
        ('Pyoseon-myeon', '표선면'),
        ('Seongsan-eup', '성산읍'),
        ('Udo-myeon', '우도면'),
    ]
    location = models.CharField(max_length=50, choices=locations)
    lat = models.FloatField(null=True)
    lng = models.FloatField(null=True)
    mainphoto = models.ImageField(blank=True, null=True)
    subphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()
    phone = models.CharField(max_length=20, null=True)
    insta = models.CharField(max_length=20, null=True)

    def __str__(self):
        return self.name
```

`models.py` 파일을 수정하면 `makemigrtions`와 `migrate`를 해야 한다는 사실 잊지 않으셨죠?

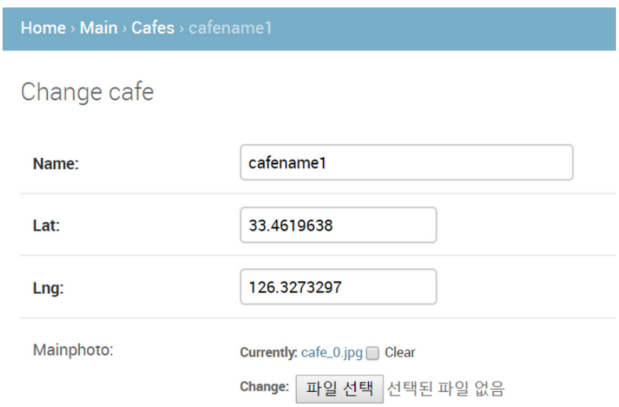
```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py makemigrations
Migrations for 'main':
  main/migrations/0005_auto_20200720_0806.py
    - Add field lat to cafe
    - Add field lng to cafe

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
...
Running migrations:
  Applying_main.0005_auto_20200720_0806... OK
```

샘플로 구글맵에서 각 지역을 대표하는 장소를 검색하여 좌표를 알아낸 다음 아래와 같이 입력합니다.

1번 카페- (33.4619638, 126.3273297) - 애월읍사무소

2번 카페 - (33.6628574, 126.0112614)- 제주시청

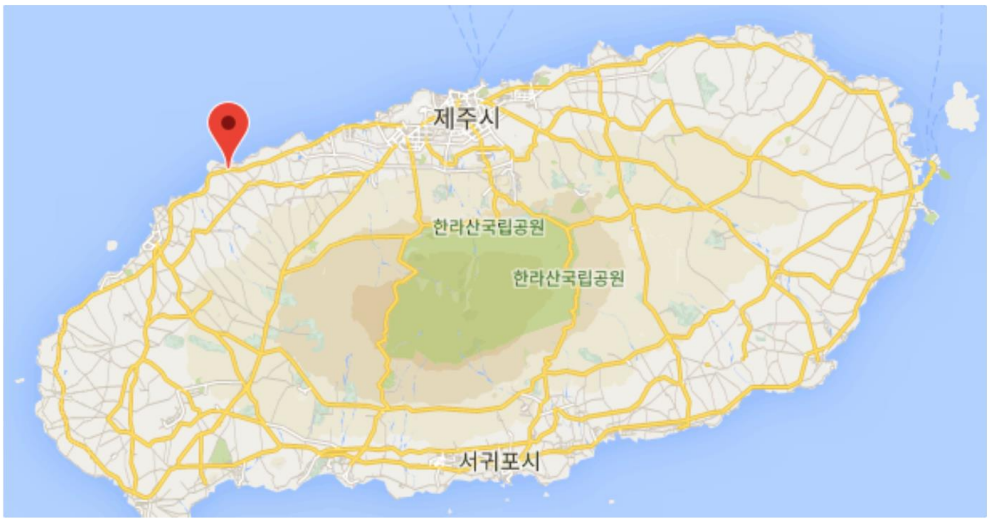
3번 카페 - (33.5380993, 126.631991) – 조천읍사무소



이제 구글 맵 API 코드에 `marker`를 추가합니다.

```
<script>
  function initMap() {
    var jeju = {lat: 33.3616658, lng: 126.5204118};
    var map = new google.maps.Map(document.getElementById('map'), {
      zoom: 10,
      center: jeju
    });

    var marker = new google.maps.Marker({
      position: {lat: {{cafe.lat}}, lng: {{cafe.lng}}},
      map: map
    });

  }
</script>
```

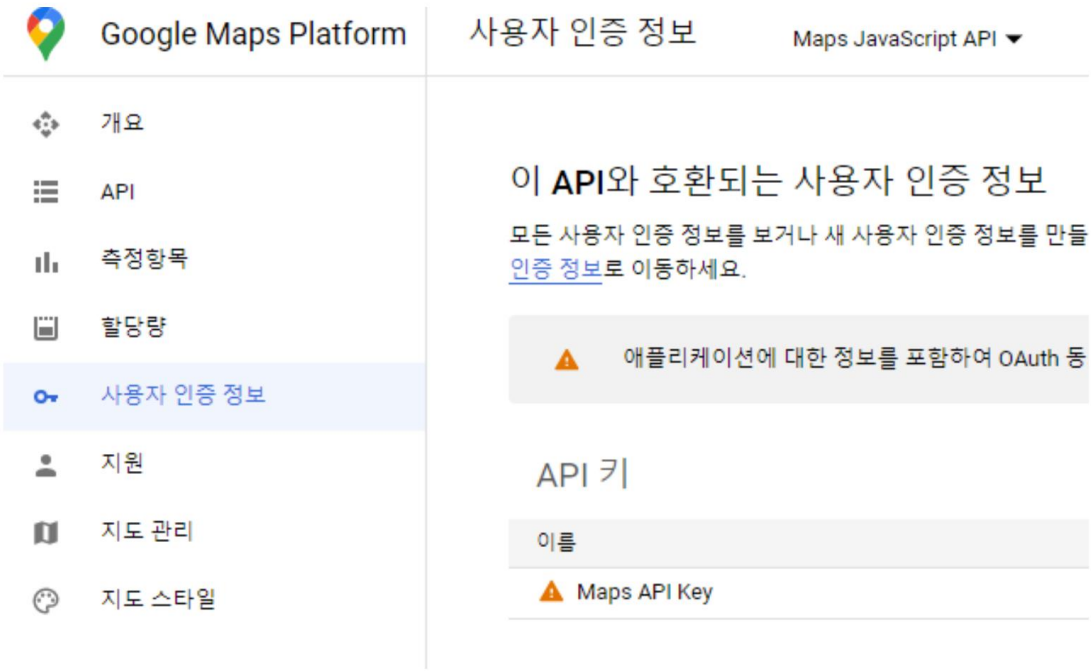
그러면 이제 지도에 해당 위도와 경도에 마커가 표시됩니다.



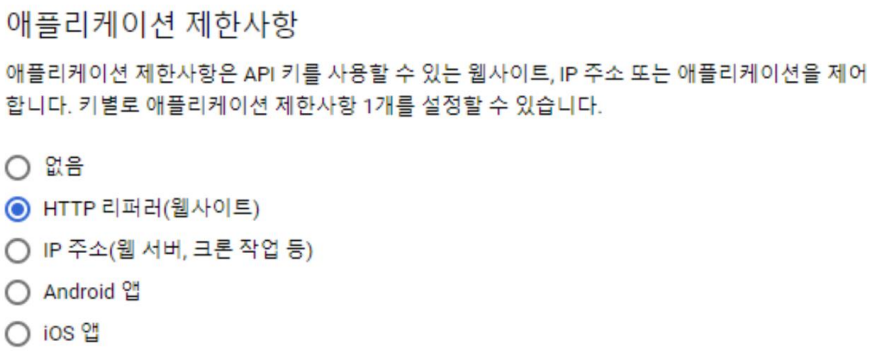
API 키 보안 설정

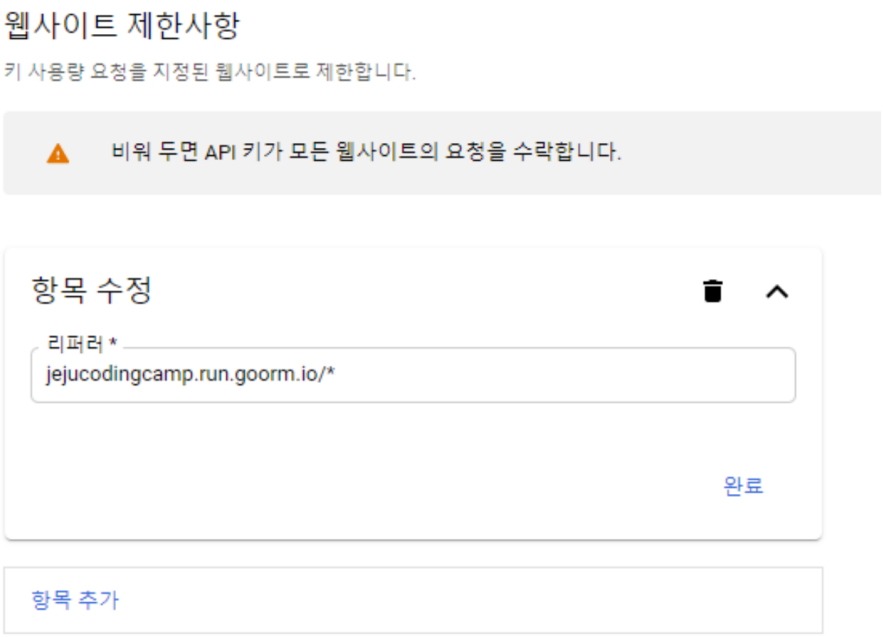
API 키는 노출 될 시 다른 이용자에게 악용될 수 있기 때문에 노출되더라도 안전하도록 사전에 막아 둘 필요가 있습니다.

1. Google Maps Platform에서 `사용자 인증 정보`로 들어가 발급 받은 키를 클릭하여 설정으로 들어갑니다.

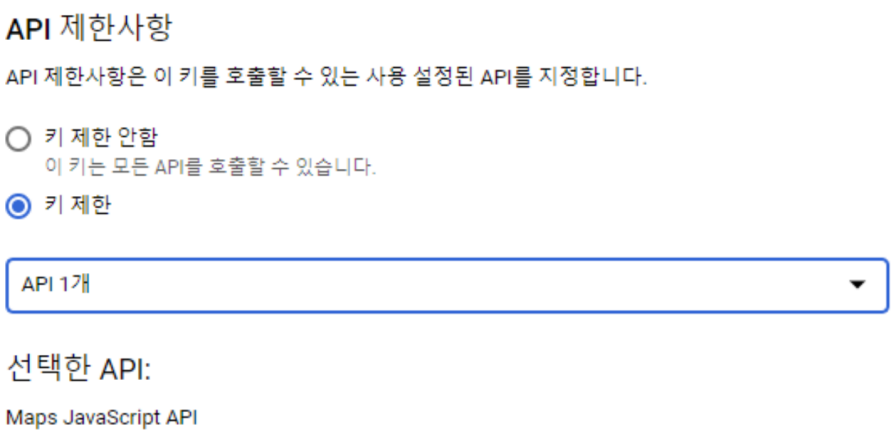


2. HTTP 리퍼러 설정을 통해 본인이 원하는 사이트, 즉 본인이 개발한 사이트에서만 사용할 수 있도록 해당 도메인을 리퍼러로 등록합니다.

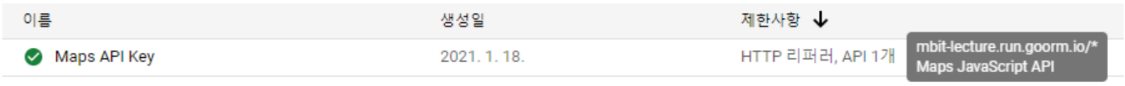




3. `API 제한사항`에서 해당 키가 사용되길 원하는 API를 제한합니다. 우리는 `Maps JavaScript API`만 사용할 것이기 때문에 이것만 허용하도록 설정합니다.



4. 설정을 저장하면 제한사항이 잘 추가된 것을 목록에서 확인할 수 있습니다.



이제 다른 사이트에서 API를 사용할 시 해당 사이트는 허용되지 않았다는 의미의 에러 메시지가 노출 됩니다. (실제 반영은 최대 5분 소요될 수 있습니다.)

google map에 대하여 좀 더 알고 싶으시다면 아래 튜토리얼을 참고하세요.

Google API Tutorial

Well organized and easy to understand Web building tutorials with lots of examples of how to use HTML, CSS, JavaScript, SQL, PHP, Python, Bootstrap, Java

https://www.w3schools.com/graphics/google_maps_intro.asp

18. 번외 튜토리얼

1. Django Shell(번외 튜토리얼)

2. Django form(번외 튜토리얼)

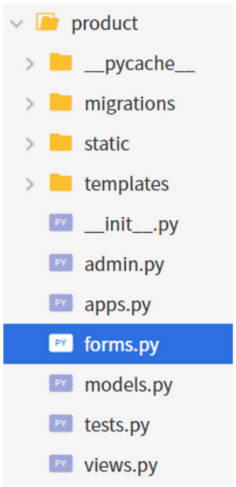
1. Django Shell(번외 튜토리얼)

Python 기초를 배웠던 colab이나 jupyter notebook을 사용해보셨다면! 그곳에서 UserData를 분석해 보거나, Django에 여러 모듈을 만져보고 싶지 않으신가요? 그래서 준비했습니다.

Django Shell과 Jupyter notebook에서 Django를 import하여 사용하는 방법입니다.

2. Django form(번외 튜토리얼)

사용자가 입력하게 하는 form은 어떻게 작성할 수 있을까요? 이 튜토리얼과는 번외로 작성해보았습니다. 필요하신 분이 있으시다면 아래 튜토리얼을 따라해보세요.



App 안에 form.py파일을 만들고 그 안에 아래와 같이 입력합니다.

```
from django import forms
from .models import Product

class PostForm(forms.ModelForm):
    class Meta:
        model= Product
        fields = ('productname', 'contents' ,)
```

form으로 전달된 값이 POST인지 확인하여 form을 처리하는 코드입니다.

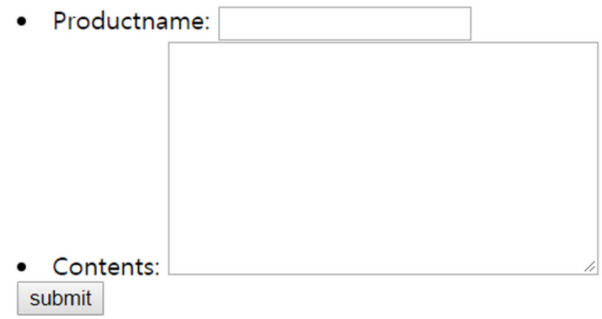
```
from django.shortcuts import render
from .models import Product
from .forms import PostForm

#Create your views here
def index(request):
    return render(request, 'product/index.html')

def product(request):
    productlist = Product.objets.all()
    if request.method == "POST":
        form = PostForm(request.POST)
        if form.is_valid():
            post = form.save(commit=False)
            post.author = request.user
            post.save()
            return render(request, 'product/product.html, {'productlist':productlist})
        else:
            form = PostForm()
        return render(request, 'product/product.html, {'form':form, 'productlist':produc
tlist}
```

html 파일은 아래와 같이 작성해주세요.

```
<form method="POST">
    {% crsf_token %}
    {{ form.as_p}}
    <button type="submit">submit</button>
</form>
```

form.as_p 탬플릿 태그를 사용하면 Django가 모델에 있는 form을 템플릿으로 전달해줍니다. 아래 3개의 형태를 추가하니, 참고바랍니다.

form.as_p



form.as_ul

form.as_table

요약정리

1. Django 공부 방향과 원리

2. URL 설계

3. VIEW 로직 설계

4. MODEL 데이터베이스 설계

5. Templates

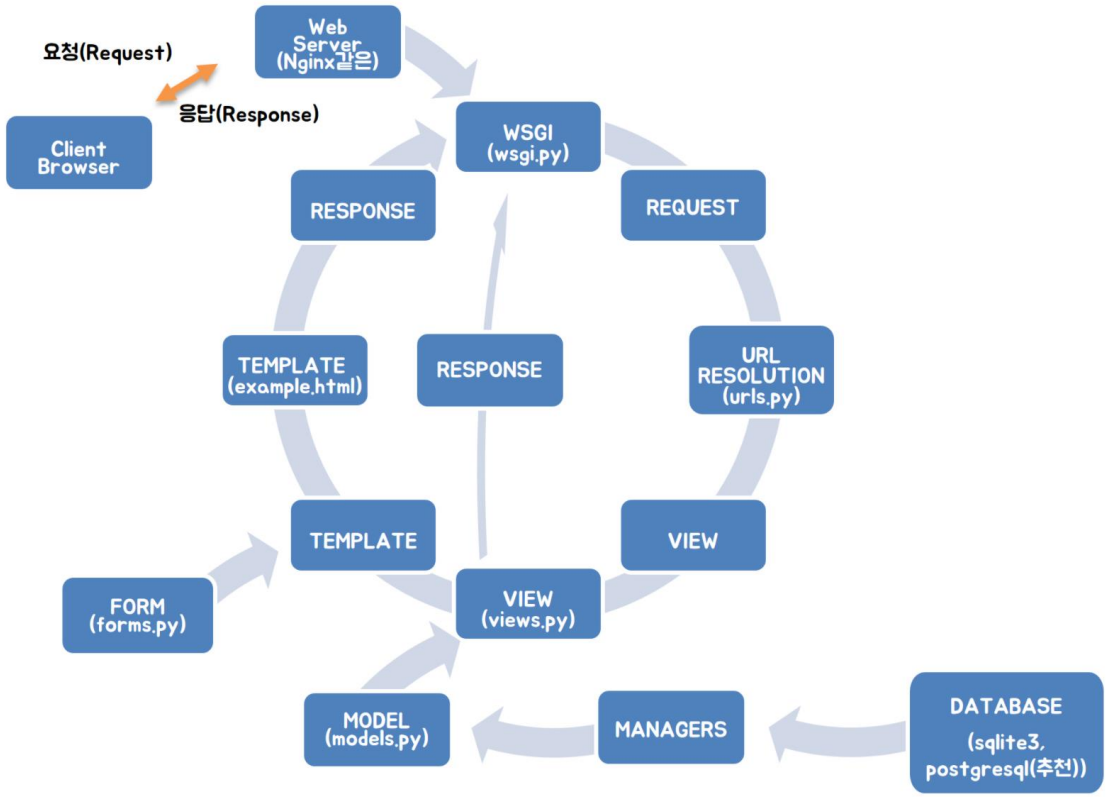
1. Django 공부 방향과 원리

Django는 Python으로 빠르게 웹을 개발하기 위해 나온 Web Framework입니다. 대부분의 기능들이 자동화 되어 있으며 공식문서도 잘 되어 있어 공식문서 정독을 한번 하면 다른 문서는 필요 없을 정도로 상세 내용을 가지고 있습니다.

우리는 지금까지 간단한 카페 블로그를 만들어 보았습니다. 그러나 Django에 극히 일부분만을 다룬 것입니다. 더 강력한 Django의 기능을 사용하고 싶으시다면 **AskDjango의 강의를 참고하시길 권해드립니다.** 또한, 책은 아래 3권을 추천해드려요.

1. Python 웹 프로그래밍(초급, 공식문서에 가까운 상세한 설명)
2. Django로 쉽게 배우는 배프의 오지랖 파이썬 웹 프로그래밍(실전 응용)
3. AWS 클라우드 기반의 Django 웹 어플리케이션(실전 응용)

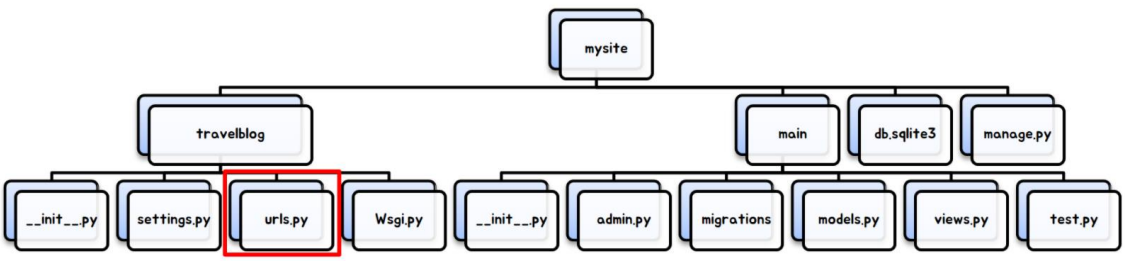
아래는 Django의 기능을 전체 도식화 정리 해놓은 것입니다. 보시면서 놓치신 것이 없으신지 체크해 보시기 바랍니다.



2. URL 설계

> 전체 소스코드는 https://github.com/paullabkorea/jejucodingcamp 에서 보실 수 있습니다. 프로젝트 실행을 해보고 싶으시다면 해당 사이트의 README 설명서를 따라 소스코드를 다운로드 받고 세팅해주세요.

챕터에서 수정한 순서대로 빨간색 네모를 그려보았습니다. 처음에는 urls.py에서 url 파싱을 하였습니다. 아래 코드는 우리가 수정한 코드 전문입니다.



urls.py

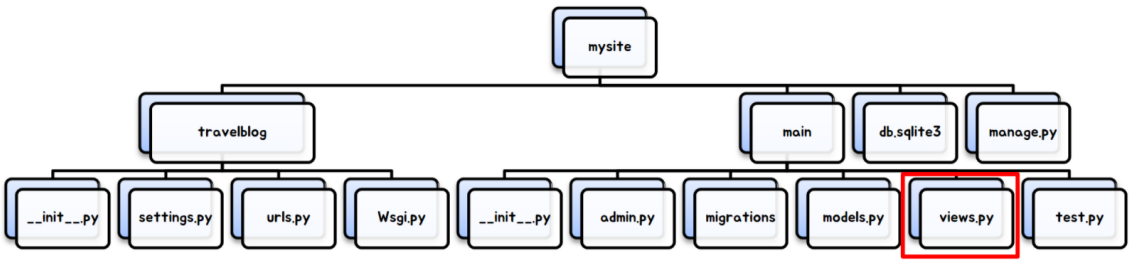
```
from django.contrib import admin
from django.urls import path
from main.views import index, about, write, cafelist, cafedetails
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index, name='index'),
    path('about/', about, name='about'),
    path('write/', write, name='write'),
    path('cafelist/', cafelist, name='cafelist'),
    path('cafelist/<int:pk>/', cafedetails, name='cafedetails'),
]

urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

settings.py

```
# (생략)

INSTALLED_APPS = [
    'main',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]

# (생략)


STATIC_URL = '/static/'

STATIC_ROOT = BASE_DIR / 'staticfiles'

MEDIA_ROOT = BASE_DIR / 'media'

MEDIA_URL = '/media/'
```

3. VIEW 로직 설계

URL 설계가 끝났다면 URL에서 연결해준 각종 함수를 views에서 만들어 줍니다.



views.py

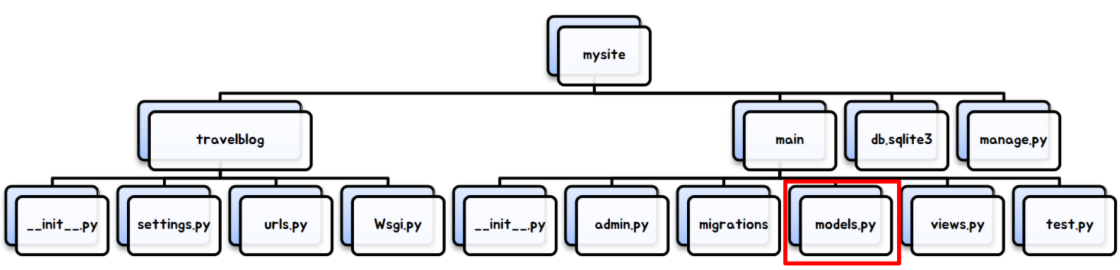
```
from django.shortcuts import render, redirect
from .models import Cafe


def index(request):
    context = {
        'locations' : Cafe.locations
    }
    return render(request, 'main/index.html', context)


def about(request):
    return render(request, 'main/about.html')


def cafelist(request):

    selected_locations = request.GET.getlist('locations')
    search = request.GET.get('search')

    if selected_locations:
        cafes = Cafe.objects.filter(location__in=selected_locations)
    elif search:
        cafes = Cafe.objects.filter(name__icontains=search)
    else:
        cafes = Cafe.objects.all()

    context = {
        'cafes': cafes
    }

    return render(request, 'main/cafelist.html', context)
```

```
def cafedetails(request, pk):
    cafe = Cafe.objects.get(pk=pk)

    context = {
        'cafe': cafe,
    }

    return render(request, 'main/cafedetails.html', context)


def write(request):
    if request.method == 'POST':

        data = {
            'name': request.POST.get('name'),
            'location': request.POST.get('location'),
            'phone': request.POST.get('phone'),
            'insta': request.POST.get('insta'),
            'content': request.POST.get('content'),
            'mainphoto': request.FILES.get('mainphoto'),
            'subphoto': request.FILES.get('subphoto'),
        }

        cafe = Cafe.objects.create(**data)

        return redirect(f'/cafelist/{cafe.pk}/')

    context = {
        'locations': Cafe.locations,
    }

    return render(request, 'main/write.html', context)
```

4. MODEL 데이터베이스 설계

아래 코드는 후에 작성하였지만 설계를 제대로 하셨다면 세 번째에서 작업할 사항입니다. 각종 게시물 등 필요한 model을 설계합니다.

전부 작성한 후에는 makemigrations와 migrate로 DB에 반영합니다.



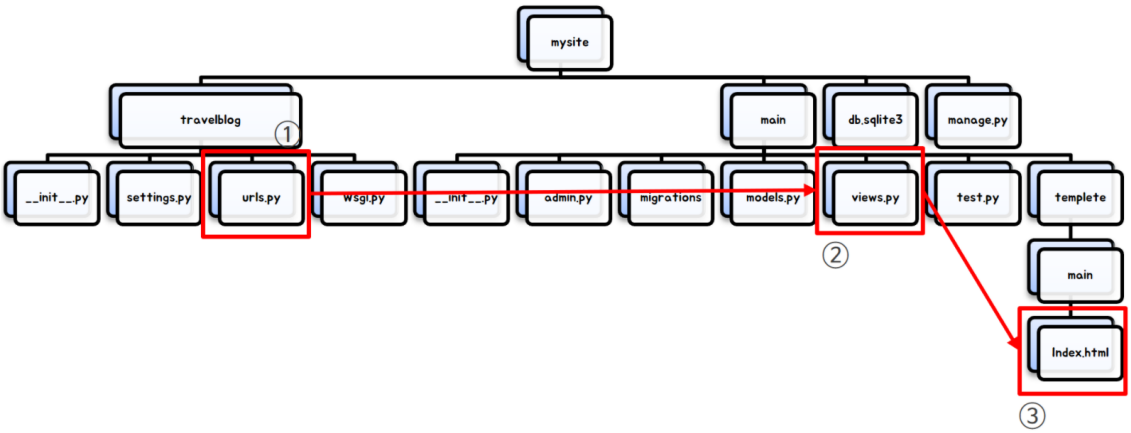
models.py

```
from django.db import models


class Cafe(models.Model):
    name = models.CharField(max_length=50)

    locations = [
        ('Hangyeong-myeon', '한경면'),
        ('Hallim-eup', '한림읍'),
        ('Aewol-eup', '애월읍'),
        ('Jeju-si', '제주시'),
        ('Jocheon-eup', '조천읍'),
        ('Gujwa-eup', '구좌읍'),
        ('Daejeong-eup', '대정읍'),
        ('Andeok-myeon', '안덕면'),
        ('Seogwipo-si', '서귀포시'),
        ('Namwon-eup', '남원읍'),
        ('Pyoseon-myeon', '표선면'),
        ('Seongsan-eup', '성산읍'),
        ('Udo-myeon', '우도면'),
    ]

    location = models.CharField(max_length=50, choices=locations)
    lat = models.FloatField(null=True)
    lng = models.FloatField(null=True)
    mainphoto = models.ImageField(blank=True, null=True)
    subphoto = models.ImageField(blank=True, null=True)
    published_date = models.DateTimeField(auto_now_add=True)
    modified_date = models.DateTimeField(auto_now=True)
    content = models.TextField()
    phone = models.CharField(max_length=20, null=True)
    insta = models.CharField(max_length=20, null=True)

    def __str__(self):
        return self.name
```

```
from django.contrib import admin
from .models import Cafe

admin.site.register(Cafe)
```

5. Templates

이번에는 Template을 작성 해보겠습니다. 중간에 모델 설계를 하지 않았다면 3번째 작업할 사항입니다. 우리는 templates라는 폴더 아래 main이라는 폴더를 만들어서 그 안에 각종 템플릿을 작성하였습니다.



초판 1쇄 발행 | 2020년 7월 27일
2쇄 발행 | 2021년 1월 27일
3쇄 발행 | 2021년 3월 12일
지은이 | 이호준 유준모 이범재 김혜원 김유진 차경림 김난화 김승민 김진 이숭신 이진우 현지연 정승한 강민정 정윤하
편 집 | 이호준 차경림 현지연
총 괄 | 이호준
펴낸곳 | 사도출판
주 소 | 제주특별자치도 제주시 동광로 137 대동빌딩 4층
표지디자인 | 차경림
홈페이지 | http://www.paullab.co.kr
E - mail | paul-lab@naver.com
ISBN | 979-11-88786-50-3

2021 JEJU 코딩 베이스캠프 시즌 7

weniv

비매품/무료
95560
9 791188 786503
ISBN 979-11-88786-50-3 (PDF)