



TUTORIAL PROJECT

여행블로그 튜토리얼로 배우는 Django 2.0 with 구름IDE and Bootstrap 4.1.0

이호준, 최수원



지은이

지은이 이호준

(現) 바울랩아이씨티기술연구원 대표이사

(現) 바울랩아이씨티컴퓨터학원 대표이사

(現) 사도출판 대표이사

제주 코딩 베이스캠프 운영(<http://www.jejucodingcamp.com>)

제주 코딩 해커톤 운영

제주 코딩 Festival 운영

제주로 온코딩 고등동아리 지원사업 진행

제주도 코딩 거버넌스 협의체(민, 관, 학) (<http://jejucoding.com/v2/>)

제주 블록체인 협의체

지은이 최수원

(주) 투게더스 웹 및 모바일 개발자

<https://suwoni-codelab.com/> 블로그 운영

지은이의 말

교단에서 강의할 때 한 학생이 찾아왔습니다. 가정 사정이 어려운데 SW를 배울 수 있는 학원은 가격이 너무 비싸고 그렇다고 혼자 공부할 수 있는 공부는 아니라 는 막막함에 찾아온 학생이었습니다.

소프트웨어는 이제 삶에 한 부분이고 무엇을 꿈꾸던지 소프트웨어를 배워야 한다면 그 진입장벽 또한 낮아져야 합니다. 물론 SW 전문 교육 기관을 운영하는 입장에서 어려운 과제임이 확실합니다. 그러나 어른들의 숙제가 이제 막 꿈꾸기 시작한 학생들에게 전가되어서는 안됩니다.

또한 쉽게, 다양하게 배울 수 있어야 합니다. 학생에게 두꺼운 책과 끝내지 못한 책은 그 자체로 진입장벽이 됩니다. 튜토리얼 시리즈는 여러분이 끝낸 한 권의 책이 되어 더 깊은 학문으로 가기 위한 발판이 되길 원합니다.

가정 형편이 어려워도 최고의 교육을 누구나 누릴 수 있도록, 이미 주어진 환경이 아닌 노력과 열정과 꿈의 크기만큼 성장할 수 있도록, 배움의 기쁨과 문제 해결의 쾌감을 누릴 수 있도록. 우리는 노력하겠습니다.

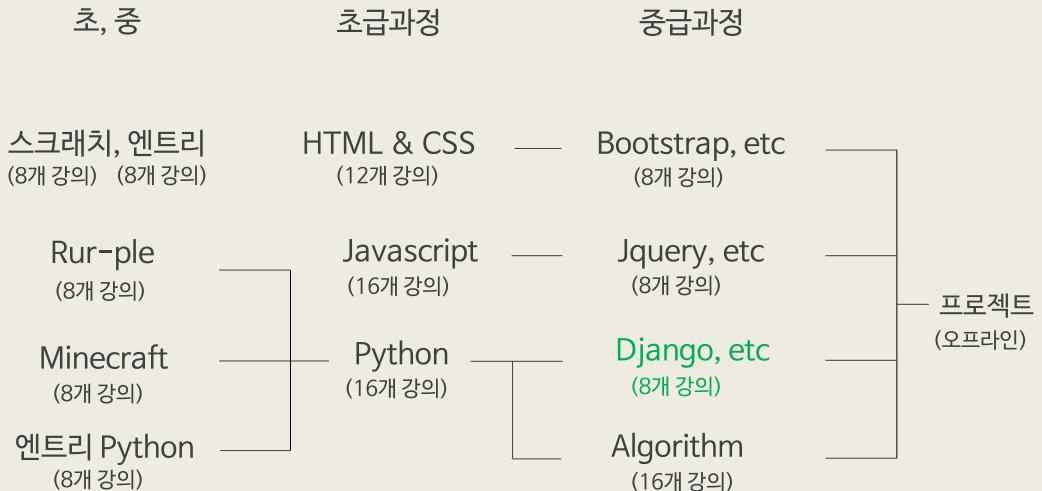
강의 사이트에 올라와 있는 1. Road Map으로 목표한 곳으로 가는 길을 그려보고 2. 강의 로드맵 동영상을 시청하여 무엇을 배워야 할지 파악한 다음 3. 튜토리얼 프로젝트를 단계별로 공부하십시오. 이 과정을 관통하여 SW의 모호함을 조금이나마 걷어내고 꿈을 구체화 시킬 수 있기를 바랍니다.

이호준

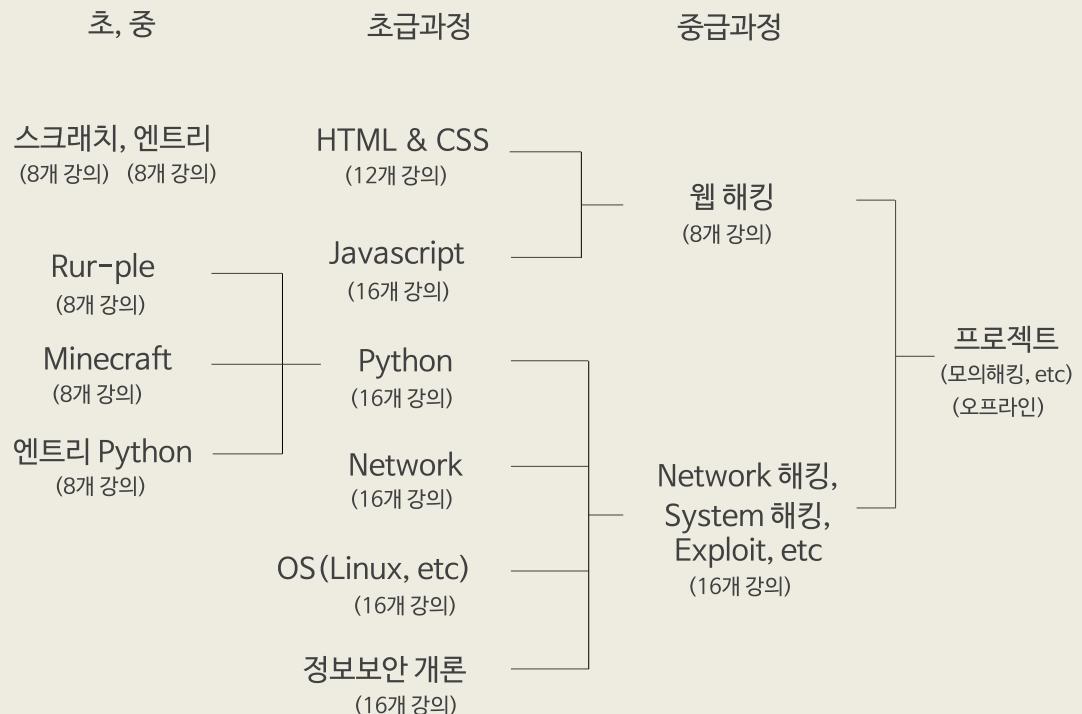
강의 Road Map

<http://paullab.co.kr> > Service > Book service

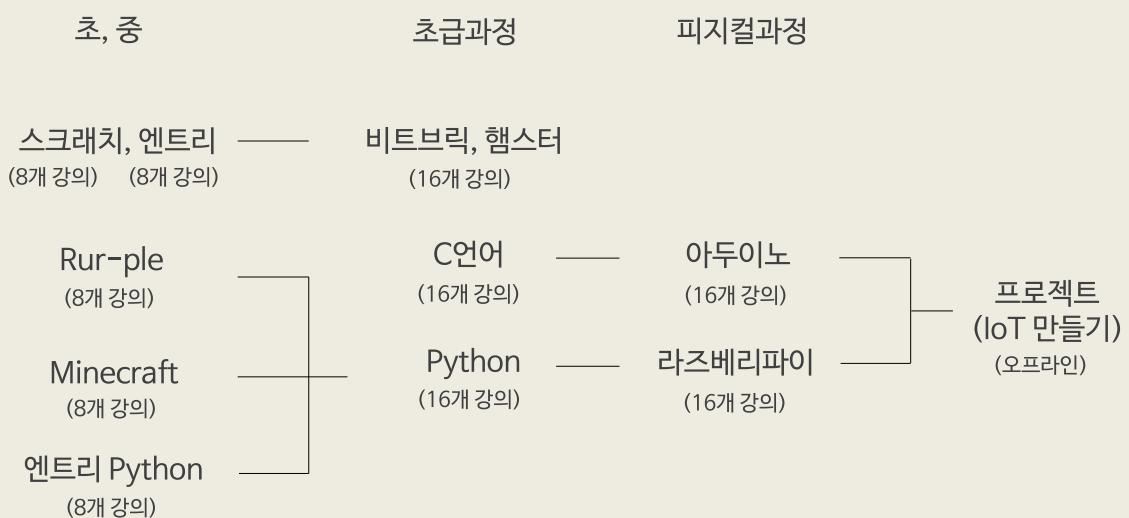
1. 웹 개발 (Front-end & Back-end : Full-Stack)



2. 보안



3. Physical Computing



4. 고급과정

- 4.1. 파이썬을 활용한 빅데이터 분석(8개 강의)
- 4.2. 파이썬 자율주행 로봇 만들기 (8개 강의)
- 4.3. 파이썬으로 배우는 정보보안 (8개 강의)
- 4.4. PyQt를 활용한 파이썬 GUI 프로그래밍 (8개 강의)
- 4.5 투토리얼로 배우는 TensorFlow (8개 강의)
- 4.6 투토리얼로 배우는 Django (8개 강의)
- 4.5 안드로이드 앱 개발 (22개 강의, java 강의 포함)
- 4.6 Front-end 심화 과정 (16개 강의)
- 4.7 Back-end 심화 과정 (16개 강의)



대상 독자

이 책은 Django로 무언가를 만들어보고 싶으신 분들에게 추천해드립니다. Django는 Python Web Framework 중 가장 사랑받는 Framework입니다. Django로 간단한 블로그를 만들고 배포해본 경험으로 자신이 원하는 사이트를 구축하는 발판이 되었으면 합니다.

이 책의 내용

이 책은 Django로 간단한 여행블로그를 만들면서 Django에 대해 익히게 되어 있는 튜토리얼 책입니다. 따라서 Django에 대해 심도 깊게 다루기 보다는 서비스를 만들기 위한 일련의 과정을 빠르게 한 사이클 도는 것에 중점을 두었습니다. 따라서만 하면 서비스를 만들 수 있도록 하였으며 각각에 코드에 대해 파일로 만들어 두었으니 참고하셔서 작업하시면 좋습니다.

이 책의 프론트엔드 부분은 ‘여행블로그 튜토리얼로 배우는 Bootstrap’편에 실려 있습니다.

* 모든 templates 소스는 paullab.co.kr/templates.zip에 있습니다.



무엇을 어떻게 배워야 할까요?

Python을 배우셨고 Django를 간단하게 실습해 보셨다면 이제는 실 서비스를 만들어 보세요. 오류를 만나면 해결하면서 몰랐던 부분들을 체킹하시고 해결한 것들을 문서화 해놓으세요. 반드시 한번 더 되돌아 보게 되어 있습니다.

Python 자체에 관심이 있으시다면 데이터 분석, 시각화, 인공지능 등에 대해 병행 공부하시면 좋습니다. 그 모든 것을 웹 서비스로 풀어 내실 수 있으시다면 좋은 공부가 될 것입니다.



목차

Step 1. 환경소개 및 설정 9

1.1 Django설치 및 세팅 환경소개	-----	10
1.2 환경설정	-----	14

Step 2. Django로 Main page 만들기 31

Step 3. Django로 Blog page 만들기 46

Step 4. Django로 Post page 만들기 59

Step 5. 댓글과 태그 기능 만들기 76

Step 6. Bootstrap 입히기 및 서비스 배포 93

Step 7. 요약정리 120

Step_1

구름 IDE 환경설정

Tutorial 1.1

Django 설치 및 세팅 환경소개

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 종합선물 세트라고 생각하시면 편합니다. 보다 빠르고 편리하게 웹 서비스를 개발할 수 있도록 필요한 공구들을 모은 것이죠.

Django는 2005년 오픈소스로 시작되어 2017년 12월 Django 2.0이 나왔으며 아직 한국에는 2.0 관련된 책이 없습니다.

책에는 상세한 내용을 다루기보다 전체적인 맥락을 잡기 위해 한 사이클을 도는 것에 초점이 맞춰져 있으니 보다 자세한 내용은 공식 홈페이지에 문서를 참고하시면 좋을 것 같아요.

<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.

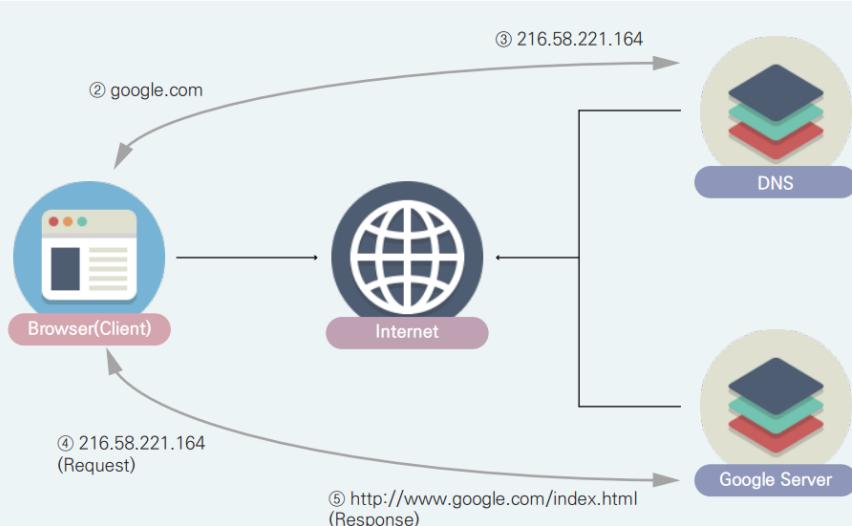
* 1870 page에 방대한 양을 pdf로도 제공합니다.

- <https://media.readthedocs.org/pdf/django/2.0.x/django.pdf>

1.1 Django설치 및 세팅 환경소개 (계속)

인터넷이 어떻게 구성되어 있는지에 대한 설명입니다. 아래 내용은 google에 접속 했을 때에 일어나는 일련의 과정들을 정리한 것입니다. 자세한 설명은 동영상으로 제공합니다. 지금은 이해하지 않고 넘어가셨다가 책을 빠르게 훑어보고 다시 돌아오셔도 좋습니다.

우리는 여기서 google Server에 해당하는 부분을 구성하게 됩니다. 사용하는 언어는 HTML, CSS, Javascript, Python이며 사용하는 프레임 워크로 Bootstrap, Django를 사용합니다.



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.



이미지 - 튜토리얼로 배우는 HTML & CSS

1.1 Django설치 및 세팅 환경소개 (계속)

우리는 구름IDE 클라우드 서비스를 사용할 것입니다. 리눅스 기반이며 우분투 14버전을 사용합니다. 대부분의 분들이 마우스를 움직이는 그래픽 기반의 GUI환경에 익숙해 있으셔서 검은 화면에 명령어만 치는 CLI환경이 낯설 것이라 생각됩니다. 그렇다 하더라도 전세계 서버의 90% 이상의 리눅스기반 환경이기 때문에 익히시는 것을 추천해 드립니다.

아래는 클래우드 환경에 대한 분류입니다. 우리가 사용하는 구름IDE는 PaaS에 해당합니다.

1. SaaS(Software as a Service)

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.

1.1 Django설치 및 세팅 환경소개 (계속)

이 책을 보시는 분들 중 대부분의 분들이 Python에 강력한 라이브러리, 직관적이면서도 뛰어난 확장성에 반해 Django로 오셨다고 생각이 됩니다. Python은 이제 배우지 않을 수 없을 정도로 인지도가 올라가 있는 상태입니다.

그렇다면 프레임 워크는 어떨까요? 아래는 wiki에서 제공하고 있는 python Framework에 관련된 설명입니다. Django 말고도 Flask 같은 유명한 프레임 워크가 있습니다. Django가 Full-stack인 반면에 Flask는 Non Full-stack입니다. 용도에 맞게 적절히 사용하시면 좋을 것 같습니다.

Popular Full-Stack Frameworks

A web application may use a combination of a base HTTP application server, a storage mechanism such as a database, a template engine, a request dispatcher, an authentication module and an AJAX toolkit. These can be individual components or be provided together in a high-level framework.

These are the most popular high-level frameworks. Many of them include components listed on the [WebComponents](#) page.

Name	Latest version	Latest update date	description
Django	1.11	2017-04-04	The Web framework for perfectionists (with deadlines). Django makes it easier to build better Web apps more quickly and with less code. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. It lets you build high-performing, elegant Web applications quickly. Django focuses on automating as much as possible and adhering to the DRY (Don't Repeat Yourself) principle. See Django
TurboGears	2.3.10	2016-12-04	the rapid Web development webframework you've been looking for. Combines SQLAlchemy (Model) or Ming (MongoDB Model), Genshi (View), Repoze and Tosca Widgets . Create a database-driven, ready-to-extend application in minutes. All with designer friendly templates, easy AJAX on the browser side and on the server side, with an incredibly powerful and flexible Object Relational Mapper (ORM), and with code that is as natural as writing a function. After reviewing the Documentation , check out the Tutorials
web2py	2.14.6	2016-05-10	* Python 2.6 to 2.7. Python 3.x friendly (compile but not tested no support yet) * All in one package with no further dependencies. Development, deployment, debugging, testing, database administration and maintenance of applications can be done via the provided web interface, but not required. * web2py has no configuration files, requires no installation, can be run off a USB drive. * web2py uses Python for the Model, View and the Controller * Built-in ticketing system to manage errors * Internationalization engine and pluralisation, caching system * Flexible authentication system (LDAP, MySQL, janrain etc) * NIX(Linux, BSD), Windows, Mac OSX, tested on EC2, Webfaction * works with MySQL, PostgreSQL, SQLite , Firebird, Oracle, MSSQL and the Google App Engine via an ORM abstraction layer. * Includes libraries to handle HTML/XML, RSS, ATOM, CSV, RTF, JSON, AJAX, XMLRPC, WIKI markup. * Production ready, capable of upload/download of very large files * Emphasis on backward compatibility.

Popular Non Full-Stack Frameworks

These projects provide the base "application server", either running as its own independent process, upon Apache or in other environments. On many of these you can then introduce your own choice of templating engines and other components to run on top, although some may provide technologies for parts of the technology stack.

- » [Bottle](#) (0.12.13 Released 2017-01-09) is a fast and simple micro-framework for small web-applications. It offers request dispatching (Routes) with url parameter support, Templates, keyValue Databases, a build-in HTTP Server and adapters for many third party WSGI/HTTP-server and template engines. All in a single file and with no dependencies other than the Python Standard Library.
- » [CherryPy](#) (10.2.1 Released 2017-03-13) is a pythonic, object-oriented HTTP framework. CherryPy powered web applications are in fact stand-alone Python applications embedding their own multi-threaded web server. TurboGears, web2py (see above) also use CherryPy.
- » [Flask](#) (0.12.1 Released 2017-03-31) is "a microframework for Python based on Werkzeug, Jinja 2 and good intentions." Includes a built-in development server, unit testing support, and is fully Unicode-enabled with RESTful request dispatching and WSGI compliance.
- » [Hug](#) (2.2.0 Released 2016-10-17) Embrace the APIs of the future. Hug aims to make developing APIs as simple as possible, but no simpler. It's one of the first fully future looking frameworks: only supporting Python3+.
- » [Pyramid](#) (1.8.3 Released 2017-03-12) a small, fast, down-to-earth, open source Python web development framework. It makes real-world web application development and deployment more fun, more predictable, and more productive. Pyramid is a Pylons Project, and is the successor to the Pylons web framework.

<https://wiki.python.org/moin/WebFrameworks>

Tutorial 1.2

환경설정

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



<https://ide.goorm.io/>

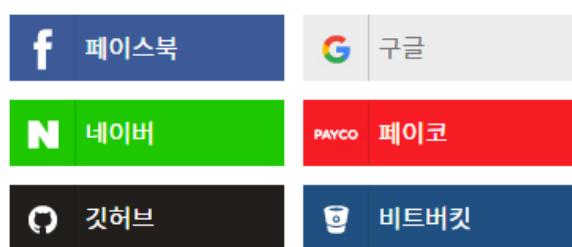
1.2 환경설정 (계속)

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

로그인

이메일 저장

다른 서비스 계정으로 로그인



[비밀번호 재설정](#)

[회원가입](#)

goormide

[가격 정책](#)

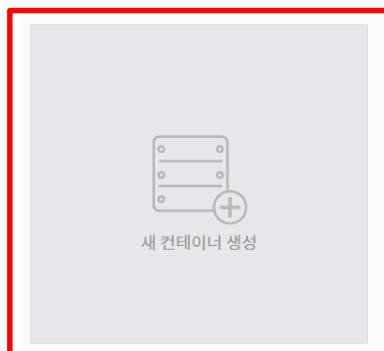
[대시보드](#)

[릴리즈 노트](#)

[친구 초대](#)

1.2 환경설정 (계속)

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



컨테이너 생성

컨테이너 생성

소스	<input checked="" type="radio"/> 템플릿 <input type="radio"/> Github <input type="radio"/> Bitbucket <input type="radio"/> Git / SVN <input type="radio"/> 압축파일																														
이름	tutorialdjango																														
설명	여행블로그 튜토리얼로 배우는 <u>Django</u>																														
소프트웨어 스택 선택	<table border="1"><tr><td>Search</td></tr><tr><td>C/C++</td></tr><tr><td>Python</td></tr><tr><td>Django</td></tr><tr><td>Flask</td></tr><tr><td>Jupyter Notebook</td></tr><tr><td>TensorFlow</td></tr><tr><td>PyQt</td></tr><tr><td>JAVA</td></tr><tr><td>Maven</td></tr></table> <p>The "Python" option is highlighted with a red box.</p> <table border="1"><tr><td>프로젝트 유형</td><td>Python 프로젝트</td></tr><tr><td>OS</td><td>Ubuntu 14.04 LTS</td></tr><tr><td>Python3</td><td>3.4.3</td></tr><tr><td>Python</td><td>2.7.6</td></tr><tr><td>pip3</td><td>9.0.1</td></tr><tr><td>pip</td><td>9.0.1</td></tr><tr><td>Jupyter</td><td>4.3.0</td></tr><tr><td>Django</td><td>1.11.12 LTS</td></tr><tr><td>Flask</td><td>0.12</td></tr><tr><td>TensorFlow</td><td>1.3.0</td></tr></table>	Search	C/C++	Python	Django	Flask	Jupyter Notebook	TensorFlow	PyQt	JAVA	Maven	프로젝트 유형	Python 프로젝트	OS	Ubuntu 14.04 LTS	Python3	3.4.3	Python	2.7.6	pip3	9.0.1	pip	9.0.1	Jupyter	4.3.0	Django	1.11.12 LTS	Flask	0.12	TensorFlow	1.3.0
Search																															
C/C++																															
Python																															
Django																															
Flask																															
Jupyter Notebook																															
TensorFlow																															
PyQt																															
JAVA																															
Maven																															
프로젝트 유형	Python 프로젝트																														
OS	Ubuntu 14.04 LTS																														
Python3	3.4.3																														
Python	2.7.6																														
pip3	9.0.1																														
pip	9.0.1																														
Jupyter	4.3.0																														
Django	1.11.12 LTS																														
Flask	0.12																														
TensorFlow	1.3.0																														

생성하기

A screenshot of a Docker Compose configuration page. It shows a form for creating a new container named "tutorialdjango" with a descriptive note about learning Django. Under "Software Stack Selection", the "Python" option is highlighted with a red box. On the right, there's a detailed list of Python-related configurations like OS, Python version, and various libraries. At the bottom right is a prominent blue "Create" button, which is also highlighted with a red box.

1.2 환경설정 (계속)

컨테이너가 만들어지고 있다는 화면이 나오고 곧 컨테이너 생성이 완료되었다고 뜹니다. 컨테이너 실행을 누르셔서 컨테이너 안으로 들어가주세요. 만약 모르고 닫으셨다면 대시보드에서 해당 컨테이너 실행을 누르시면 됩니다.

The image consists of three screenshots from a Docker interface:

- Screenshot 1:** Shows a progress bar at 14% completion. Below it, a message says "곧 멋진 작업공간이 생성됩니다. 잠시만 기다려주세요." (A nice workspace will be created soon. Please wait a moment.)
- Screenshot 2:** A modal window titled "컨테이너 실행" (Container Run) asking "컨테이너를 실행하시겠습니까?" (Do you want to run the container?). It has two buttons: "대시보드로 이동" (Move to Dashboard) and a redboxed "실행" (Run) button.
- Screenshot 3:** The container details page for "tutorialdjango". It shows the last update was "2018. 4. 30. Updated". It lists the following configuration:
 - 소프트웨어 스택: Python
 - 저장 공간: 10GB
 - 항상 켜두기: PREMIUM 상품 구매 시 사용 가능
 - SSH: 컨테이너 실행 시 사용 가능A red box highlights the "실행" (Run) button at the bottom of the container card.

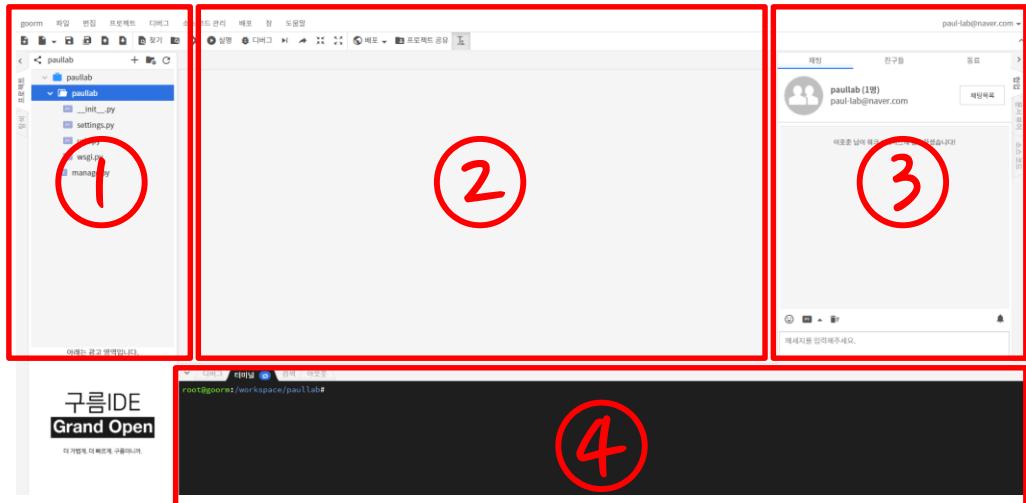
1.2 환경설정 (계속)

컨테이너가 로딩되는 화면입니다. 주로 구름에 TIP이나 유명인사의 명언이 나옵니다.



공유된 프로젝트에서 동료의 커서를 보고 싶지 않다면 '프로젝트 > 커서 공유 토글'의 체크 상태를 해제해 주세요.

1.2 환경설정 (계속)

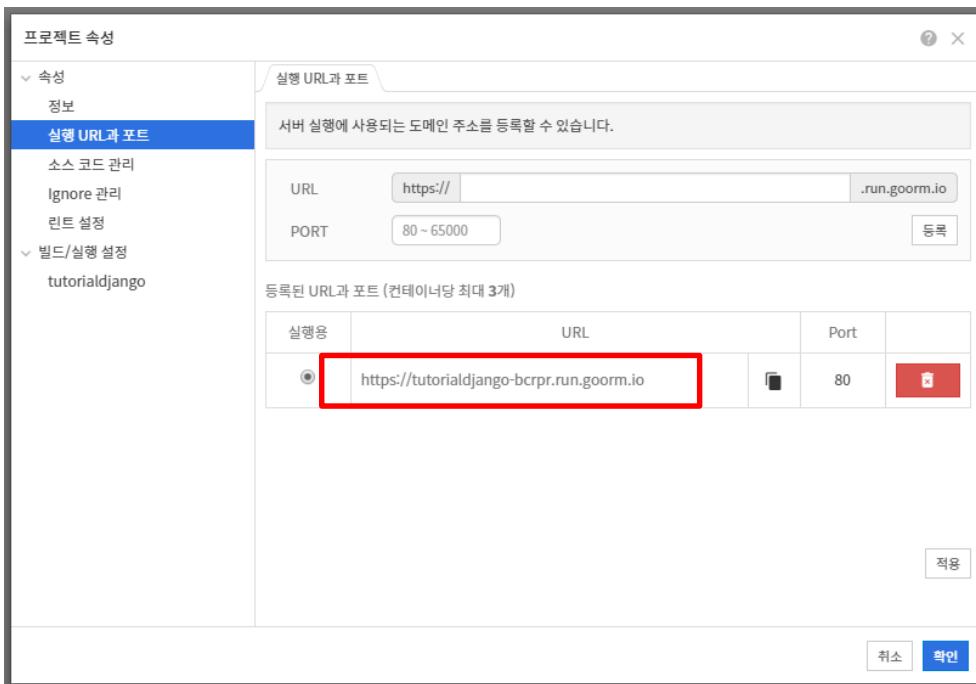
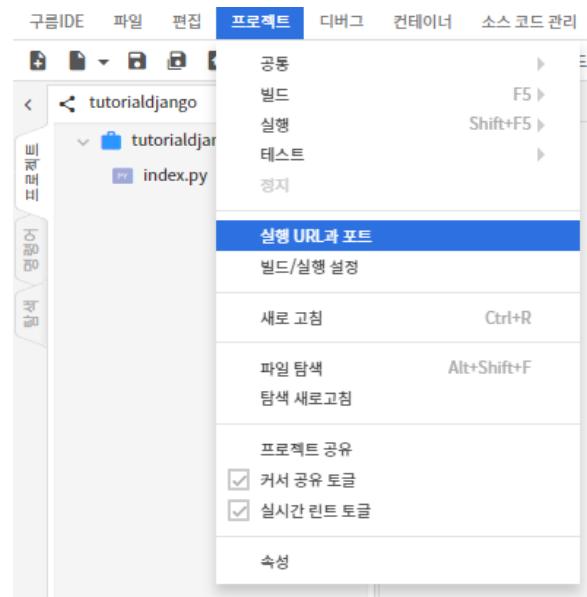


1. 폴더 구조를 볼 수 있는 공간입니다. 우측 상단에 보시면 새로고침 버튼이 있어요. 뭔가 작업을 했는데 보이지 않는다면 새로고침을 해보시기 바랍니다.
2. txt, html, py파일 등 다양한 파일을 edit 할 수 있는 공간입니다.
3. 협업을 위한 공간이에요. 이 책에서는 사용하지 않는 기능입니다. 채팅 등 협업을 위한 다양한 기능을 제공합니다.
4. 콘솔창입니다. 우리는 대부분의 명령어를 이곳에 입력하게 됩니다.

1.2 환경설정 (계속)

프로젝트를 누르시고 실행 URL과 포트를 눌러주세요. 아래 빨간색 네모로 표시된 것이 우리가 앞으로 사용할 URL입니다. 아직은 실행되지 않습니다.

* 만약 등록이 되어 있지 않다면 위에서 원하는 url을 입력하고 PORT는 80으로 하여 등록버튼을 눌러주세요.



1.2 환경설정 (계속)

가장 먼저 해야 할 것은 홈페이지 기획 및 구성입니다. 이 책에서는 기획 단계를 서술하지 않습니다.(여행블로그 튜토리얼로 배우는 bootstrap에서 다룹니다.)

여기서는 전체 URL 구조만 살펴보도록 하겠습니다.

1. https://사용자_지정_도메인.run.goorm.io/ – main page

- 메인페이지, 서비스 소개, google map api, 최근 post 소개
- Template : index.html

2. https://사용자_지정_도메인.run.goorm.io/blog – blog page

- 메인페이지에서 blog를 눌렀을 때 이동, post List page
- Template : blog.html

3. https://사용자_지정_도메인.run.goorm.io/post번호 – contents page

- blog 페이지에서 post를 눌렀을 때 이동
- Template : contents.html

1.2 환경설정 (계속)

Main page 입니다. 안에 있는 글귀들은 로렘입숨(임의의 글귀)으로 작성되었습니다.

The screenshot shows a travel blog's main page layout. At the top, there is a header with navigation links (Travel, Blog, Home, About, Blog) and a search bar. Below the header is a large landscape photograph of a mountainous coastline with a horse grazing in the foreground. Overlaid on this image is a placeholder text: "Example headline. Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta gravida at eget metus. Nullam id dolor id nibh ultricies vehicula ut id sit." A blue "Sign up today" button is positioned below the headline. Below the main image is a world map with several location markers and small thumbnail images. Three specific locations are highlighted with larger thumbnail images and circular "Heading" sections:

- Heading**

Donec sed odio dictum portas sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Praesent commodo cursus magna.

[View details >](#)
- Heading**

Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Orci mattis consectetur purus sit amet fermentum. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, pulvinar aenean massa ut enim rhoncus.

[View details >](#)
- Heading**

Donec sed odio dictum portas sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Praesent commodo cursus magna.

[View details >](#)

Below these featurettes is a section titled "First featurette heading. It'll blow your mind." with a placeholder text: "Donec ullamcorper nulla non metus auctor fringilla. Vestibulum id ligula porta felis euismod semper. Praesent commodo cursus magna, vel scelerisque nisl consectetur. Fusce dapibus, tellus ac cursus commodo." To the right of this text is a small image of a rocky cliffside.

Further down the page is another section titled "Oh yeah, it's that good. See for yourself." with a placeholder text: "Donec ullamcorper nulla non metus auctor fringilla. Vestibulum id ligula porta felis euismod semper. Praesent commodo cursus magna, vel scelerisque nisl consectetur. Fusce dapibus, tellus ac cursus commodo." This section includes a large image of a coastal landscape with a bridge.

At the bottom of the page, there is a footer with copyright information ("© 2017-2018 Company, Inc. - Privacy - Terms") and a "Back to top" link.

1.2 환경설정 (계속)

blog page 입니다. 역시 안에 있는 글귀들은 임의의 글귀로 작성되었습니다.

Travel Blog Home About Blog Search Search

Album example

Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but not too short so folks don't simply skip over it entirely.

Main call to action Secondary action

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.
View Edit 9 mins

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.
View Edit 9 mins

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.
View Edit 9 mins

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.
View Edit 9 mins

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.
View Edit 9 mins

© 2017-2018 Company, Inc. · [Privacy](#) · [Terms](#) Back to top

1.2 환경설정 (계속)

contents page 입니다. 역시 안에 있는 글귀들은 임의의 글귀로 작성되었습니다.

Travel Blog Home About Blog

Search

Search

From the Firehose

Sample blog post

January 1, 2014 by [Mark](#)

This blog post shows a few different types of content that's supported and styled with Bootstrap. Basic typography, images, and code are all supported.

Cum sociis natoque penatibus et magnis **dis parturient montes**, nascetur ridiculus mus. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum. Sed posuere consectetur est at lobortis. Cras mattis consectetur purus sit amet fermentum.

Curabitur blandit tempus porttitor. **Nullam quis risus eget urna mollis** ornare vel eu leo. Nullam id dolor id nibh ultricies vehicula ut id elit.

Etiam porta sem malesuada magna mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Heading

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros.

Sub-heading

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

[Example code block](#)

Aenean lacinia bibendum nulla sed consectetur. Etiam porta sem malesuada magna mollis euismod. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa.

[Older](#)

[Newer](#)

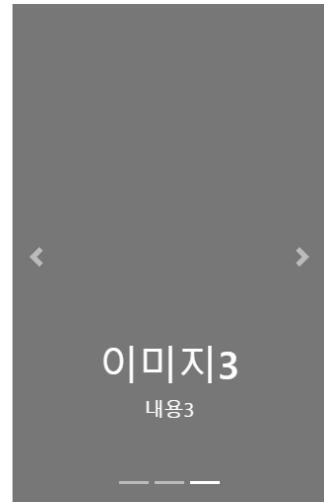
© 2017-2018 Company, Inc. · [Privacy](#) · [Terms](#)

[Back to top](#)

About

Etiam porta sem malesuada magna mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

이미지 캐러셀 1



1.2 환경설정 (계속)

본격적인 화면 구성입니다. 해당 명령은 4번 콘솔창에서 입력해주세요.

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

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

```
root@goorm:/workspace/컨테이너명/mysite# apt-get install python-virtualenv
```

```
root@goorm:/workspace/컨테이너명/mysite#
```

```
virtualenv --python=python3.4 myvenv
```

```
root@goorm:/workspace/컨테이너명/mysite# source myvenv/bin/activate
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite#
```

```
pip install django whitenoise
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite#
```

```
django-admin startproject 컨테이너이름 .
```

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py
```

```
migrate
```

'startproject 컨테이너이름' 뒤에 '.'이 있으니 주의해 주세요.

1.2 환경설정 (계속)

```
root@goorm:/workspace/tutorialdjango# mkdir mysite
root@goorm:/workspace/tutorialdjango# cd mysite
root@goorm:/workspace/tutorialdjango/mysite# apt-get install python-virtualenv
파키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 새 패키지를 설치할 것입니다:
  python-virtualenv
0개 업그레이드, 1개 새로 설치, 0개 제거 및 22개 업그레이드 안 함.

root@goorm:/workspace/tutorialdjango/mysite# virtualenv --python=python3.4 myvenv
Running virtualenv with interpreter /usr/bin/python3.4
Using base prefix '/usr'
New python executable in myvenv/bin/python3.4
Also creating executable in myvenv/bin/python
Installing setuptools, pip...done.

root@goorm:/workspace/tutorialdjango/mysite# source myvenv/bin/activate
(myvenv)root@goorm:/workspace/tutorialdjango/mysite#
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# pip install django whitenoise
Downloading/unpacking django
  Downloading Django-2.0.4-py3-none-any.whl (7.1MB): 7.1MB downloaded
Downloading/unpacking whitenoise
  Downloading whitenoise-3.3.1-py2.py3-none-any.whl
Downloading/unpacking pytz (from django)
  Downloading pytz-2018.4-py2.py3-none-any.whl (510kB): 510kB downloaded
Installing collected packages: django, whitenoise, pytz
Successfully installed django whitenoise pytz
Cleaning up...

(myvenv)root@goorm:/workspace/tutorialdjango/mysite# django-admin startproject tutorialdjango .
(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 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
```

1.2 환경설정 (계속)

사용하셨던 명령어를 해석하는 부분입니다.

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

Python의 버전을 확인합니다. 2.x버전으로 되어있지요? 우리는 3.4 버전으로 개발을 할 것입니다.

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

mkdir은 디렉토리를 만들어주는 명령어 입니다. mysite라는 폴더를 만들 것입니다.

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

cd명령어는 change directory 명령어 입니다. 우리가 앞서 만든 mysite라는 폴더로 이동합니다.

```
root@goorm:/workspace/컨테이너명/mysite# apt-get install python-virtualenv
```

가상환경 설정을 위한 명령어 입니다. 가상환경이란 현재 2.x 버전으로 실행되고 있는 환경을 마치 3.X 버전으로 쓰게끔 만들어주는 가상의 환경이에요.

```
root@goorm:/workspace/컨테이너명/mysite# virtualenv --python=python3.4 myvenv
```

3.4버전으로 사용을 하겠다는 것입니다.

```
root@goorm:/workspace/컨테이너명/mysite# source myvenv/bin/activate
```

이 명령어를 실행하고 앞서 실행한 python --version을 실행하면 3.4버전으로 나옵니다.

이 명령어는 구름IDE 컨테이너를 다시 실행할때마다 쳐주셔야 합니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install django whitenoise
```

패키지까지 설치하기 위한 명령어 입니다. Django만 설치해주셔도 됩니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# django-admin startproject 컨테이너이름 .
```

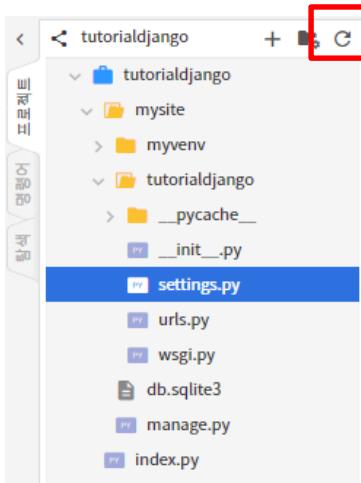
현재 있는 폴더에 프로젝트를 시작하겠다는 명령어에요.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py migrate
```

Migrate 명령어는 하편에서 설명해 드리도록 하겠습니다. 쉽게 말해 DB에 값을 넣는 작업입니다.

1.2 환경설정 (계속)

이제 settings.py로 이동하셔서 28번째 줄을 아래와 같이 고쳐주세요. 그런 다음 아래 명령어를 콘솔에 입력해주세요.



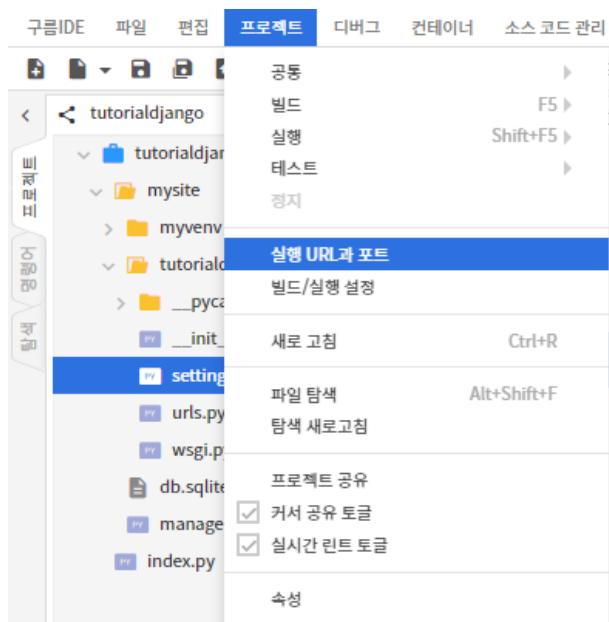
28 ALLOWED_HOSTS = ['*']

Settings.py에 28번째 줄에 가셔서 해당 값을 '*'로 바꿔 주십시오. 모든 사용자의 접속을 허락하겠다는 것입니다.

(myvenv)root@goorm:/workspace/컨테이너명/mysite# python manage.py runserver 0:80

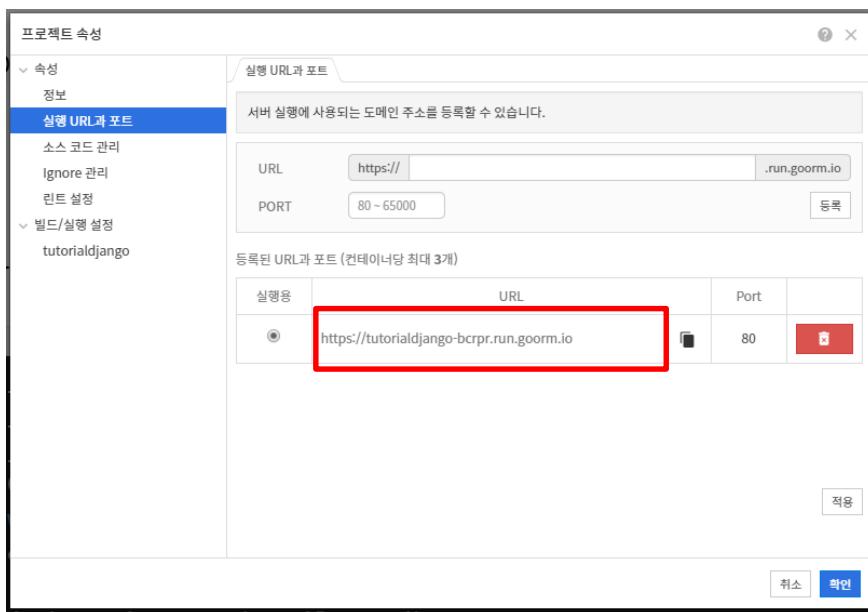
```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py runserver 0:80
Performing system checks...

System check identified no issues (0 silenced).
April 30, 2018 - 07:10:49
Django version 2.0.4, using settings 'tutorialdjango.settings'
Starting development server at http://0:80/
Quit the server with CONTROL-C.
```



1.2 환경설정 (계속)

해당 url을 클릭하시면 아래와 같이 실행화면이 뜹니다. Django가 실행되고 있어요!



django View release notes for Django 2.0



Step_2

Django로 Main page 만들기

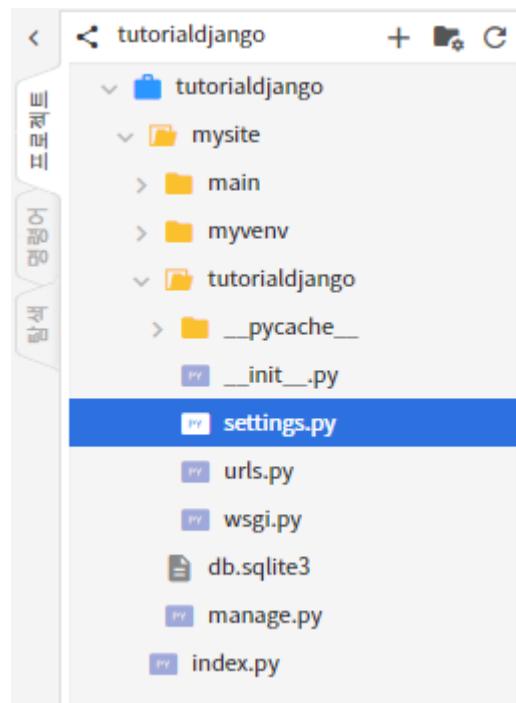
step 2 Django로 Main page 만들기(계속)

Django는 여러 개의 앱을 동시에 만들어 조립하는 과정으로 보실 수 있어요. 우리는 간단한 실습이기 때문에 main이라는 앱만을 만들 것입니다. 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.

* Main뒤에 쉼표가 있으니 꼭 체크하세요.

```
root@goorm:/workspace/컨테이너명/mysite# python manage.py startapp main
```

```
(myvenv) root@goorm:/workspace/tutorialdjango/mysite# python manage.py startapp main  
(myvenv) root@goorm:/workspace/tutorialdjango/mysite# █
```



```
33 INSTALLED_APPS = [  
34     'django.contrib.admin',  
35     'django.contrib.auth',  
36     'django.contrib.contenttypes',  
37     'django.contrib.sessions',  
38     'django.contrib.messages',  
39     'django.contrib.staticfiles',  
40     'main',  
41 ]
```

step 2 Django로 Main page 만들기(계속)

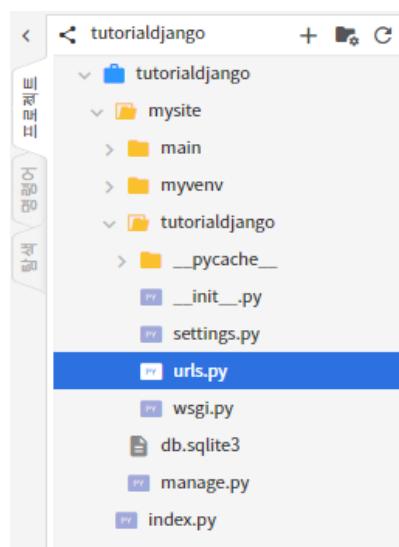
이제 urls.py를 작성할 것입니다. urls는 사용자가 어떤 url을 사용하여 들어오느냐에 따라 어떤 화면을 보여줄지를 결정하게 됩니다. 여기서

from main.views import index는

main폴더 안에 views.py파일 안에 index라는 함수를 연결시켜 주겠다는 뜻입니다. 아직 만들 어지지 않았기 때문에 오류가 뜹니다.

또한 path(“, index),

의 의미는 사용자가 뒤에 무언가 붙지 않은 있는 그대로의 url을 입력하고 들어왔을 경우 index라는 것을 연결시켜 주겠다는 얘기입니다. 여기서 index에는 index.html을 연결시켜 줄 것입니다.



mysite > tutorialdjango > urls.py

step 2 Django로 Main page 만들기(계속)

이제 mysite > product > views.py로 들어와 index라는 함수를 만듭니다. 여기서 사용자가 index.html을 볼 수 있게끔 연결을 해줍니다.

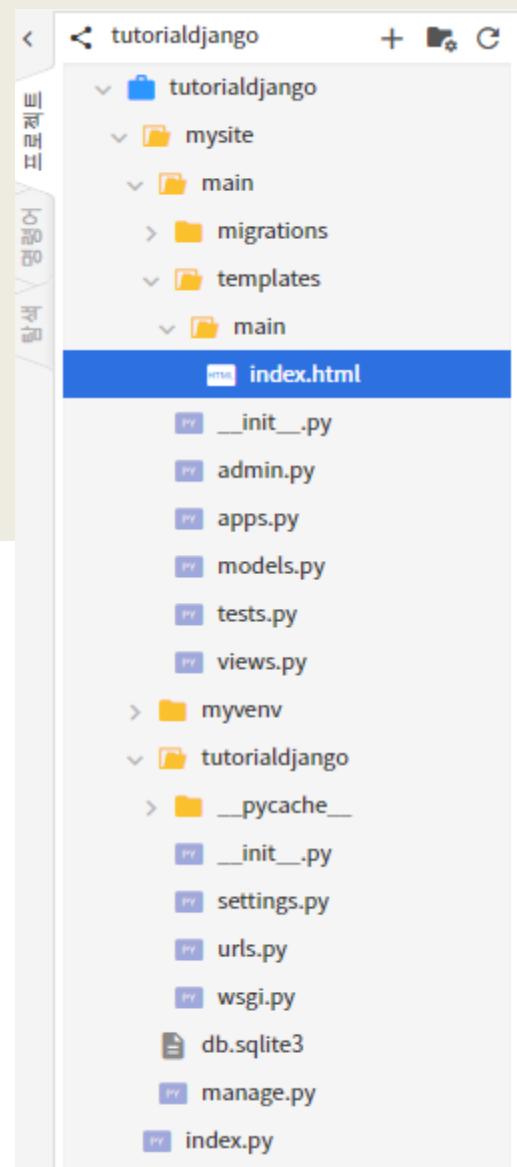
mysite > product > views.py

```
1 from django.shortcuts import render
2
3 def index(request):
4     return render(request, 'main/index.html')
```

step 2 Django로 Main page 만들기(계속)

Template을 연결 시키는 과정으로 mysite > main > templates > main > index.html 을 만들어 주세요. 기본적인 테스트를 위하여 아래 내용을 붙여 넣어 주세요.

```
<html>
<head>
    <title>Django!</title>
</head>
<body>
    <h1>Test!</h1>
</body>
</html>
```



```
1  <html>
2  <head>
3      <title>Django!</title>
4  </head>
5  <body>
6      <h1>Test!</h1>
7  </body>
8  </html>
```

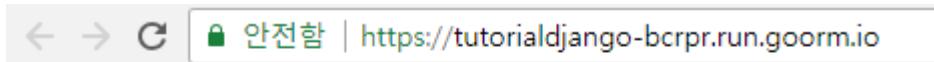
step 2 Django로 Main page 만들기(계속)

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
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py runserver 0:80
Performing system checks...

System check identified no issues (0 silenced).
April 30, 2018 - 07:34:58
Django version 2.0.4, using settings 'tutorialdjango.settings'
Starting development server at http://0:80/
Quit the server with CONTROL-C.
Performing system checks...
```

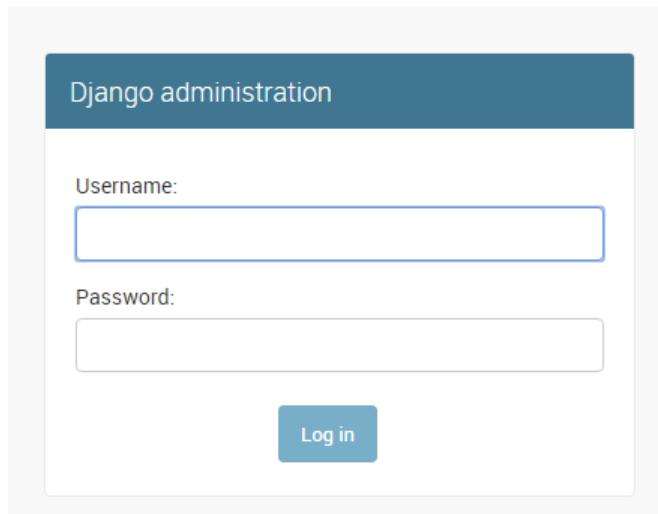


Test!

step 2 Django로 Main page 만들기(계속)

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.

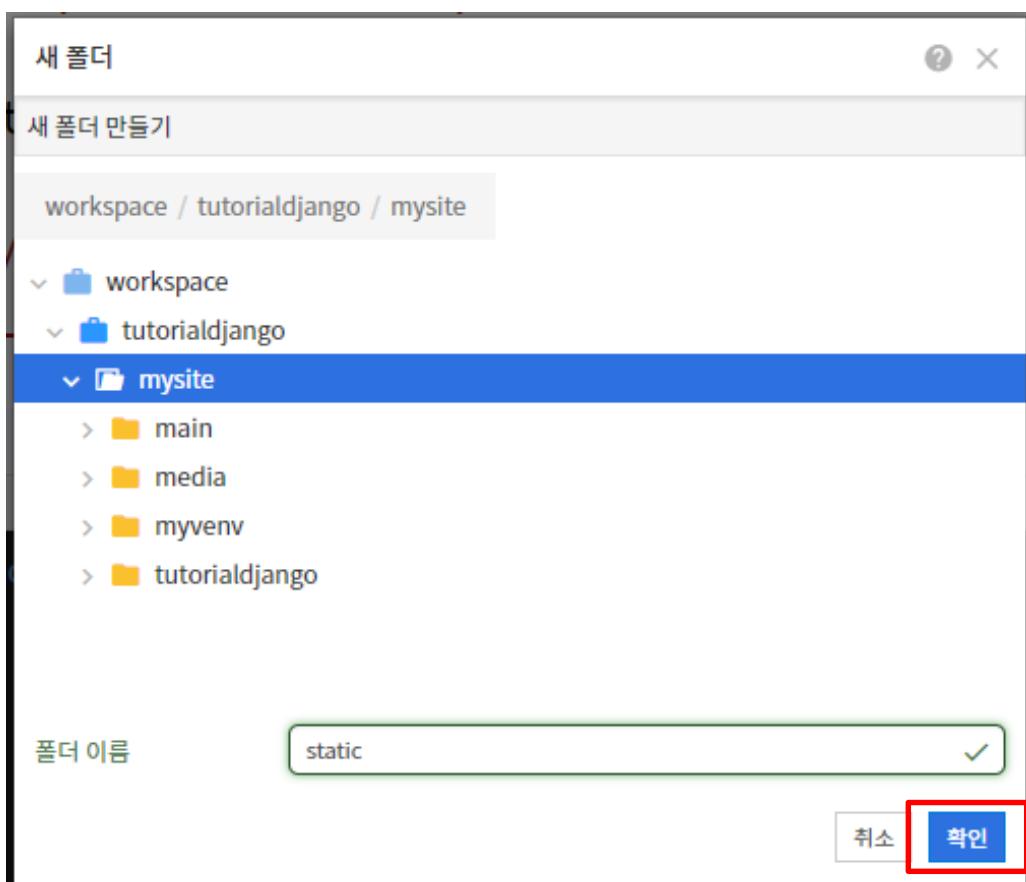


<https://여러분URL/admin>

step 2 Django로 Main page 만들기(계속)

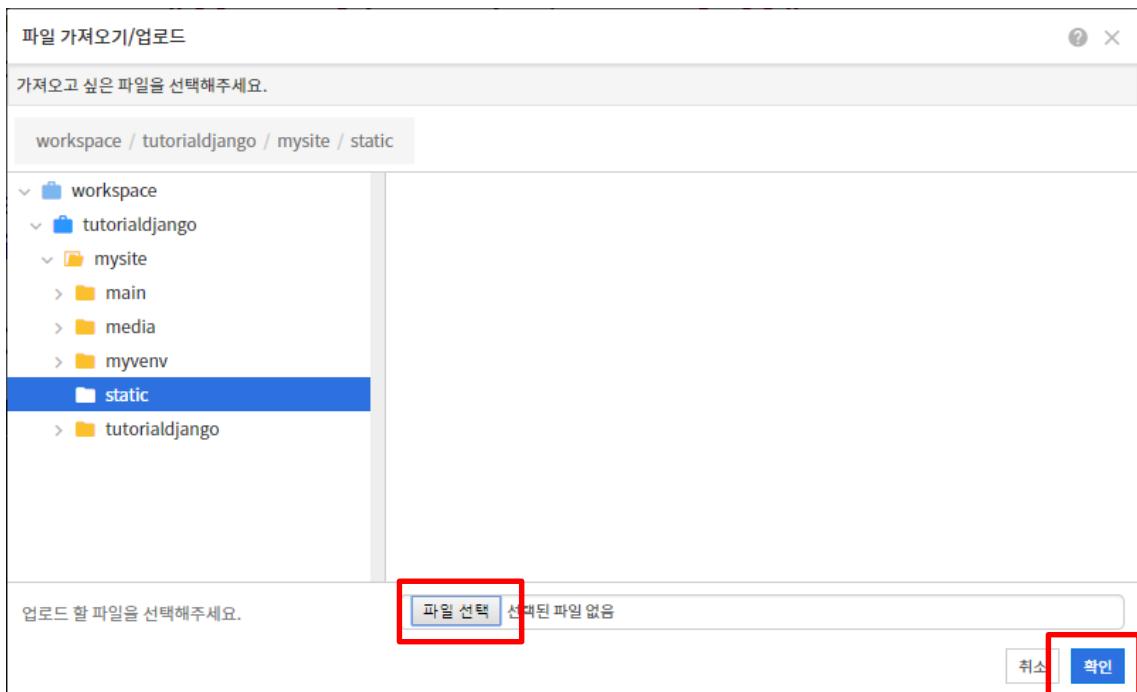
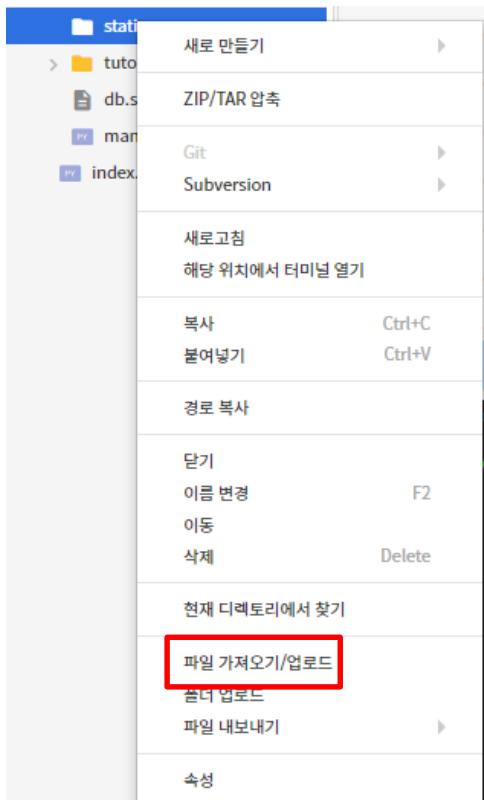
다음으로는 메인페이지에서 이미지를 불러오는 방법에 대해 소개하려 합니다. Django에서는 일반적인 HTML, CSS에서 지원하는 상대경로를 지원하지 않습니다. 우선 정적 파일들이 저장될 static이라는 폴더를 mysite 밑에 만들어 줍니다.

1) 새폴더 만들기



step 2 Django로 Main page 만들기(계속)

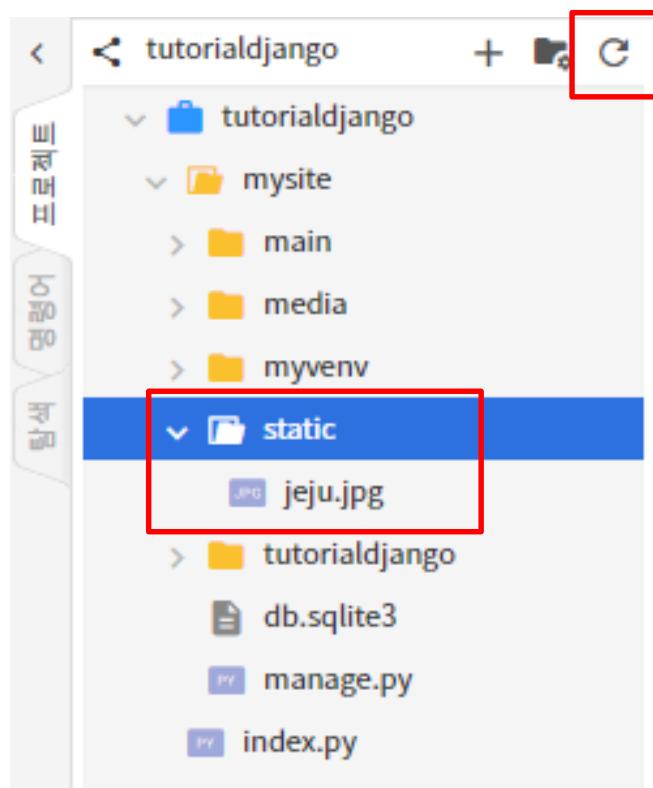
2) 파일 업로드



step 2 Django로 Main page 만들기(계속)

3) 파일 업로드 확인

혹시 이미지가 안떴을 경우에는 새로고침을 눌러주세요.



step 2 Django로 Main page 만들기(계속)

다음은 static 폴더를 django와 연결시켜 주는 작업이 남았습니다. settings.py로 이동하셔서 아래와 같이 수정해주세요. 뒤에 콤마가 있으니 빼먹지 않도록 주의해주세요. 기왕 온김에 Time Zone 변경하고 가도록 하겠습니다.

```
121 STATIC_URL = '/static/'  
122 STATICFILES_DIRS = (  
123     os.path.join(BASE_DIR, 'static'),  
124 )
```

```
109 TIME_ZONE = 'Asia/Seoul'  
110  
111 USE_I18N = True  
112  
113 USE_L10N = True  
114  
115 USE_TZ = True
```

step 2 Django로 Main page 만들기(계속)

이제 index.html로 들어가 static 파일을 호출하는 code를 넣어주세요. { % 문법 %}의 구조는 template tag라고 불립니다. 자세한 내용은 Django 공식문서 (<https://docs.djangoproject.com/en/2.0/ref/templates/builtins/>)를 참고하세요.

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Test!</h1>
7     {% load staticfiles %}
8     
9 </body>
10 </html>
```

tutorialdjango/mysite/main/templates/main/index.html

step 2 Django로 Main page 만들기(계속)

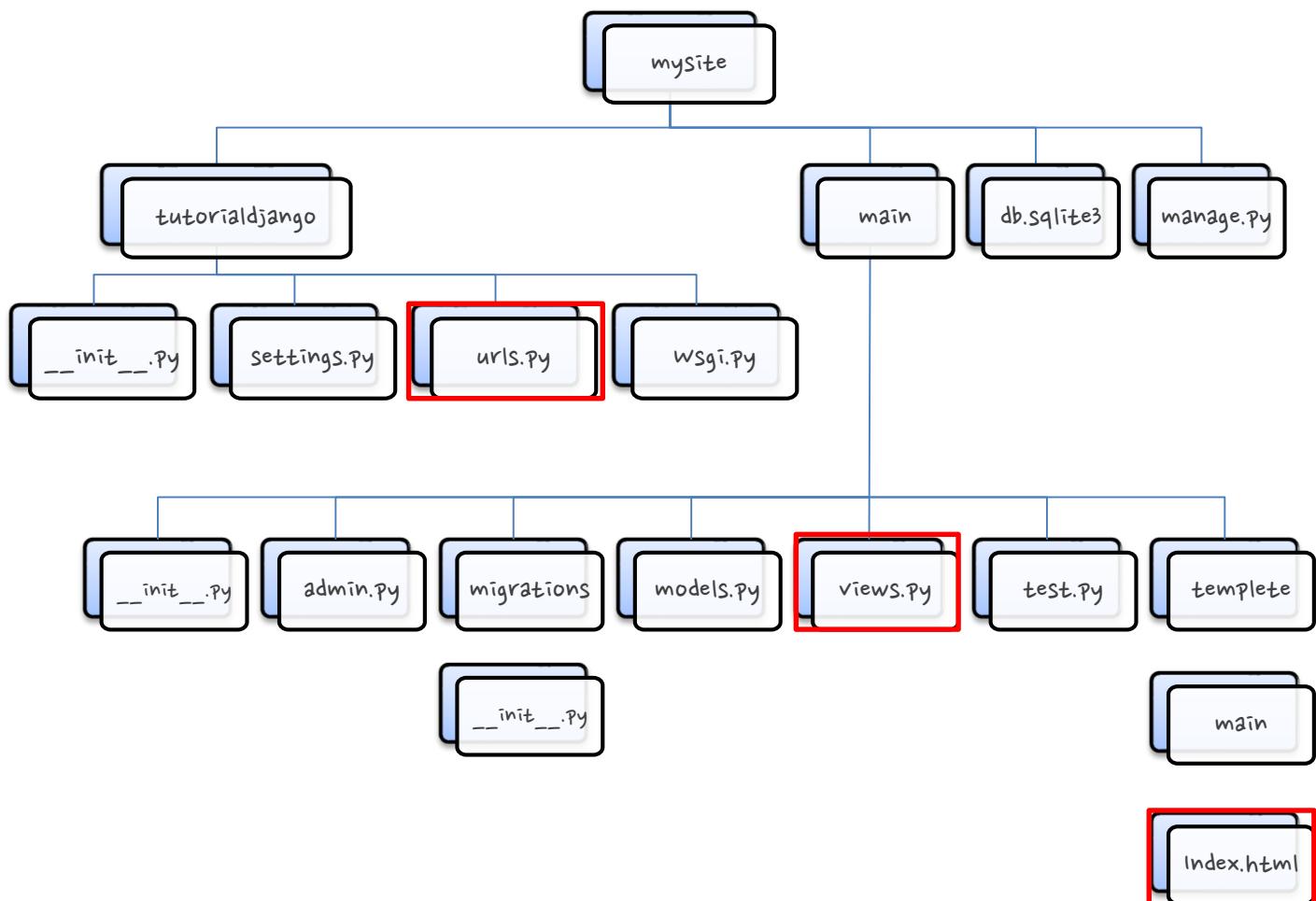
콘솔창에 `python manage.py runserver 0:800`이 입력이 안되어 있으시다면 입력하시고 프로젝트 > 실행URL과 포트를 누르신 다음 URL을 클릭하시면 아래와 같은 화면이 나옵니다.

Test!



step 2 Django로 Main page 만들기(계속)

현재 폴더 구성 트리입니다. 빨간색으로 네모를 친 것이 우리가 수정한 파일입니다.

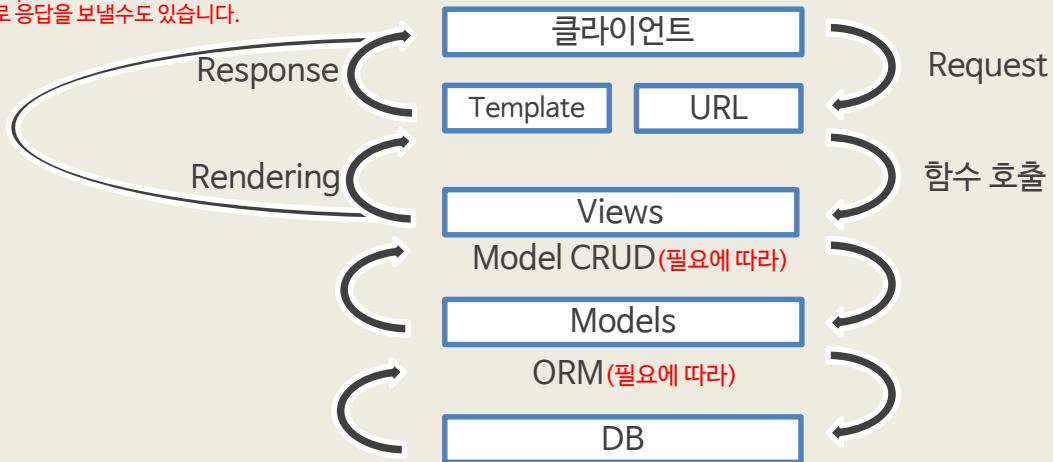


step 2 Django로 Main page 만들기(계속)

Django는 MVC모델을 MTV라고 부릅니다. 용어가 바뀌었을 뿐이지 기본적인 개념은 같습니다.
아래 표는 Django 작동원리를 도식화 한 것입니다. 8장에서 좀 더 상세하게 다룹니다.

MTV(빨간색으로 표시된 부분은 설계에 따라 작동하지 않을 수도 있습니다.)

Template를 거치지 않고
바로 응답을 보낼 수도 있습니다.



Model - DB와 연결된 Python Class

Template - 사용자에게 response될 Client View

View - Django에서 처리한 데이터를 Template에게 전달

각 문자는 다음과 같이 표준 SQL문으로 대응 가능하다.

이름	조작	SQL
Create	생성	INSERT
Read(또는 Retrieve)	읽기(또는 인출)	SELECT
Update	갱신	UPDATE
Delete(또는 Destroy)	삭제(또는 파괴)	DELETE

CRUD는 대부분의 컴퓨터 소프트웨어가 가지는 기본적인 데이터 처리 기능인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 묶어서 일컫는 말입니다.

<https://ko.wikipedia.org/wiki/CRUD>

Step_3

Django로 Blog page 만들기

step 3 Django로 Blog page 만들기(계속)

이번에는 포스팅이 모아져 있는 blog 페이지를 만들어 보도록 하겠습니다. 우선 urls.py 파일을 수정하고 views.py 파일을 수정합니다.

```
16 from django.contrib import admin
17 from django.urls import path
18 from main.views import index, blog
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('', index),
23     path('blog/', blog),
24 ]
```

tutorialdjango/mysite/tutorialdjango/urls.py

```
3 def index(request):
4     return render(request, 'main/index.html')
5
6 def blog(request):
7     return render(request, 'main/blog.html')
```

tutorialdjango/mysite/main/views.py

step 3 Django로 Blog page 만들기(계속)

이제 html 파일을 만들 차례입니다. blog.html 파일을 mysite/main/templates/blog.html에 작성해주세요.

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Blog Page!</h1>
7 <body>
8 <html>
```

/mysite/main/templates/blog.html

step 3 Django로 Blog page 만들기(계속)

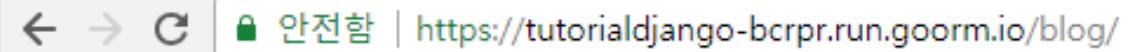
만약 중간에 한번 나갔다 오셨다면 아래 명령어처럼 쳐주세요. 계속해서 진행하고 계셨다면
python manage.py runserver 0:80
만 입력하시면 됩니다.

프로젝트 > 실행 URL과 포트를 누르시고 URL을 클릭하신 다음 뒤에 /blog라고 치시면 blog페이지가 나옵니다.

```
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
Performing system checks...

System check identified no issues (0 silenced).
May 02, 2018 - 06:25:59
Django version 2.0.4, using settings 'tutorialdjango.settings'
Starting development server at http://0:80/
Quit the server with CONTROL-C.
```

실행화면(<https://tutorialdjango-bcrpr.run.goorm.io/blog/>)



step 3 Django로 Blog page 만들기(계속)

이제 각각의 포스팅에 저장될 공간을 만들어보도록 하겠습니다. 우선 postname과 contents를 만듭니다. 사진을 저장할 공간은 다른 챕터에서 만들도록 하겠습니다.

tutorialdjango/mysite/main/models.py

```
1 from django.db import models
2
3 class Post(models.Model):
4     postname = models.CharField(max_length=50)
5     contents = models.TextField()
```

Ctrl + C를 누르시고 빠져나오신 다음 아래 명령어를 입력해주세요.

(myvenv)root@goorm:/workspace/컨테이너명/mysite#

python manage.py makemigrations main

(myvenv)root@goorm:/workspace/컨테이너명/mysite#

python manage.py migrate

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py makemigrations main
Migrations for 'main':
  main/migrations/0001_initial.py
    - Create model Post
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, main, sessions
Running migrations:
  Applying main.0001_initial... OK
```

step 3 Django로 Blog page 만들기(계속)

Admin 사이트에 등록 할 수 있도록 합니다. 이 공간에서 admin 사이트 설정을 할 수 있습니다.

tutorialdjango/mysite/main/admin.py

```
1 from django.contrib import admin  
2 from .models import Post  
3  
4 admin.site.register(Post)
```

step 3 Django로 Blog page 만들기(계속)

이제 Superuser를 만듭니다. Superuser는 게시물을 삭제, 수정, 저장할 수 있으며 다른 유저를 관리할 수 있습니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite#  
python manage.py createsuperuser
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py createsuperuser  
Username (leave blank to use 'root'): tutorialdjango  
Email address: tutorialdjango@gmail.com  
Password:  
Password (again):  
Superuser created successfully.
```

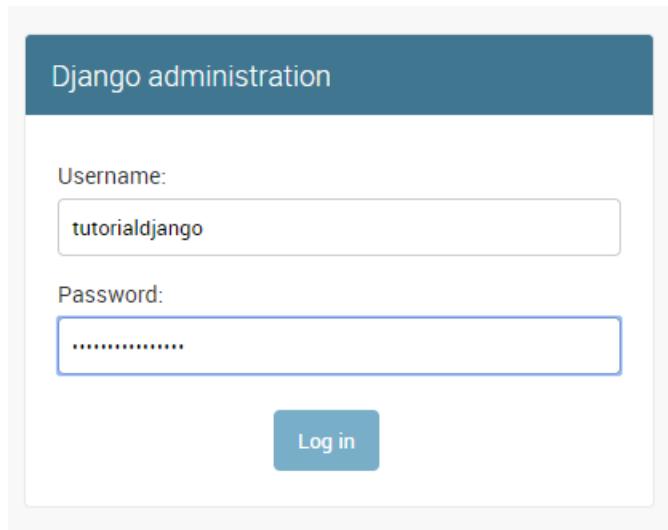
Admin user를 만들 것입니다.

비밀번호는 원래 콘솔창에 안뜨니 입력하시고 엔터를 치시면 됩니다.

```
python manage.py runserver 0:80
```

를 치셔서 Django를 실행시킨 다음 프로젝트 > 실행 URL과 포트를 누르시고 URL을 클릭하신다음 뒤에 /admin라고 치시면 admin페이지가 나옵니다.

<https://tutorialdjango-bcrpr.run.goorm.io/admin/>



step 3 Django로 Blog page 만들기(계속)

Admin 사이트 안으로 들어오셨다면 게시물을 올려보도록 하겠습니다. 구분을 위해서 postname과 contents의 내용을 다르게 입력해주세요.

Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change
Users	+ Add	Change

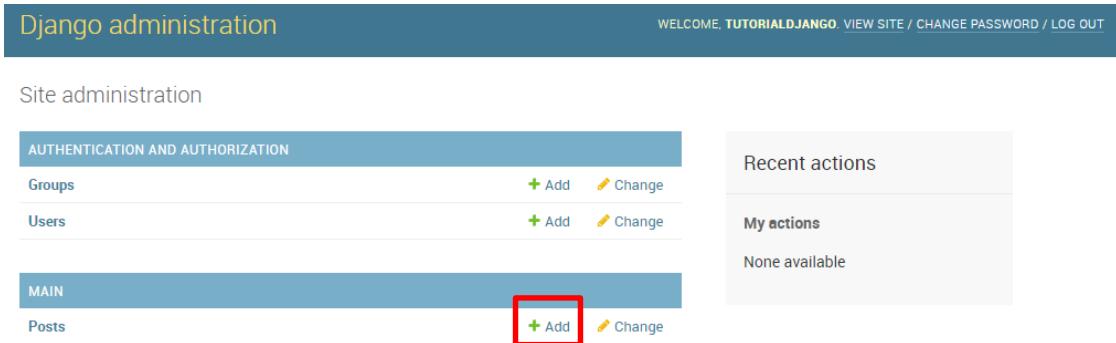
MAIN

Posts	+ Add	Change
-------	-------	--------

Recent actions

My actions

None available



- 캡쳐화면 -

Django administration

Welcome, TUTORIALDJANGO. [View Site](#) / [Change Password](#) / [Log Out](#)

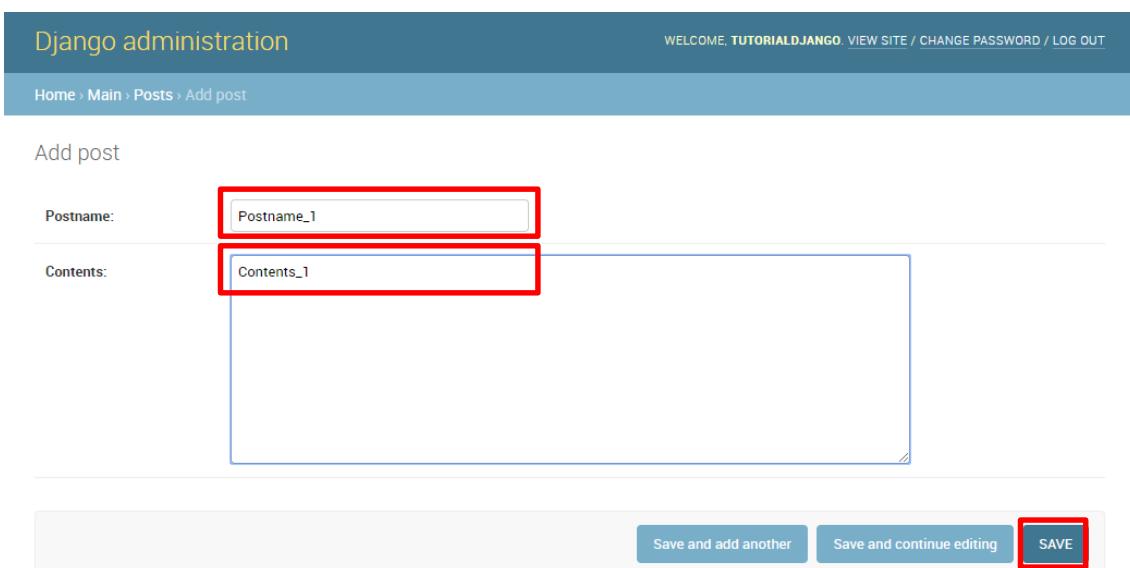
Home > Main > Posts > Add post

Add post

Postname: Postname_1

Contents: Contents_1

Save and add another Save and continue editing **SAVE**



- 캡쳐화면 -

step 3 Django로 Blog page 만들기(계속)

SAVE를 클릭하시면 제목이 postname_1이 아닙니다. 제목을 postname으로 출력하라는 명령이 없었기 때문입니다. 이게 제목이라는 것을 아는 것은 우리뿐이라는 것이죠. models.py로 들어가서 다시 제목이 무엇인지 설정해줄게요.

The screenshot shows the Django admin 'Posts' list page. At the top, it says 'Django administration' and 'WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT'. Below that, the breadcrumb navigation shows 'Home > Main > Posts'. A green success message box contains the text 'The post "Post object (1)" was added successfully.' On the right, there's an 'ADD POST +' button. The main list area has a table with one row. The first column has a checkbox labeled 'POST' and another checkbox labeled 'Post object (1)', which is highlighted with a red rectangle. The second column shows the count '1 post'.

mysite/product/models.py

```
1 from django.db import models
2
3 class Post(models.Model):
4     postname = models.CharField(max_length=50)
5     contents = models.TextField()
6
7     def __str__(self):
8         return self.postname
```

step 3 Django로 Blog page 만들기(계속)

브라우저 창에서 새로고침을 해보시면 이름이 바뀐 것을 볼 수 있습니다. 이제 새로운 포스팅을 몇 개 작성해 보겠습니다.(나중에 삭제할 게시물입니다.)

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Main > Posts

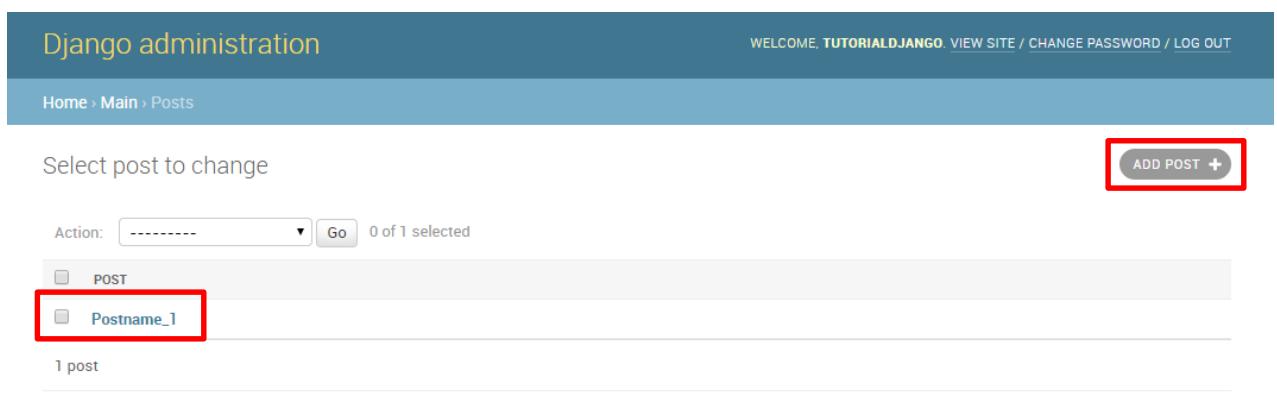
Select post to change

Action: ----- Go 0 of 1 selected

POST	Postname_1
<input type="checkbox"/>	Postname_1

1 post

ADD POST +



게시물 추가

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

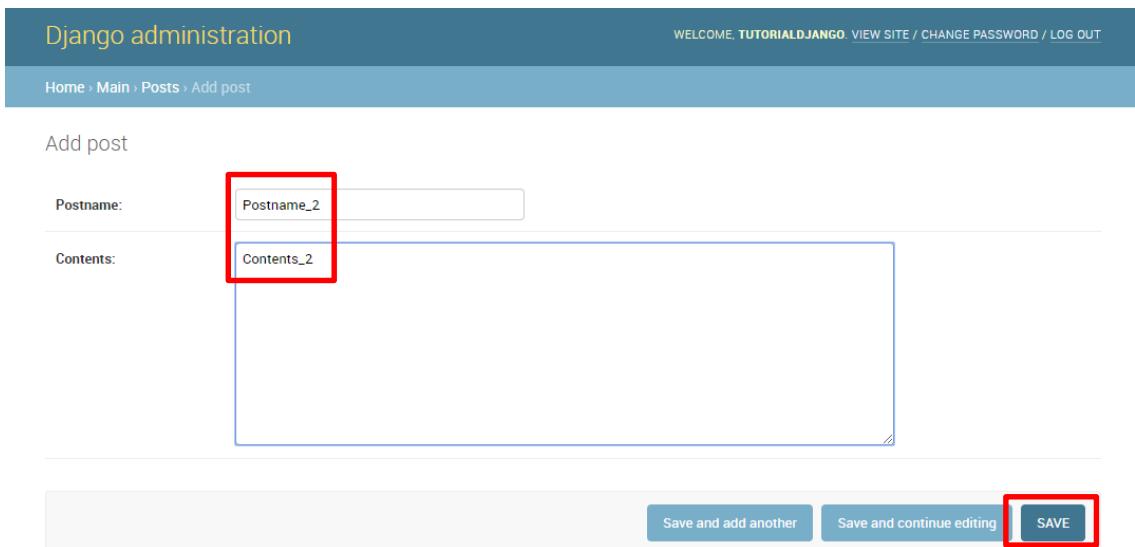
Home > Main > Posts > Add post

Add post

Postname: Postname_2

Contents: Contents_2

Save and add another Save and continue editing **SAVE**



step 3 Django로 Blog page 만들기(계속)

Test를 위해 3개의 게시물을 작성하였습니다. 이번에는 이 게시물을 blog 페이지에 띄어 보도록 하겠습니다. views 파일을 아래와 같이 수정해주세요.

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Main › Posts

The post "Postname_3" was added successfully.

Select post to change ADD POST +

Action: 0 of 3 selected

POST
 Postname_3
 Postname_2
 Postname_1

3 posts

tutorialdjango/mysite/main/views.py

```
1 from django.shortcuts import render
2 from .models import Post
3
4 def index(request):
5     return render(request, 'main/index.html')
6
7 def blog(request):
8     postlist = Post.objects.all()
9     return render(request, 'main/blog.html', {'postlist':postlist})
```

step 3 Django로 Blog page 만들기(계속)

Template tag를 사용하여 DB에 입력된 내용을 모두 가져옵니다. 템플릿 테그와 랭귀지에 관련된 자세한 내용은

* 공식 문서 : <https://docs.djangoproject.com/en/2.0/>

* templates language :

<https://docs.djangoproject.com/en/2.0/ref/templates/language/>

를 참고해주세요.

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Blog Page!</h1>
7     <table>
8         {% for list in postlist %}
9             <tr>
10                 <td>{{list.postname}}</td>
11                 <td>{{list.contents}}</td>
12             </tr>
13         {% endfor %}
14     </table>
15 </body>
16 </html>
```

Blog Page!

Postname_1 Contents_1

Postname_2 Contents_2

Postname_3 Contents_3

- 실행화면 -

step 3 Django로 Blog page 만들기(계속)

소스코드보기를 하면 우리가 작성했던 템플릿 랭귀지가 보이지 않습니다.

이렇게 사용자에게 보여질 때에는 html 형식으로 넘겨주게 됩니다.

Blog Page!

Postname_1 Contents_1

Postname_2 Contents_2

Postname_3 Contents_3

- 실행화면 -

```
1 <html>
2 <head>
3   <title>Django!</title>
4 </head>
5 <body>
6   <h1>Blog Page!</h1>
7   <table>
8     <tr>
9       <td>Postname_1</td>
10      <td>Contents_1</td>
11    </tr>
12
13    <tr>
14      <td>Postname_2</td>
15      <td>Contents_2</td>
16    </tr>
17
18    <tr>
19      <td>Postname_3</td>
20      <td>Contents_3</td>
21    </tr>
22
23  </table>
24 </body>
25 <html>
```

- 소스보기 -

Step_4

Django로 Post page 만들기

step 4 Django로 Post page 만들기(계속)

챕터가 넘어왔습니다. 아래 명령어는 구름IDE에서 다시 접속하실 때마다 실행해 주셔야 하는 명령어입니다.

```
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 권장해 드립니다.

(myvenv)는 (myvenv)가 붙지 않은 환경과 완전히 다른 환경이라고 보시면 됩니다.

step 4 Django로 Post page 만들기(계속)

이제 blog페이지에서 post를 눌렀을 때 나오는 상세 posting 화면을 만들어 보도록 하겠습니다.
홈페이지 전체 구성 중에서는 3번을 작업하는 중에 있습니다.

3. <http://사용자아이름.run.goorm.io/post번호>

- blog 페이지에서 post를 눌렀을 때 이동
- Template : postdetails.html

```
16 from django.contrib import admin
17 from django.urls import path
18 from main.views import index, blog, postdetails
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('', index),
23     path('blog/', blog),
24     path('blog/<int:pk>', postdetails),  

25 ]
```

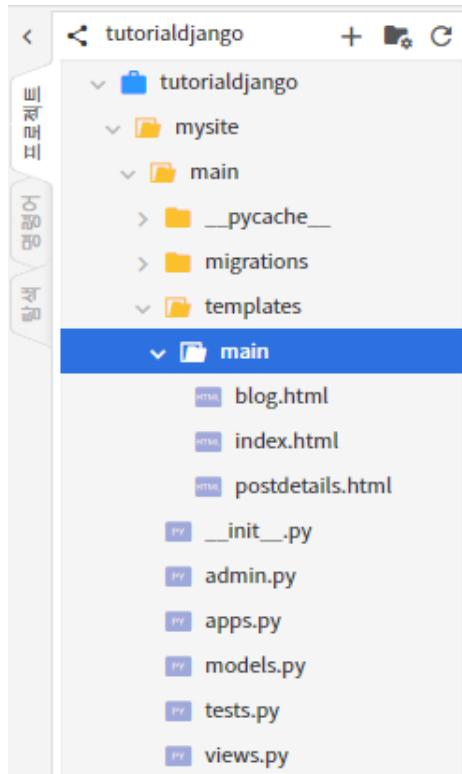
tutorialdjango/mysite/tutorialdjango/urls.py

```
1 from django.shortcuts import render
2 from .models import Post
3
4 def index(request):
5     return render(request, 'main/index.html')
6
7 def blog(request):
8     postlist = Post.objects.all()
9     return render(request, 'main/blog.html', {'postlist':postlist})
10
11 def postdetails(request, pk):
12     postlist = Post.objects.get(pk=pk)
13     return render(request, 'main/postdetails.html', {'postlist':postlist})
```

tutorialdjango/mysite/main/views.py

step 4 Django로 Post page 만들기(계속)

이제 아래 화면처럼 main 안에 템플릿은 3개가 됩니다. 마지막 템플릿입니다.(꾸미는 것은 7장에서 합니다.)



```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Postdetails Page!</h1>
7     <p>{{postlist.postname}}</p>
8     <p>{{postlist.contents}}</p>
9 </body>
10 <html>
```

tutorialdjango/mysite/main/templates/main/postdetails.html

step 4 Django로 Post page 만들기(계속)

아래 url패턴처럼 blog/post번호를 입력하시면 아래 페이지처럼 출력이 됩니다.

<https://tutorialdjango-bcrpr.run.goorm.io/blog/1/>

← → C 🔒 안전함 | <https://tutorialdjango-bcrpr.run.goorm.io/blog/1/>

Postdetails Page!

Postname_1

Contents_1

- 실행화면 -

step 4 Django로 Post page 만들기(계속)

간단하게 blog화면에서 post제목과 내용을 클릭하였을 때 해당 posting으로 이동하도록 만들어 보도록 하겠습니다.

'<https://tutorialdjango-mysite/main/templates/main/blog.html>' 이 부분은 여러분의 URL로 교체해 주세요.

`tutorialdjango/mysite/main/templates/main/blog.html`

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Blog Page!</h1>
7     <table>
8         {% for list in postlist %}
9         <tr onclick="location.href='https://tutorialdjango-bcrpr.run.goorm.io/blog/{{ list.pk }}'">
10            <td>{{list.postname}}</td>
11            <td>{{list.contents}}</td>
12        </tr>
13    {% endfor %}
14 </table>
15 <body>
16 <html>
```

step 4 Django로 Post page 만들기(계속)

연결된 화면입니다. 해당 post 제목과 내용을 클릭하면 해당 포스팅으로 이동합니다.

← → ⌂ 안전함 | https://tutorialdjango-bcrpr.run.goorm.io/blog/

Blog Page!

Postname_1 Contents_1

Postname_2 Contents_2

Postname_3 Contents_3



← → ⌂ 안전함 | https://tutorialdjango-bcrpr.run.goorm.io/blog/1/

Postdetails Page!

Postname_1

Contents_1

step 4 Django로 Post page 만들기(계속)

이번에는 반대로 posting 상세 내용에서 목록으로 넘어가는 버튼을 만들어 보도록 하겠습니다.
아래 url은 적절하게 수정해주세요.

컨테이너명/mysite/main/templates/main/productdetails.html

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Postdetails Page!</h1>
7     <p>{{postlist.postname}}</p>
8     <p>{{postlist.contents}}</p>
9     <a href='https://tutorialdjango-bcrpr.run.goorm.io/blog/'>목록</a>
10 </body>
11 <html>
```

step 4 Django로 Post page 만들기(계속)

이제 사진이 들어갈 부분을 만들어 보도록 하겠습니다. tutorialdjango > mysite > main > models.py로 이동하겠습니다. 여기서 null True를 주게 되면 기존에 있는 게시판 게시물은 null 값으로 들어가게 됩니다.

tutorialdjango/mysite/main/models.py

```
1 from django.db import models
2
3 class Post(models.Model):
4     postname = models.CharField(max_length=50)
5     mainphoto = models.ImageField(blank=True, null=True)
6     contents = models.TextField()
7
8     def __str__(self):
9         return self.postname
```

step 4 Django로 Post page 만들기(계속)

위와 같이 수정하셨다면 이제 사진을 처리할 수 있는 라이브러리를 설치 해야 합니다. Pillow라는 라이브러리이며 아래와 같이 설치 할 수 있습니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install pillow==2.9.0
(myvenv)root@goorm:/workspace/컨테이너명/mysite#
python manage.py makemigrations
(myvenv)root@goorm:/workspace/컨테이너명/mysite#
python manage.py migrate
(myvenv)root@goorm:/workspace/컨테이너명/mysite#
python manage.py runserver 0:80
```

```
^C(myvenv)root@goorm:/workspace/tutorialdjango/mysite# pip install pillow==2.9.0
Downloading/unpacking pillow==2.9.0
  Downloading Pillow-2.9.0.tar.gz (9.3MB): 9.3MB downloaded
  Running setup.py (path:/workspace/tutorialdjango/mysite/myenv/build/pillow/setup.py) egg_info for package pillow

Installing collected packages: pillow
  Running setup.py install for pillow
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py makemigrations
  Migrations for 'main':
    main/migrations/0002_post_mainphoto.py
      - Add field mainphoto to post
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate
  Operations to perform:
    Apply all migrations: admin, auth, contenttypes, main, sessions
  Running migrations:
    Applying main.0002_post_mainphoto... OK
(myvenv)root@goorm:/workspace/tutorialdjango/mysite#
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py runserver 0:80
Performing system checks...

System check identified no issues (0 silenced).
May 02, 2018 - 07:38:06
Django version 2.0.4, using settings 'tutorialdjango.settings'
Starting development server at http://0:80/
Quit the server with CONTROL-C.
```

step 4 Django로 Post page 만들기(계속)

* Change post

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Main > Posts > Postname_1

Change post

HISTORY

Postname: Postname_1

Mainphoto: 파일 선택 선택된 파일 없음

Contents: Contents_1

Delete Save and add another Save and continue editing SAVE

* Add post

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Main > Posts > Add post

Add post

Postname:

Mainphoto: 파일 선택 선택된 파일 없음

Contents:

Save and add another Save and continue editing SAVE

step 4 Django로 Post page 만들기(계속)

이제 사진이 저장될 공간을 설정해주도록 하겠습니다. 이 설정을 하지 않으면 사진을 업로드 할 경우 mysite 바로 아래로 파일이 들어가게 됩니다.

tutorialdjango/mysite/tutorialdjango/settings.py

```
126     MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
127     MEDIA_URL = '/media/'
```

이미지 파일은 총 5개를 올려보도록 하겠습니다.



우선은 jeju의 포스팅을 만들어 올려보았습니다.

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Main › Posts › Add post

Add post

Postname: Postname.jeju

Mainphoto: 파일 선택 jeju.jpg

Contents: contents_jeju

Save and add another Save and continue editing **SAVE**

The screenshot shows the Django Admin 'Add post' form. The 'Postname' field contains 'Postname.jeju', the 'Mainphoto' field has a file selection button labeled '파일 선택 jeju.jpg', and the 'Contents' rich text area has the text 'contents_jeju'. The 'SAVE' button at the bottom right is highlighted with a red box. The entire form is also partially enclosed in a red box.

step 4 Django로 Post page 만들기(계속)

첫번째로 jeju의 파일을 업로드 해보았습니다. 업로드한 파일을 확인하기 위해 해당 포스팅을 클릭합니다.

Django administration

WELCOME TUTORIALDJANGO [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Main > Posts

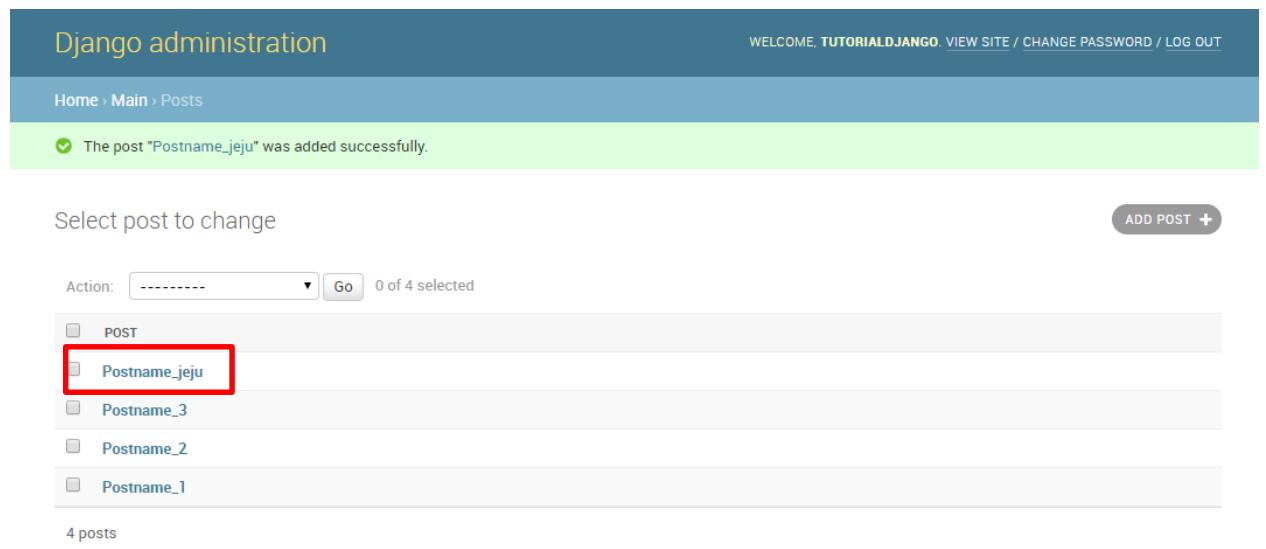
✓ The post "Postname_jeju" was added successfully.

Select post to change [ADD POST +](#)

Action: 0 of 4 selected

<input type="checkbox"/> POST
<input checked="" type="checkbox"/> Postname_jeju
<input type="checkbox"/> Postname_3
<input type="checkbox"/> Postname_2
<input type="checkbox"/> Postname_1

4 posts



step 4 Django로 Post page 만들기(계속)

클릭하니 page를 찾을 수 없다고 뜹니다. 이미지 url을 찾을 수 없기 때문에 발생되는 애러입니다. 이미지 url을 설정하려 가보도록 하겠습니다.

Django administration

WELCOME, TUTORIALDJANGO [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Main > Posts > Postname_jeju

Change post

HISTORY

Postname: Postname_jeju

Mainphoto: Currently: jeju.jpg [Clear](#)

Change: 선택된 파일 없음

Contents: contents_jeju

[Delete](#) [Save and add another](#) [Save and continue editing](#) [SAVE](#)

Page not found (404)

Request Method: GET
Request URL: <http://tutorialdjango-bcrpr.run.goorm.io/media/jeju.jpg>

Using the URLconf defined in `tutorialdjango.urls`, Django tried these URL patterns, in this order:

1. `admin/`
2. `blog/`
3. `blog/<int:pk>/`

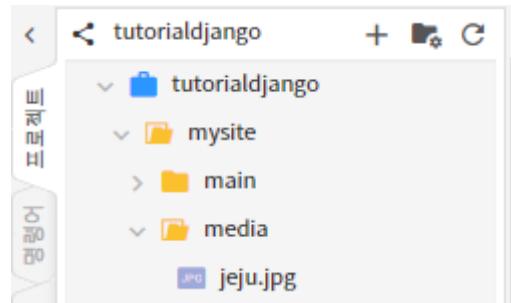
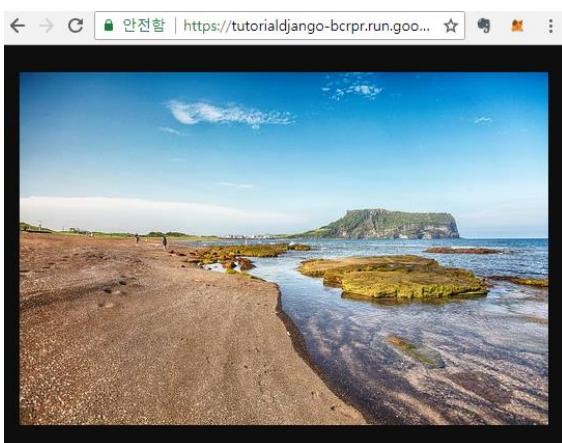
The current path, `media/jeju.jpg`, didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

step 4 Django로 Post page 만들기(계속)

아래와 같이 urlpatterns를 넣어주세요. 그러면 이미지가 정상적으로 보입니다.

```
16 from django.contrib import admin
17 from django.urls import path
18 from main.views import index, blog, postdetails
19 from django.conf.urls.static import static
20 from django.conf import settings
21
22 urlpatterns = [
23     path('admin/', admin.site.urls),
24     path('', index),
25     path('blog/', blog),
26     path('blog/<int:pk>', postdetails),
27 ]
28
29 urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```



step 4 Django로 Post page 만들기(계속)

이번에는 postdetails page에서 해당 이미지를 불러와 보도록 하겠습니다. 빨간색 네모 부분은 여러분 홈페이지 주소로 바꿔주세요. 해당 내용의 자세한 내용은 공식 홈페이지 file부분 <https://docs.djangoproject.com/en/2.0/topics/files/> 을 참고해주세요.

tutorialdjango/mysite/main/templates/main/postdetails.html

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Postdetails Page!</h1>
7     <p>{{ postlist.postname }}</p>
8     <p>{{ postlist.contents|linebreaks }}</p>
9     {% if postlist.mainphoto %}
10        
11    {% endif %}
12    <a href="https://tutorialdjango-bcrpr.run.goorm.io/blog/'>목록</a>
13 </body>
14 </html>
```

- 공식홈페이지 -

Using files in models

When you use a [FileField](#) or [ImageField](#), Django provides a set of APIs you can use to deal with that file.

Consider the following model, using an [ImageField](#) to store a photo:

```
from django.db import models

class Car(models.Model):
    name = models.CharField(max_length=255)
    price = models.DecimalField(max_digits=5, decimal_places=2)
    photo = models.ImageField(upload_to='cars')
```

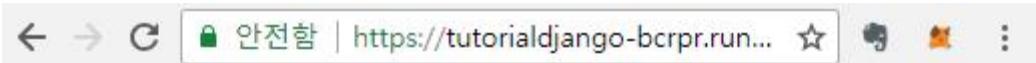
Any `Car` instance will have a `photo` attribute that you can use to get at the details of the attached photo:

```
>>> car = Car.objects.get(name="57 Chevy")
>>> car.photo
<ImageFieldfile: chevy.jpg>
>>> car.photo.name
'cars/chevy.jpg'
>>> car.photo.path
'/media/cars/chevy.jpg'
>>> car.photo.url
'http://media.example.com/cars/chevy.jpg'
```

This object – `car.photo` in the example – is a [File](#) object, which means it has all the methods and attributes described below.

step 4 Django로 Post page 만들기(계속)

이미지가 제대로 뜯 것을 확인할 수 있습니다.



Postdetails Page!

Postname_jeju

contents_jeju



[목록](#)

Step_5

댓글과 태그 가능 만들기

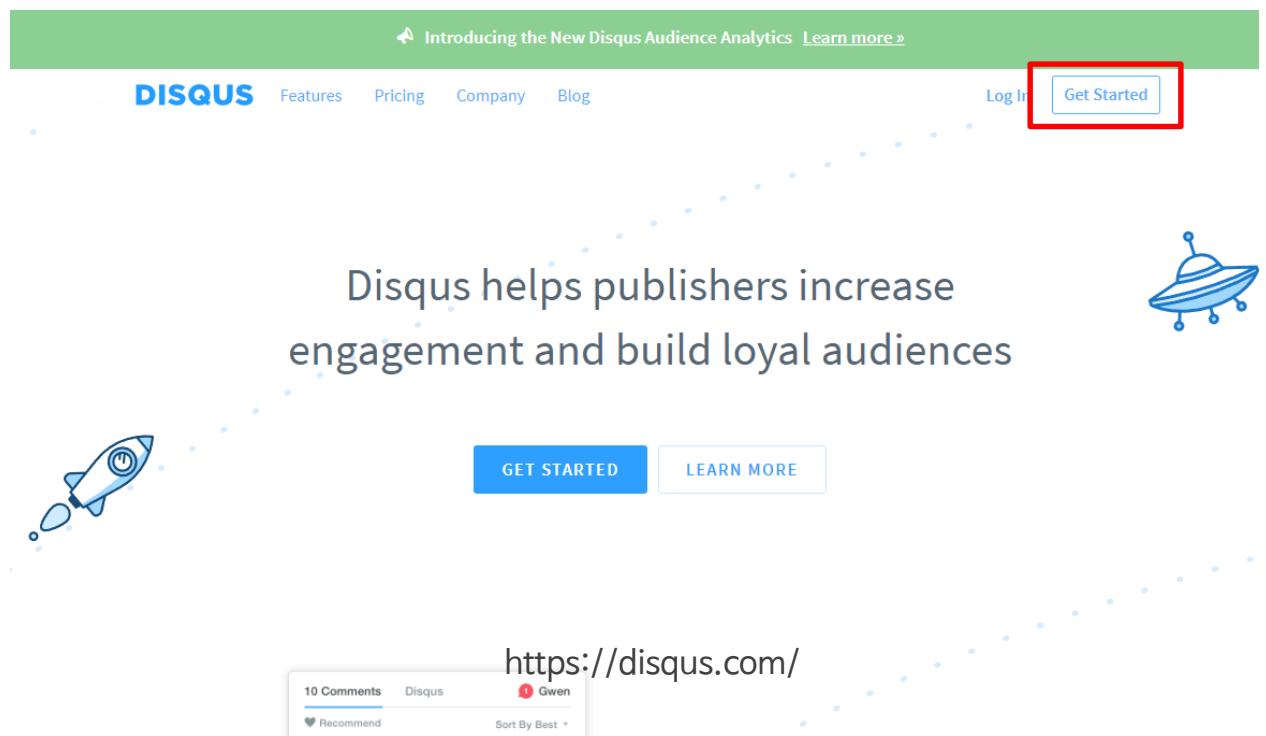
step 5 댓글 기능 만들기(계속)

이제 댓글을 달 수 있도록 설정해 보도록 하겠습니다 아래 명령어를 사용하여 disqus를 Django 내에 설치할 수도 있지만 우리는 좀 더 편한 방법을 사용해 보도록 하겠습니다.

disqus 설치하기

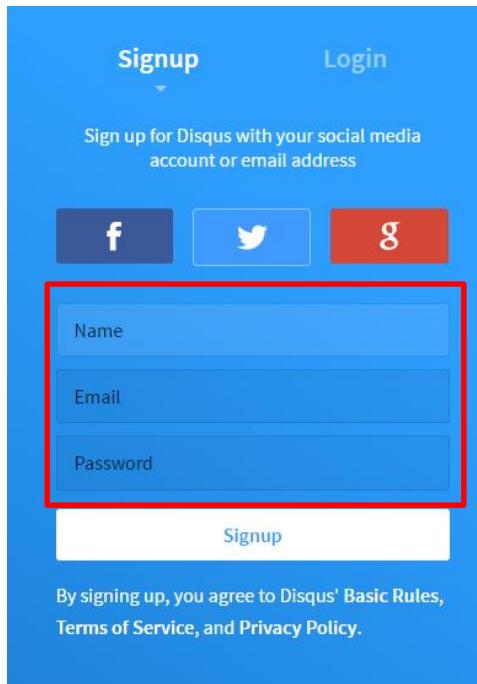
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install django-disqus

* 우리는 설치하지 않는 방법을 사용하도록 하겠습니다. 따라서 위 명령어를 사용하지는 않습니다. 그러나 설치하셔서 사용하시는 분도 계시기 때문에 명시해 두도록 하겠습니다.



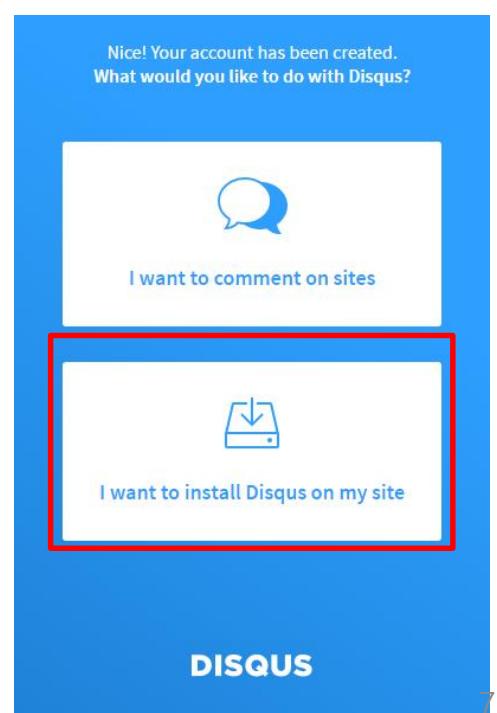
step 5 댓글 기능 만들기(계속)

이제 댓글을 달 수 있도록 설정해 보도록 하겠습니다 아래 명령어를 사용하여 disqus를 Django 내에 설치할 수도 있지만 우리는 좀 더 편한 방법을 사용해 보도록 하겠습니다.



Name, E-mail 등은 관리하기 쉽도록
구름 IDE ID, PW 사용

Disqus helps publishers increase
engagement and build loyal audiences



step 5 댓글 기능 만들기(계속)

아래 내용을 작성하신 다음(작성하는 내용이 중요하지는 않습니다.) Create Site를 눌러주세요.
우리는 BASIC으로 설치를 하도록 하겠습니다.

Create a new site
All fields are required.

Site Owner  paul lab
To associate a different account as the site owner,
[login with a different account](#)

Website Name
Your unique disqus URL will be: tutorialdjango.disqus.com
[Customize Your URL](#)

Category

Language

Create Site

Basic
Free, Ads Supported
Any number of total daily pageviews
Get all of the core Disqus features with the option to configure ads.
✓ Comments Plug-in
✓ Advanced Spam Filters
✓ Moderation Tools
✓ Basic Analytics
✓ Configurable Ads
[Learn more about Disqus' Basic features](#)

Plus
\$10 \$9 per month
Under 50,000 total daily pageviews
Get everything in Disqus Basic and the option to turn ads on and off.
✓ Everything In Basic
✓ Direct Support
✓ Ads Optional

Pro
\$99 \$89 per month
Under 150,000 total daily pageviews
Get everything in Disqus Basic, Plus, and extra premium features.
✓ Everything in Plus
✓ Priority Support
✓ Single Sign-On
✓ Advanced Analytics
✓ Shadow Banning
✓ Timeouts
✓ Email Subscriptions
[Learn more about Disqus' Pro features](#)

POPULAR

Free
For small, personal, non-commercial sites who do not run any ads. This plan includes everything in Plus except Direct Email Support.

Subscribe Now

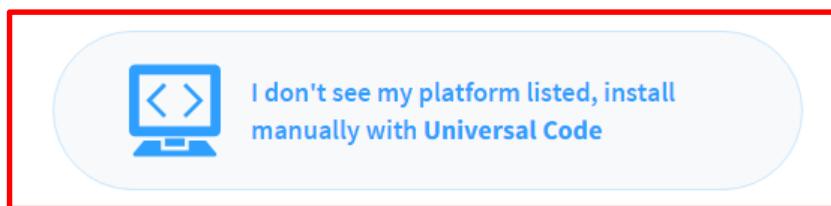
No credit card required **Start Trial (30 days)**

No credit card required **Start Trial (30 days)**

Subscribe Now

step 5 댓글 기능 만들기(계속)

다음을 누르시면 여러 플랫폼이 나오는데 맨 아래 있는 것을 클릭하겠습니다. 그런 다음 1번이나 와 있는 모든 소스를 복사해 두세요. (나중에 홈페이지에 붙여 넣을 것입니다.)



1

Place the following code where you'd like Disqus to load:

```
<div id="disqus_thread"></div>
<script>

/**
 * RECOMMENDED CONFIGURATION VARIABLES: EDIT AND UNCOMMENT THE SECTION BELOW TO INSERT
 * DYNAMIC VALUES FROM YOUR PLATFORM OR CMS.
 * LEARN WHY DEFINING THESE VARIABLES IS IMPORTANT:
 * https://disqus.com/admin/universalcode/#configuration-variables*/
 */

var disqus_config = function () {
this.page.url = PAGE_URL; // Replace PAGE_URL with your page's canonical URL variable
this.page.identifier = PAGE_IDENTIFIER; // Replace PAGE_IDENTIFIER with your page's unique identifier
variable
}
```

A screenshot of the Disqus configuration settings page. The page title is 'Configure Disqus'. It contains various input fields and dropdown menus for site settings like 'Website Name' (tutorialdjango), 'Website URL', 'Comment Policy URL', 'Comment Policy Summary' (with an example text), 'Category' (Tech), 'Description', and 'Language' (Korean). At the bottom right, there is a blue button labeled 'Complete Setup' which is highlighted with a red rectangle.

step 5 댓글 기능 만들기(계속)

작동하는지 한번 붙여보겠습니다. Postdetails.html로 오셔서 body 닫기 태그 위에 해당 소스를 붙여주세요.(주석은 보기 편하게 일부러 지웠습니다.)

tutorialdjango/mysite/main/templates/main/postdetails.html

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Postdetails Page!</h1>
7     <p>{{ postlist.postname }}</p>
8     <p>{{ postlist.contents|linebreaks }}</p>
9     {% if postlist.mainphoto %}
10        
11    {% endif %}
12        <a href='https://tutorialdjango-bcrpr.run.goorm.io/blog/'>목록</a>
13        <div id="disqus_thread"></div>
14 <script>
15 (function() {
16 var d = document, s = d.createElement('script');
17 s.src = 'https://tutorialdjango.disqus.com/embed.js';
18 s.setAttribute('data-timestamp', +new Date());
19 (d.head || d.body).appendChild(s);
20 })();
21 </script>
22 <noscript>Please enable JavaScript to view the <a href="https://disqus
23 </a></noscript>
24 </body>
<html>
```

step 5 댓글 기능 만들기(계속)

실행된 화면입니다. 목록 아래 댓글창이 달립니다.



Postname_jeju
contents_jeju

목록 댓글 0건 tutorialdjango paul lab ▾ 인기순 ▾

추천 공유 토론 시작

1등으로 댓글 달기

구독

당신의 사이트에 Disqus 추가하기

개인정보 처리방침 **DISQUS**

step 5 댓글 기능 만들기(계속)

이번에는 게시된 날짜와 수정된 날짜를 추가해 보도록 하겠습니다.

tutorialdjango/mysite/main/models.py

```
1 from django.db import models
2
3 class Post(models.Model):
4     postname = models.CharField(max_length=50)
5     mainphoto = models.ImageField(blank=True, null=True)
6     publishedDate = models.DateTimeField(blank=True, null=True)
7     modifiedDate = models.DateTimeField(blank=True, null=True)
8     contents = models.TextField()
9
10    def __str__(self):
11        return self.postname
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py makemigrations
Migrations for 'main':
  main/migrations/0003_auto_20180502_0830.py
    - Add field modifiedDate to post
    - Add field publishedDate to post
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, main, sessions
Running migrations:
  Applying main.0003_auto_20180502_0830... OK
```

step 5 댓글 기능 만들기(계속)

게시된 날짜와 수정된 날짜가 추가되어 있는 것을 확인할 수 있습니다.

Django administration

WELCOME, TUTORIALDJANGO [VIEW SITE / CHANGE PASSWORD / LOG OUT](#)

Home › Main › Posts › Postname_1

Change post HISTORY

Postname:

Mainphoto: 선택된 파일 없음

PublishedDate: Date: Today | Time: Now |
Note: You are 9 hours ahead of server time.

ModifiedDate: Date: Today | Time: Now |
Note: You are 9 hours ahead of server time.

Contents:

Delete Save and add another Save and continue editing SAVE

step 5 댓글 기능 만들기(계속)

올린 모든 게시물들을 수정해 보도록 하겠습니다.

Django administration

WELCOME, TUTORIALDJANGO. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home > Main > Posts > Postname_jeju

Change post

HISTORY

Postname: Postname_jeju

Mainphoto: Currently: jeju.jpg Change: 선택된 파일 없음

PublishedDate: Date: 2018-05-02 Time: 08:32:26

Note: You are 9 hours ahead of server time.

ModifiedDate: Date: 2018-05-02 Time: 08:32:27

Note: You are 9 hours ahead of server time.

Contents: contents_jeju

Delete Save and continue editing

step 5 댓글 기능 만들기(계속)

이번에는 blog에 수정된 날짜를 함께 올려보도록 하겠습니다.

tutorialdjango/mysite/main/templates/main/blog.html

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Blog Page!</h1>
7     <table>
8         {% for list in postlist %}
9         <tr onclick="location.href='https://t
10             <td>{{list.postname}}</td>
11             <td>{{list.contents}}</td>
12             <td>{{list.modifiedDate}}</td>
13         </tr>
14         {% endfor %}
15     </table>
16 <body>
17 <html>
```

<https://tutorialdjango-bcrpr.run.goorm.io/blog/>

Blog Page!

Postname_1	Contents_1	May 2, 2018, 8:33 a.m.
Postname_2	Contents_2	May 2, 2018, 8:33 a.m.
Postname_3	Contents_3	May 2, 2018, 8:33 a.m.
Postname_jeju	contents_jeju	May 2, 2018, 8:32 a.m.

step 5 댓글 기능 만들기(계속)

이번에는 태그를 달 수 있는 패키지은 taggit을 설치 해보도록 하겠습니다. 태그를 이용하여 검색이 가능하도록 구현할 수 있으며 태그의 별도 페이지를 만들 수도 있습니다.

```
(myvenv)root@goorm:/workspace/컨테이너명/mysite# pip install django-taggit
```

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# pip install django-taggit
Downloading/unpacking django-taggit
  Downloading django_taggit-0.22.2-py2.py3-none-any.whl (45kB): 45kB downloaded
Installing collected packages: django-taggit
  Successfully installed django-taggit
Cleaning up...
```

tutorialdjango/mysite/main/models.py

```
1 from django.db import models
2 from taggit.managers import TaggableManager
3
4 class Post(models.Model):
5     postname = models.CharField(max_length=50)
6     mainphoto = models.ImageField(blank=True, null=True)
7     publishedDate = models.DateTimeField(blank=True, null=True)
8     modifiedDate = models.DateTimeField(blank=True, null=True)
9     contents = models.TextField()
10    tag = TaggableManager(blank=True)
11
12    def __str__(self):
13        return self.postname
```

step 5 댓글 기능 만들기(계속)

App을 설치하였으니 settings에서 taggit을 추가해 주도록 하겠습니다.

```
33 INSTALLED_APPS = [  
34     'django.contrib.admin',  
35     'django.contrib.auth',  
36     'django.contrib.contenttypes',  
37     'django.contrib.sessions',  
38     'django.contrib.messages',  
39     'django.contrib.staticfiles',  
40     'main',  
41     'taggit',  
42 ]
```

tutorialdjango/mysite/tutorialdjango/settings.py

```
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py makemigrations  
Migrations for 'main':  
  main/migrations/0004_post_tag.py  
    - Add field tag to post  
(myvenv)root@goorm:/workspace/tutorialdjango/mysite# python manage.py migrate  
Operations to perform:  
  Apply all migrations: admin, auth, contenttypes, main, sessions, taggit  
Running migrations:  
  Applying taggit.0001_initial... OK  
  Applying taggit.0002_auto_20150616_2121... OK  
  Applying main.0004_post_tag... OK
```

step 5 댓글 기능 만들기(계속)

Tag가 작동하는지 확인하기 위해 admin page로 들어오면 하단에 tags라고 tag가 모여진 것을 볼 수 있습니다. 또한 각각에 페이지에 tag 입력 폼이 달린 것을 볼 수 있습니다. 우리는 이곳에 일단 여행 국가를 입력하도록 할 것입니다.

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change
Users	+ Add	Change

MAIN

Posts	+ Add	Change
-------	-------	--------

TAGGIT

Tags	+ Add	Change
------	-------	--------

Recent actions

My actions

- Postname_1 Post
- Postname_2 Post
- Postname_3 Post
- Postname_jeju Post
- Postname_jeju Post
- Postname_3 Post
- Postname_2 Post
- Post object (1) Post

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Main > Posts > Postname_jeju

Change post

HISTORY

Postname: Postname_jeju

Mainphoto: Currently: jeju.jpg [Clear] Change: 파일 선택 선택된 파일 없음

PublishedDate: Date: 2018-05-02 Today [Today] Time: 08:32:26 Now [Now]

Note: You are 9 hours ahead of server time.

ModifiedDate: Date: 2018-05-02 Today [Today] Time: 08:32:27 Now [Now]

Note: You are 9 hours ahead of server time.

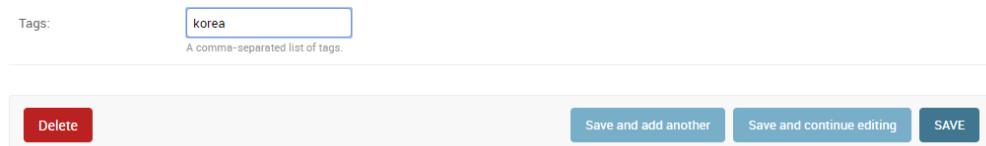
Contents: contents_jeju

Tags: A comma-separated list of tags.

Delete Save and add another Save and continue editing SAVE

step 5 댓글 기능 만들기(계속)

Tags를 입력하고 SAVE를 누른 다음 postdetails에서 출력을 해보도록 하겠습니다.



tutorialdjango/mysite/main/templates/main/postdetails.html

```
1 <html>
2 <head>
3     <title>Django!</title>
4 </head>
5 <body>
6     <h1>Postdetails Page!</h1>
7     <p>{{ postlist.postname }}</p>
8     <p>{{ postlist.contents|linebreaks }}</p>
9     {% if postlist.mainphoto %}
10        
11    {% endif %}
12    <p>{{ postlist.tag.names }}</p>
13    <br>
14    <a href='https://tutorialdjango-bcrpr.run.goorm.io/blog/'>목록</a><br>
15    <div id="disqus_thread"></div>
16 <script>
17 <(function() {
18     var d = document, s = d.createElement('script');
19     s.src = 'https://tutorialdjango.disqus.com/embed.js';
20     s.setAttribute('data-timestamp', +new Date());
21     (d.head || d.body).appendChild(s);
22 })();
23 </script>
24 <noscript>Please enable JavaScript to view the <a href="https://disqus.com
25 </a></noscript>
26 </body>
</html>
```

step 5 댓글 기능 만들기(계속)

출력은 되는데 Query Set이라는 문구가 붙습니다.

Postdetails Page!

Postname_jeju

contents_jeju



<QuerySet ['korea']>

목록

댓글 0건 tutorialdjango

1 paul lab ▾

♥ 추천 □ 공유

인기순 ▾



토론 시작

1등으로 댓글 달기

✉ 구독

엮 당신의 사이트에 Disqus 추가하기

🔒 개인정보 처리방침

DISQUS

step 5 댓글 기능 만들기(계속)

for문으로 순회를 하면 여러 개의 태그도 출력할 수 있습니다. 우선은 1개만 출력해보도록 하겠습니다. 뒤에 .0을 붙이면 가장 첫번째 것을 출력합니다.

```
<h1>Postdetails Page!</h1>
<p>{{ postlist.postname }}</p>
<p>{{ postlist.contents|linebreaks }}</p>
{% if postlist.mainphoto %}
    
{% endif %}
<p>{{ postlist.tag.names.0 }}</p>
<br>
<a href='https://tutorialdjango-bcrpr.run.goorm.io/blog/'>목록</a><br>
<div id="disqus_thread"></div>
```

Postdetails Page!

Postname_jeju

contents_jeju



korea

목록

댓글 0건 tutorialdjango

paul lab

인기순

추천

공유



토론 시작

1등으로 댓글 달기

Step_6

Bootstrap 입하기 및 배포

Tutorial 6.1

부트스트랩 소개

부트스트랩은 웹 프레임 워크입니다. 프레임 워크는 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 1년이 다 되었습니다. 그럼에도 아직 시중에는 부트스트랩 3 책만 나와있어 집필을 하게 되었습니다.

<https://www.w3schools.com>에서도 아직 부스트랩 3를 기술해 놓았습니다. 한국어 공식 홈페이지는 로는 정식 번역된 것이 없으며 <http://bootstrap4.kr/>에서 공식 홈페이지의 한국어 번역이 어느정도 되어 있는 상태(Ver 4.0)입니다. 자세한 내용은 부트스트랩 공식 홈페이지인(영문) <https://getbootstrap.com/docs/4.1/content/reboot/>에서 확인하세요.

Tutorial 6.2

부트스트랩 다운로드

<https://getbootstrap.com/>와 <http://bootstrap4.kr/>에서 다운로드 받을 수 있습니다.
<http://bootstrapk.com/>에서 다운로드를 받으면 Ver 3.3.2를 다운받게 되니 주의하세요.
* 실습은 다운로드를 받지 않고 진행하니 다운로드를 안받으셔도 됩니다.

- 영문 홈페이지 : <https://getbootstrap.com/> (Ver 4.1.0)



Bootstrap

Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

[Get started](#) [Download](#)

Currently v4.1.0



- 한글 홈페이지 : <http://bootstrap4.kr/> (Ver 4.0.0)



부트스트랩

세계에서 가장 널리 사용되는 프론트엔드 컴포넌트 라이브러리를 사용하여 반응형 웹, 모바일 우선 프로젝트를 구축하세요.

부트스트랩은 HTML, CSS, JS로 개발하기 위한 오픈소스 툴킷입니다. 아이디어를 빠르게 모델링하거나, Sass 변수와 mixin, 반응형 그리드 시스템, 그리고 jQuery 기반의 강력한 플러그인을 통해 전체 앱을 신속하게 제작할 수 있습니다.

[시작하기](#) [다운로드](#)

현재 베포버전 v4.0.0-beta



Tutorial 6.3

기본 Template 작성

기본 Template는 cdn을 사용해서 작성하였습니다. cdn은 굳이 내 폴더 내에 관련 소스를 다운로드 받지 않고도 사용할 수 있게 해줍니다.

```
1  <!doctype html>
2  <html lang="ko">
3      <head>
4          <!-- Required meta tags -->
5          <meta charset="utf-8">
6          <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7
8          <!-- Bootstrap CSS -->
9          <link rel="stylesheet"
•             href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css"
•             integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSWFphJiwGPXR1jddIhOegiu1FwO5qRGvFX0dJZ4"
•             crossorigin="anonymous">
10
11      <title>Hello, world!</title>
12  </head>
13  <body>
14      <h1>Hello, world!</h1>
15
16      <!-- Optional JavaScript -->
17      <!-- jQuery first, then Popper.js, then Bootstrap JS -->
18      <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
•             integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo"
•             crossorigin="anonymous"></script>
19      <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.0/umd/popper.min.js"
•             integrity="sha384-cs/chFZiN24E4KMATLdqdvsezGxaGsi4hLG0z1Xwp5UZB1LY//20VyM2taTB4QvJ"
•             crossorigin="anonymous"></script>
20      <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"
•             integrity="sha384-uefMccjFJAIV6A+rW+L4AHF99KvxDjWSu1z9VI8SKNVmz4sk7buKt/6v9KI65qnm"
•             crossorigin="anonymous"></script>
21  </body>
22 </html>
```

001.html

최신 버전의 cdn을 사용하는 방법은 구글에서 bootstrap cdn을 검색하신 후 최상단에 노출되어 있는 <https://www.bootstrapcdn.com/> 페이지로 들어가시면 cdn을 사용할 수 있습니다. 현재 버전은 4.1.1 입니다.

A screenshot of a Google search results page. The search query is "bootstrap cdn". The top result is a link to "Quick Start · BootstrapCDN by StackPath" with the URL <https://www.bootstrapcdn.com/>. Below the link, there is a snippet of text: "The recommended CDN for Bootstrap, Font Awesome and Bootswatch. Bootstrap · Font Awesome · Showcase · Bootswatch". It also shows the last visit date: "이 페이지를 여러 번 방문했습니다. 최근 방문 날짜: 18. 4. 29". To the right of the snippet, there is a sidebar titled "함께 검색한 항목" with links like "bootstrap cdn 이란", "bootstrap table 예제", "maxcdn jquery", "vue js cdn", "maxcdn bootstrap 4", and "bootstrap3".

A screenshot of the "Quick Start" page for Bootstrap v4.1.1. It displays three main sections: "Complete CSS", "Complete JavaScript", and "Complete JavaScript Bundle". Each section contains a URL and a "Click to copy" button, which is highlighted with a red box.

- Complete CSS**
URL: <https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css>
- Complete JavaScript**
URL: <https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js>
- Complete JavaScript Bundle**
URL: <https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.bundle.min.js>

Tutorial 6.4

기본 Template 실습

앞으로 실습 캡쳐는 <body>와 </body> 안에 있는 태그들 중 <script>…</script> 를 생략하고 작성하도록 하겠습니다. 아래 빨간색 네모를 그린 곳에 코드가 들어가게 됩니다.

```
1  <!doctype html>
2  <html lang="ko">
3      <head>
4          <!-- Required meta tags -->
5          <meta charset="utf-8">
6          <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7
8          <!-- Bootstrap CSS -->
9          <link rel="stylesheet"
•             href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css"
•             integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSWFphJiwGPXR1jddIhOegiu1Fw05qRGvFX0dJZ4"
•             crossorigin="anonymous">
10
11      <title>Hello, world!</title>
12  </head>
13  <body>
14      <h1>Hello, world!</h1>
15
16      <!-- Optional JavaScript -->
17      <!-- jQuery first, then Popper.js, then Bootstrap JS -->
18      <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
•             integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo"
•             crossorigin="anonymous"></script>
19      <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.0/umd/popper.min.js"
•             integrity="sha384-cs/chFZiN24E4KMATLdqdvsezGxaGsi4hLG0z1Xwp5UZB1LY//20VyM2taTB4QvJ"
•             crossorigin="anonymous"></script>
20      <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"
•             integrity="sha384-uefMccjFJAIV6A+rW+L4AHF99KvxDjWSu1z9VI8SKNVmz4sk7buKt/6v9KI65qnm"
•             crossorigin="anonymous"></script>
21  </body>
22 </html>
```

001.html

6.4 기본 Template 실습(계속)

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.

<https://getbootstrap.com/docs/4.1/examples/grid/>

6.4 기본 Template 실습(계속)

앞서 말씀드린 것처럼 실습 캡쳐는 <body>와 </body> 안에 있는 태그들 중 <script>…</script> 를 생략하고 작성하도록 하겠습니다. Code와 같이 작성한다음 002.html로 저장을 하고 실행하면 아래와 같이 나옵니다.
빨간색 네모는 이해를 돋기 위해 작성한 것입니다. 1개의 row에 4 컬럼씩 할당했으므로 균등한 넓이로 출력됩니다.

Code

```
14      <div class="container">
15          <div class="row">
16              <div class="col-md-4">
17                  <h1>hello</h1>
18              </div>
19
20              <div class="col-md-4">
21                  <h1>hello</h1>
22              </div>
23
24              <div class="col-md-4">
25                  <h1>hello</h1>
26              </div>
27          </div>
28      </div>
```

Output

hello

4컬럼

hello

4컬럼

hello

4컬럼

002.html

6.4 기본 Template 실습(계속)

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

Code

```
14      <div class="container">          26      <div class="row">
15          <div class="row">            27          <div class="col-md-6">
16              <div class="col-md-4">        28              <h1>hello</h1>
17                  <h1>hello</h1>    29          </div>
18              </div>            30          <div class="col-md-6">
19              <div class="col-md-4">    31              <h1>hello</h1>
20                  <h1>hello</h1>  32          </div>
21              </div>            33          </div>
22              <div class="col-md-4">    34      </div>
23                  <h1>hello</h1>
24              </div>
25      </div>
```

Output

The output shows four rows of text 'hello'. The first row contains three boxes, each with a red border. The second row contains two boxes, each with a red border. This visualizes how the CSS grid layout divides the available space into columns of different widths (4, 6, 4, 6).

003.html

6.4 기본 Template 실습(계속)

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

<https://getbootstrap.com/docs/4.1/examples/grid/>

6.4 기본 Template 실습(계속)

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

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

공식 홈페이지에 나와있는 그리드 시스템에 대한 설명입니다. Nestable은 컬럼을 또다른 컬럼으로 나눌 수 있다는 얘기입니다.

우리는 주로 md를 사용합니다.

	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				

<https://getbootstrap.com/docs/4.1/layout/grid/>

6.4 기본 Template 실습(계속)

offset을 주면 1컬럼을 띄게 됩니다. 여백을 주는 방법은 여러가지가 있지만 주로 offset을 이용한 방법과 Flexbox를 이용하여 컬럼 컬럼을 정렬하는 방법을 사용합니다. 여기서는 offset만 다루고 있습니다.

Code

```
14      <div class="container">
15        <div class="row">
16          <div class="col-md-5 offset-md-1">
17            <h1>hello</h1>
18          </div>
19          <div class="col-md-5">
20            <h1>hello</h1>
21          </div>
22        </div>
```

```
23      <div class="row">
24        <div class="col-md-6">
25          <h1>hello</h1>
26        </div>
27        <div class="col-md-6">
28          <h1>hello</h1>
29        </div>
30      </div>
31    </div>
```

Output

hello
hello

hello
hello

003.html

Tutorial 6.5

Django에 Template 입히기

압축파일에 보시면 index.html, blog.html, postdetails.html 파일이 있을 것입니다. 여기서 index.html을 실행해 보시면 지도가 보입니다. 여기서 지도에 마커를 찍기 위해서는 각 post마다 좌표가 필요합니다. 이 좌표는 필수로 찍도록 하겠습니다. 여기서 LatLng는 위도와 경도입니다.

뒤에서는 템플릿 태그를 사용하여 네모 박스 친 곳에 각각에 게시물들의 값이 들어가게 할 것입니다.

```
285     function initMap() {  
286  
287         var bluemountain = new google.maps.LatLng(-33.3493206, 149.7360613);  
288         var jejucityhall = new google.maps.LatLng(33.499597, 126.5290653);  
289         var perth = new google.maps.LatLng(-32.0388312, 115.4010747);  
290         var sydney = new google.maps.LatLng(-33.8567844, 151.213108);  
291         var tasmania = new google.maps.LatLng(-42.200633, 146.643736);
```

Index_001.html

* 모든 templates 소스는 paullab.co.kr/templates.zip에 있습니다.

6.5 Django에 Template 입하기(계속)

공식문서 :

<https://docs.djangoproject.com/en/2.0/ref/models/fields/#django.db.models.FloatField>

공식문서에 보시면 blank와 null에 관련된 내용이 있습니다.

```
1 from django.db import models
2 from taggit.managers import TaggableManager
3
4 class Post(models.Model):
5     postname = models.CharField(max_length=50)
6     Lat = models.FloatField(null=True)
7     Lng = models.FloatField(null=True)
8     mainphoto = models.ImageField(blank=True, null=True)
9     publishedDate = models.DateTimeField(blank=True, null=True)
10    modifiedDate = models.DateTimeField(blank=True, null=True)
11    contents = models.TextField()
12    tag = TaggableManager(blank=True)
13
14    def __str__(self):
15        return self.postname
```

tutorialdjango/mysite/main/models.py

null

If **True**, Django will store empty values as **NULL** in the database. Default is **False**.

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.

공식문서에 null과 blank에 대한 내용

6.5 Django에 Template 입하기(계속)

```
python manage.py makemigrations
```

```
python manage.py migrate
```

명령어를 사용하여 DB에 반영합니다. DB에 반영이 제대로 되었으면 admin page로 이동하여 입력이 가능한지 확인합니다. 실행명령어인

```
python manage.py runserver 0:80
```

을 실행해 주세요.

```
(myenv)root@goorm:/workspace/tutordjango/mysite# python manage.py makemigrations
Migrations for 'main':
 main/migrations/0005_auto_20180504_0204.py
 - Add field Lat to post
 - Add field Lng to post
(myenv)root@goorm:/workspace/tutordjango/mysite# python manage.py migrate
Operations to perform:
 Apply all migrations: admin, auth, contenttypes, main, sessions, taggit
Running migrations:
 Applying main.0005_auto_20180504_0204... OK
```

- 캡쳐 화면 -

6.5 Django에 Template 입히기(계속)

캡쳐된 화면을 보면 제대로 반영되었다는 것을 알 수 있습니다. 만약 아이디와 비밀번호를 잊어버리셨다면 콘솔창에서 다시 만드시면 됩니다.

Django administration

WELCOME, TUTORIALDJANGO. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Home › Main › Posts › Postname_jeju

Change post

HISTORY

Postname: Postname_jeju

Lat:

Lng:

Mainphoto: Currently: jeju.jpg [Clear](#)
Change: [파일 선택](#) 선택된 파일 없음

PublishedDate: Date: 2018-05-02 [Today](#) | [Calendar](#)
Time: 08:32:26 [Now](#) | [Clock](#)

Note: You are 9 hours ahead of server time.

ModifiedDate: Date: 2018-05-02 [Today](#) | [Calendar](#)
Time: 08:32:27 [Now](#) | [Clock](#)

Note: You are 9 hours ahead of server time.

Contents: contents_jeju

Tags: korea
A comma-separated list of tags.

[Delete](#) [Save and add another](#) [Save and continue editing](#) [SAVE](#)

6.5 Django에 Template 입하기(계속)

여기서 5개의 값을 입력하도록 하겠습니다. 5개의 값은

bluemountain, 위도 -33.3493206, 경도 149.7360613

jejucityhall, 위도 33.499597, 경도 126.5290653

perth, 위도 -32.0388312, 경도 115.4010747

sydney, 위도 -33.8567844, 경도 151.213108

tasmania, 위도 -42.200633, 경도 146.643736

입니다. 내용은 로렘입술값을 넣겠습니다. 사진은 img 폴더에 있는 각각의 사진으로 저장하도록 하겠습니다.

* 5개를 제외한 게시물은 모두 지우겠습니다.

Django administration

WELCOME, TUTORIALDJANGO. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Main › Posts

The post "tasmania" was added successfully.

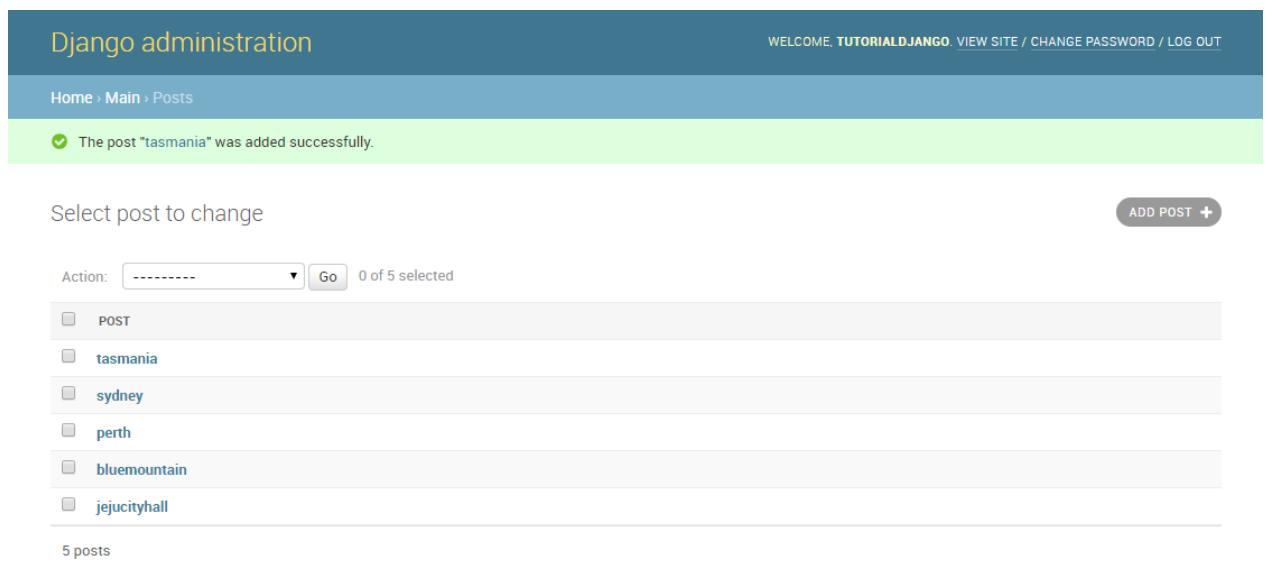
Select post to change

Action: ----- Go 0 of 5 selected

POST
 tasmania
 sydney
 perth
 bluemountain
 jejucityhall

5 posts

ADD POST +



6.5 Django에 Template 입하기(계속)

이제 해당 내용이 index.html에 반영이 되는지 확인해 보도록 하겠습니다. 우선 전체 postlist를 index에서 사용할 수 있도록 아래와 같이 수정해 줍니다. 서비스를 중지한 다음 다시 켜시면 아래와 같이 이미지가 깨진채로 마킹이 되는 것을 볼 수 있습니다. 이제 본격적으로 index.html 파일을 수정해 보도록 하겠습니다.

```
4 def index(request):
5     postlist = Post.objects.all()
6     return render(request, 'main/index.html', {'postlist':postlist})
```

tutorialdjango/mysite/main/views.py

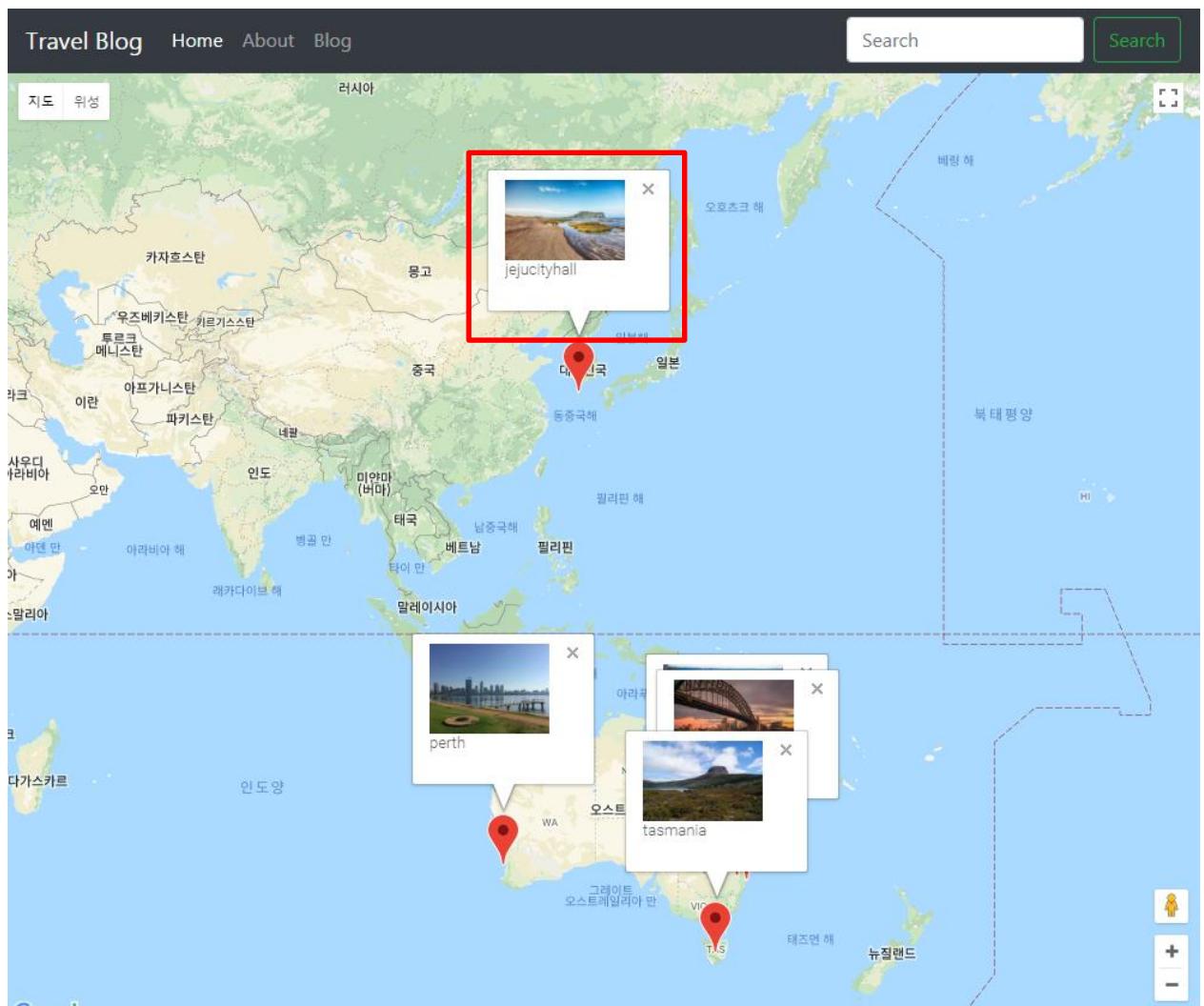
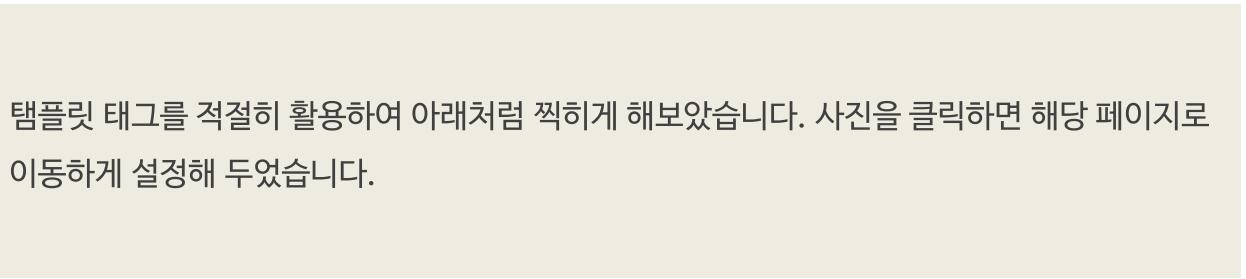
```
285     function initMap() {
286         {% for list in postlist %}
287             var {{list.postname}} = new google.maps.LatLng({{list.Lat}},{{list.Lng}});
288         {% endfor %}
```

Index_002.html



해당 화면이 보이지 않으신다면 서버를 내렸다가 다시 올려주세요.

6.5 Django에 Template 입히기(계속)



6.5 Django에 Template 입히기(계속)

템플릿 태그를 적절히 활용하여 아래처럼 찍히게 해보았습니다. 사진을 클릭하면 해당 페이지로 이동하게 설정해 두었습니다.

아래 내용이 수정한 내용입니다. 반복되는 내용들은 삭제하였습니다.

```
{% for list in postlist %}  
var {{list.postname}}marker = new google.maps.Marker({  
    position: {{list.postname}},  
    map: map  
});  
var infowindow = new google.maps.InfoWindow({  
    content: "<a href='https://tutorialdjango-bcrpr.run.goorm.io/blog/{{  
        list.pk }}'><img width='100px;' src='{{ list.mainphoto.url }}'></a><br><p>{{list.postname}}</p>"  
});  
infowindow.open(map,{{list.postname}}marker);  
{% endfor %}
```

Index_003.html

6.5 Django에 Template 입하기(계속)

이제 blog.html을 수정하도록 하겠습니다. 162번째 줄 아래 내용이 핵심 내용입니다. For문으로 postlist를 순회하면서 각 값을 불러오고 있습니다.

```
{% for list in postlist %}  
<div class="col-md-4">  
  <div class="card mb-4 box-shadow">  
    {% if list.mainphoto %}  
      <a href="https://tutoraldjango-bcrpr.run.goorm.io/blog/{{ list.pk }}/"></a>  
    {% endif %}  
    <div class="card-body">  
      <p class="card-text">{{list.postname}}</p>  
      <div class="d-flex justify-content-between align-items-center">  
        <div class="btn-group">  
          <button type="button" class="btn btn-sm btn-outline-secondary">View</button>  
          <button type="button" class="btn btn-sm btn-outline-secondary">Edit</button>  
        </div>  
        <small class="text-muted">9 mins</small>  
      </div>  
    </div>  
  </div>  
</div>  
{% endfor %}
```

blog.html

6.5 Django에 Template 입히기(계속)

실행해 보았습니다. 여기서 각 사진을 클릭하면 해당 포스팅으로 이동합니다.

Travel Blog Home About **Blog** Search **Search**



jejucityhall

[View](#) [Edit](#) 9 mins



bluemountain

[View](#) [Edit](#) 9 mins



perth

[View](#) [Edit](#) 9 mins



sydney

[View](#) [Edit](#) 9 mins



tasmania

[View](#) [Edit](#) 9 mins

- 캡쳐화면 -

6.5 Django에 Template 입하기(계속)

이제 postdetails.html을 수정하도록 하겠습니다. 앞에서 만들었던 postdetails.html에서 가져온 템플릿 태그들을 147번째 줄부터 165줄까지 적절한 곳에 배치하였습니다.

```
<div class="blog-post">
  <h2 class="blog-post-title">{{ postlist.postname }}</h2>
  <p class="blog-post-meta">수정된 날짜 : {{ postlist.modifiedDate }}</p>
  <p class="blog-post-meta">올려진 날짜 : {{ postlist.publishedDate }}</p>
  {% if postlist.mainphoto %}
    
  {% endif %}

  <p>{{ postlist.contents|linebreaks }}</p>
  <h2>tag</h2>
  <p>{{ postlist.tag.names.0 }}</p>

  <div id="disqus_thread"></div>

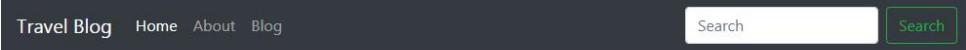
</div><!-- /.blog-post -->

<nav class="blog-pagination">
  <a class="btn btn-outline-primary"
     href="https://tutorialdjango-bcrpr.run.goorm.io/blog/">List</a>
</nav>
```

postdetails.html

6.5 Django에 Template 입하기(계속)

실행화면입니다.



From the Firehose

jejucityhall

수정된 날짜 : May 2, 2018, 8:32 a.m.

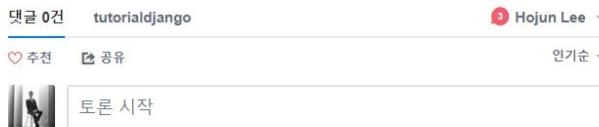
올려진 날짜 : May 2, 2018, 8:32 a.m.



*Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure
dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.*

tag

korea

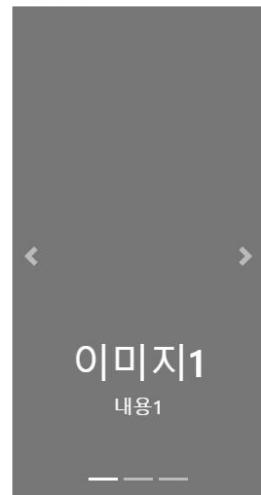


1등으로 댓글 달기

About

*Etiam porta sem malesuada magna
mollis euismod. Cras mattis
consectetur purus sit amet
fermentum. Aenean lacinia
bibendum nulla sed consectetur.*

O/미지/ 캐러셀 1



이미지1

내용1



vanillavalue.com의 응답을 기다리는 중...

blog.html

6.5 Django에 Template 입히기(계속)

이 튜토리얼에서는 템플릿 상속을 넣지 않았습니다. 하지만 템플릿 상속은 서비스를 만드신다면 꼭 넣으셔야 하는 내용이므로 언급하고 넘어갑니다.

메뉴와 푸터 같이 모든 페이지에 들어가는 템플릿은 레이아웃 템플릿을 따로 만들어 확장하여 템플릿을 작성합니다.

만약 메뉴가 추가되어 수정된다면 우리는 3페이지 밖에 안되기 때문에 3번만 수정하면 되지만 100페이지가 넘는 템플릿이 있다면 일일이 수정하기 어려울 것입니다. 다음은 공식문서에 나온 템플릿 상속 예제입니다.

year_archive.html이 base.html을 상속받아 사용하고 있습니다.

```
mysite/templates/base.html
  {% load static %}
<html>
<head>
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
    
    {% block content %}{% endblock %}
</body>
</html>

mysite/news/templates/news/year_archive.html
  {% extends "base.html" %}

  {% block title %}Articles for {{ year }}{% endblock %}

  {% block content %}
<h1>Articles for {{ year }}</h1>

  {% for article in article_list %}
    <p>{{ article.headline }}</p>
    <p>By {{ article.reporter.full_name }}</p>
    <p>Published {{ article.pub_date|date:"F j, Y" }}</p>
  {% endfor %}
  {% endblock %}
```

Tutorial 6.6

서비스 배포

구름 IDE에서는 항상 켜두기 기능을 프리미엄 계정에서만 제공되고 있습니다. 클릭만 하시면 항상 켜두기가 활성화 됩니다. 더보기를 클릭하셔서 다음 장으로 넘어가세요.



6.6 서비스 배포(계속)

실행 URL과 포트에서 추가를 누르시고 구매하신 URL을 입력합니다. 자동실행 스크립트는 Django가 실행되기 위한 최소한의 스크립트를 적어놓는 것이 좋습니다. 저는 주로을 넣어놓고 실행해 놓습니다.

```
source /workspace/컨테이너이름/mysite/myvenv/bin/activate  
python3 /workspace /컨테이너이름/mysite/manage.py runserver 0:80
```

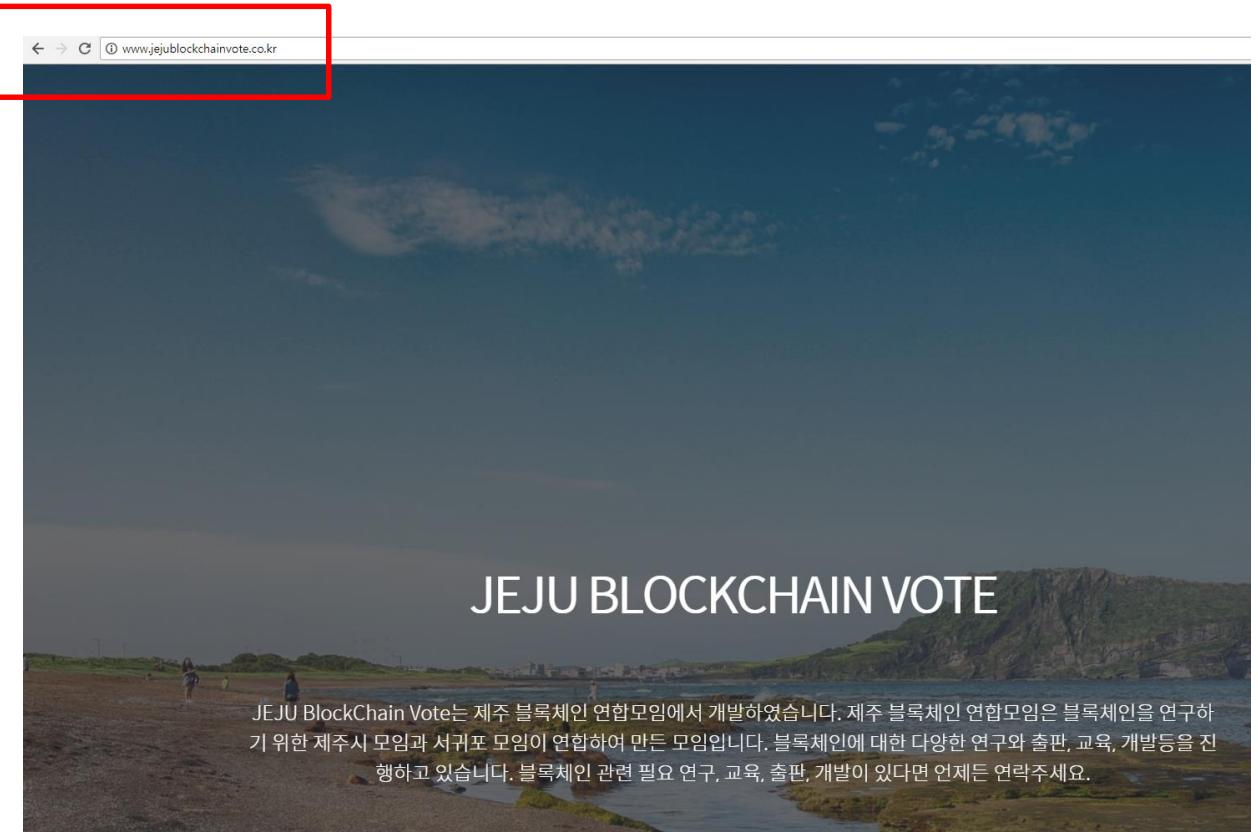
컨테이너 설정

항상 켜두기	<input checked="" type="checkbox"/> ON 사용자 VM이 종료되지 않고 항상 켜져있습니다.
자동실행 스크립트	삭제 설정 1 Script goes here...
컨테이너 정지	컨테이너 정지 실행중인 'jejublockvote' 컨테이너가 모두 정지됩니다.
컨테이너 삭제	컨테이너 삭제 이 작업은 되돌릴 수 없습니다

실행 URL과 포트		+ 추가	편집
URL	포트		
jejublockvote.run.goorm.io	3001		
www.jejublockchainvote.co.kr	3000		
jejublockvote-vrlkh.run.goorm.io	80		

6.6 서비스 배포(계속)

URL이 제대로 연결되고 작동하는지 확인해보세요.



JEJU BlockChain Vote는 제주 블록체인 연합모임에서 개발하였습니다. 제주 블록체인 연합모임은 블록체인을 연구하기 위한 제주시 모임과 서귀포 모임이 연합하여 만든 모임입니다. 블록체인에 대한 다양한 연구와 출판, 교육, 개발 등을 진행하고 있습니다. 블록체인 관련 필요 연구, 교육, 출판, 개발이 있다면 언제든 연락주세요.

Step_7

요약정리

Django

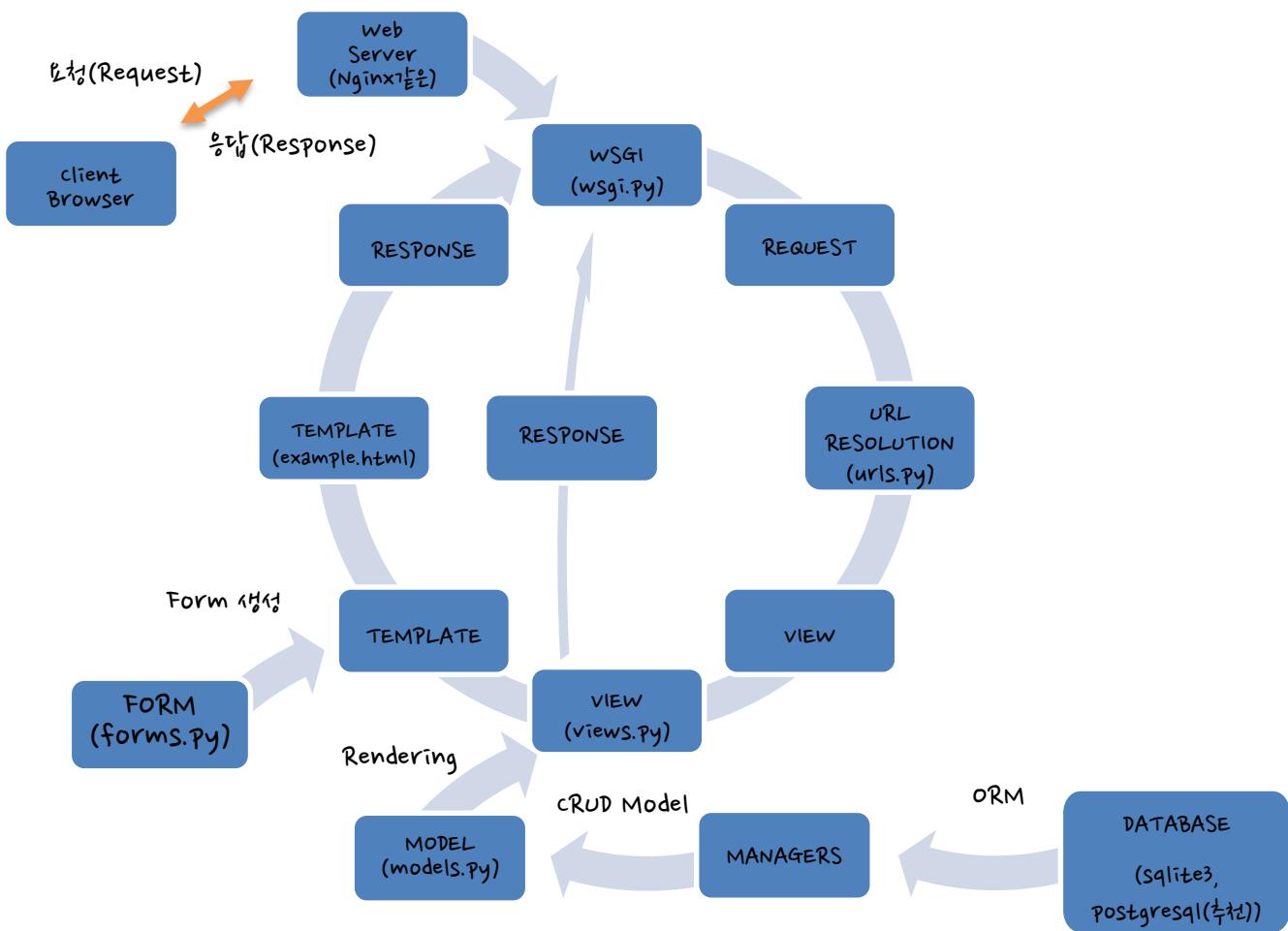
Django는 Python으로 빠르게 웹을 개발하기 위해 나온 Web Framework입니다. 대부분의 기능들이 자동화 되어 있으며 공식문서도 잘 되어 있어 공식문서 정독을 한번 하면 다른 문서는 필요 없을 정도로 상세 내용을 가지고 있습니다.

우리는 지금까지 간단한 여행 블로그를 만들어 보았습니다. 그러나 Django에 극히 일부분만을 다룬 것입니다. 더 강력한 Django의 기능을 사용하고 싶으시다면 지금 바로 공식문서를 정독하시길 권해드립니다.

아래는 Django의 기능을 전체 도식화 정리 해놓은 것입니다. 보시면서 놓치신 것이 없으신지 체크해보시기 바랍니다.

* 해당 책에서는 form에 대해 다루지 않았습니다.

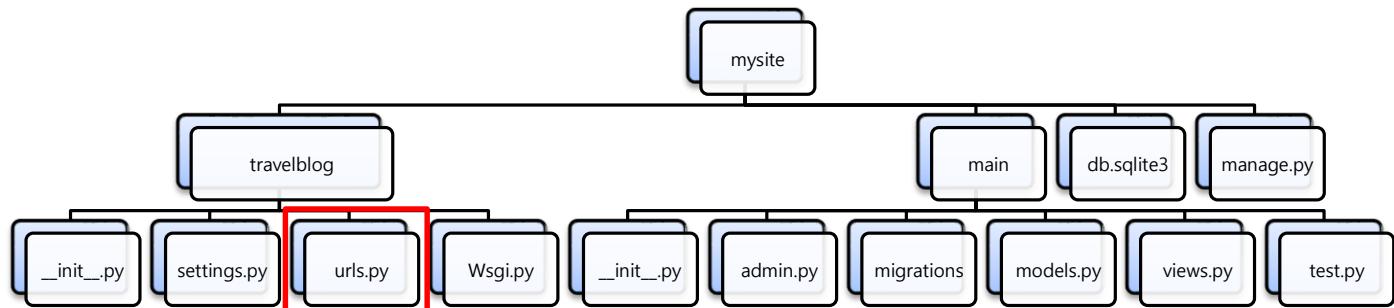
Django



URL 설계

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, blog, postdetails
from django.conf.urls.static import static
from django.conf import settings

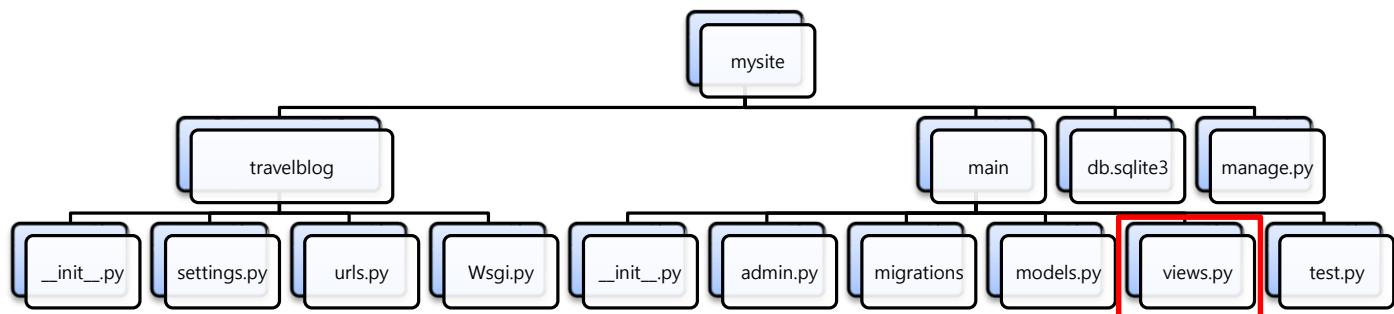
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index),
    path('blog/', blog),
    path('blog/<int:pk>', postdetails),
]

urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

View -로직 설계

URL 설계가 끝났다면 URL에서 연결해준 각종 함수를 views에서 만들어 줍니다.

views.py



```
from django.shortcuts import render
from .models import Post

def index(request):
    postlist = Post.objects.all()
    return render(request, 'main/index.html', {'postlist':postlist})

def blog(request):
    postlist = Post.objects.all()
    return render(request, 'main/blog.html', {'postlist':postlist})

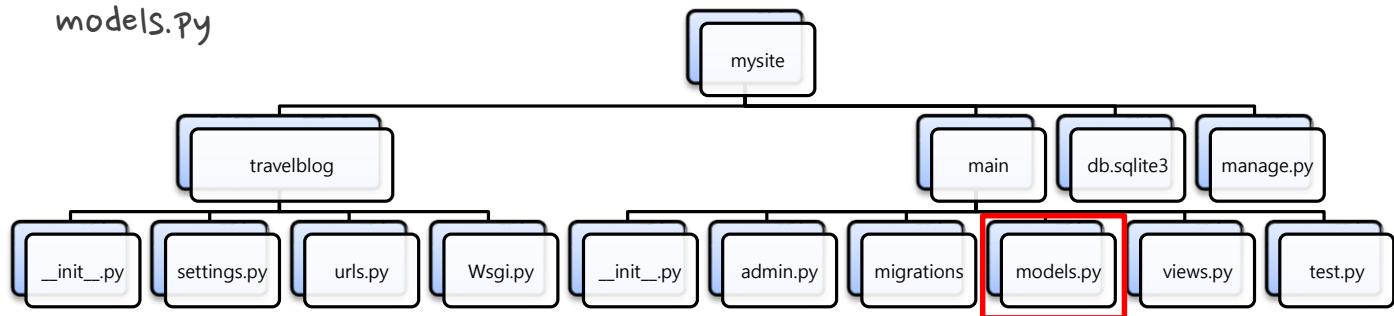
def postdetails(request, pk):
    postlist = Post.objects.get(pk=pk)
    return render(request, 'main/postdetails.html', {'postlist':postlist})
```

Model – 데이터 베이스 설계

아래 코드는 후에 작성하였지만 설계를 제대로 하셨다면 세번째에서 작업할 사항입니다. 각종 게시물 등 필요한 model을 설계합니다.

전부 작성한 후에는 makemigrations와 migrate로 DB에 반영합니다.

models.py



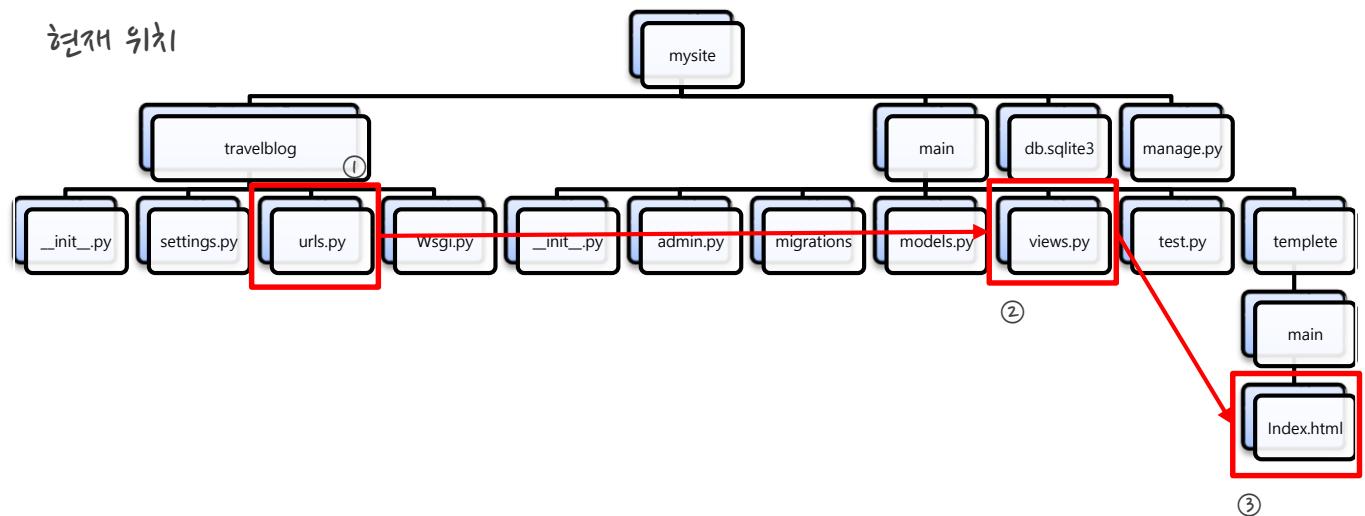
```
from django.db import models
from taggit.managers import TaggableManager

class Post(models.Model):
    postname = models.CharField(max_length=50)
    Lat = models.FloatField(null=True)
    Lng = models.FloatField(null=True)
    mainphoto = models.ImageField(blank=True, null=True)
    publishedDate = models.DateTimeField(blank=True, null=True)
    modifiedDate = models.DateTimeField(blank=True, null=True)
    contents = models.TextField()
    tag = TaggableManager(blank=True)

    def __str__(self):
        return self.postname
```

Template – 화면 UI 설계

이번에는 Template을 작성 해보겠습니다. 중간에 모델 설계를 하지 않았다면 3번째 작업할 사항입니다. 우리는 templates라는 폴더 아래 main이라는 폴더를 만들어서 그 안에 각종 템플릿을 작성하였습니다.



여행블로그 튜토리얼로 배우는 Django 2.0 with 구름IDE and Bootstrap 4.1.0

초판 1쇄 발행 | 2017년 5월 4일

지은이 | 이호준, 최수원

편집 | 이호준

총괄 | 이호준

펴낸곳 | 사도출판

주소 | 제주특별자치도 제주시 사라봉 9길 18, 102호

표지디자인 | 사도출판

홈페이지 | <http://www.paullab.co.kr>

E-mail | paul-lab@naver.com

ISBN | 979-11-88786-05-3

Copy right © 2017 by. 사도출판

이 책의 저작권은 사도출판에 있습니다.

저작권법에 의해 보호를 받는 저작물이므로 무단 복제 및 무단 전재를 금합니다.

이 책에 대한 의견을 주시거나 오탈자 및 잘못된 내용의 수정 정보는 사도출판의
이메일로 연락을 주시기 바랍니다.

여행블로그 튜토리얼로 배우는 Django 2.0 with 구름IDE and Bootstrap 4.1.0

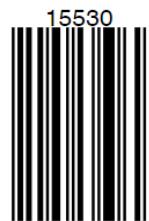
대상 독자

이 책은 Django로 무언가를 만들어보고 싶으신 분들에게 추천해드립니다. Django는 Python 웹 프레임 워크 중 가장 사랑받는 프레임 워크입니다. Django로 간단한 블로그를 만들고 배포해본 경험으로 자신이 원하는 사이트를 구축하는 발판이 되었으면 합니다.

이 책의 내용

이 책은 Django로 간단한 여행블로그를 만들면서 Django에 대해 익히게 되어있는 튜토리얼 책입니다. 따라서 Django에 대해 심도깊게 다루기 보다는 서비스를 만들기 위한 일련의 과정을 빠르게 한 사이클 도는 것에 중점을 두었습니다. 따라만 하면 서비스를 만들 수 있도록 하였으며 각각에 코드에 대해 파일로 만들어 두었으니 참고하셔서 작업하시면 좋습니다.

값 5,000원



9 791188 786053
ISBN 979-11-88786-05-3 (PDF)