알찬대학 따뜻한동행

2024

응용소프트웨어개발 8주차 Django 시작하기

JEONBUK NATIONAL UNIVERSITY

전북대학교
JEONBUK NATIONAL UNIVERSITY

웹 사이트의 작동구조

그림 5-1 HTML, CSS, 자바스크립트로만 구성된 웹 사이트

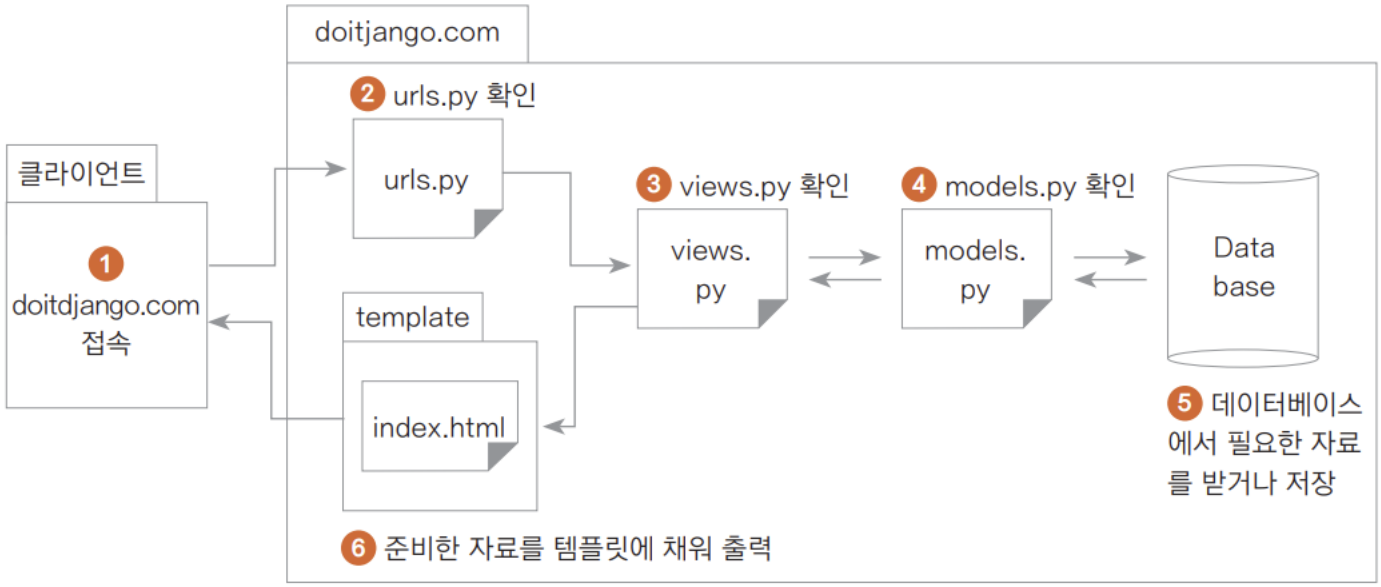


그림 5-3 장고로 만든 웹 사이트의 작동 구조

MTV 패턴

- 앞에서 알아봤듯이 장고로 만든 웹 사이트는 모델model로 자료의 형태를 정의하고, 뷰view로 어떤 자료를 어떤 동작으로 보여줄지 정의하고, 템플릿template으로 웹 페이지에서 출력할 모습을 정의합니다. 이러한 작동 구조를 줄여서 MTV 패턴이라고 부릅니다. 이렇게 분리해서 웹 사이트 기능을 관리함으로써 프런트엔드 개발자는 HTML을 비롯한 화면 구성에 집중할 수 있게 되고, 백엔드 개발자도 화면 뒤의 작업에 집중할 수 있게 되죠. 혼자 개발할 때도 마찬가지입니다. 장고의 MTV 패턴을 따라 개발한다면 백엔드 로직과 프런트엔드 디자인이 뒤죽박죽인 코드가 되어 어디서 무엇을 수정해야 할지 모르는 사태를 방지할 수 있습니다.

Django 프로젝트 시작하기

- 저장소 새로 만들기
- 가상환경 설정하기
- 장고 설치하기
- 장고 프로젝트 생성하기
- 장고 첫 페이지 띄우기
- 관리페이지 만들고 접속하기

장고 프로젝트 생성하기

01단계 장고 프로젝트 생성하기

이제 장고 프로젝트를 생성해 봅시다. 터미널에서 `django-admin startproject do_it_django_prj .`라고 입력합니다. 이때 맨 마지막의 닷dot(.)을 잊지 마세요. 그래야 필자와 똑같은 구조로 파일이 생성됩니다. 여기에서 닷의 의미는 '이 폴더에 장고 프로젝트를 만들자'는 의미입니다.

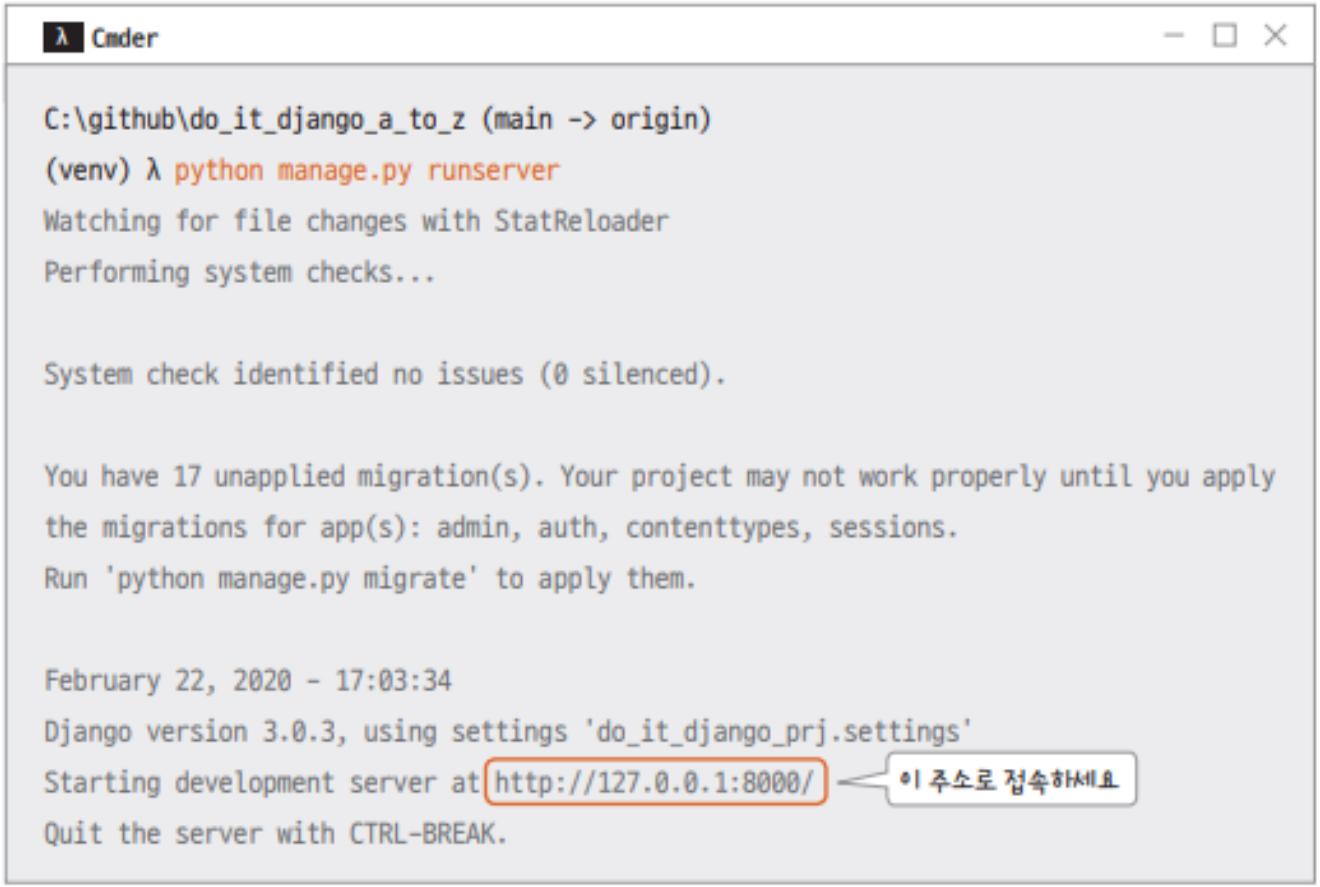
```
λ Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ django-admin startproject do_it_django_prj .
```

꼭 닷(.)을 입력하세요

이제 `ls` 명령어로 어떤 파일과 폴더가 생겼는지 확인합시다. do_it_django_prj 폴더와 manage.py가 새로 생성되었습니다.

```
λ Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ ls
do_it_django_prj/  manage.py*  README.md  venv/
```

장고 프로젝트 실행하기


λ Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
You have 17 unapplied migration(s). Your project may not work properly until you apply
the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
February 22, 2020 - 17:03:34
Django version 3.0.3, using settings 'do_it_django_prj.settings'
Starting development server at http://127.0.0.1:8000/
이 주소로 접속하세요
Quit the server with CTRL-BREAK.

장고 관리자 페이지용 DB 만들기

01단계 데이터베이스 생성하기

마이그레이션을 적용하기 위해 먼저 터미널에서 Ctrl + C를 눌러 지금 돌아가고 있는 서버를 중단하세요. 원래는 `python manage.py migrations` 명령어로 먼저 마이그레이션을 만들어야 하지만 지금은 장고가 알아서 만든 마이그레이션이 있으므로 생략하겠습니다. 바로 `python manage.py migrate` 명령어를 입력하세요. 현재는 아직 데이터베이스를 만들지 않은 상태이므로 db.sqlite3라는 파일이 새로 생성되고, 그 안에 마이그레이션을 반영한 데이터베이스가 생성됩니다.

```
Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ 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
(...생략...)
```

완료된 후에 `ls`를 입력하면 새로 생성된 db.sqlite3가 보입니다.

```
Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ ls
db.sqlite3  do_it_django_prj/  manage.py*  README.md  venv/
```

장고 관리자 계정 만들기

02단계 관리자 계정 생성하기

이제 웹 사이트의 관리자 계정[super user]을 생성합시다. 터미널에서 `python manage.py createsuperuser`를 입력하세요. 그리고 사용자명과 이메일 주소를 입력하고, 비밀번호도 두 번 입력합니다.

장고 관리자 페이지에 로그인하기

```
λ Cmder
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
February 22, 2020 - 17:07:33
Django version 3.0.3, using settings 'do_it_django_prj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

서버를 실행한 다음 웹 브라우저에서 127.0.0.1:8000/admin/으로 접속해 보세요. 다음과 같은 화면에서 방금 만든 관리자 계정의 사용자명과 비밀번호를 입력하고 〈Log in〉 버튼을 클릭하면 관리자 페이지에 접속할 수 있습니다.

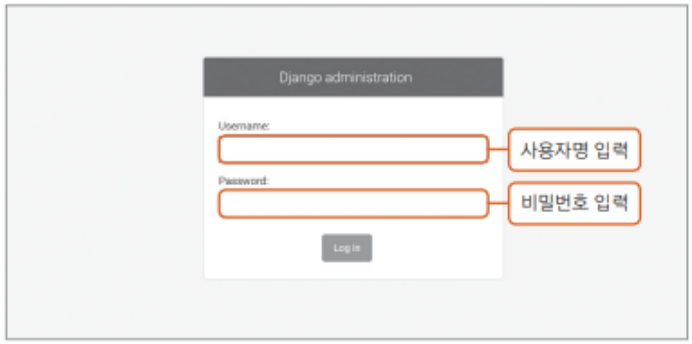


그림 6-12 관리자 페이지 로그인 화면

single_pages 앱 만들기

Cmder

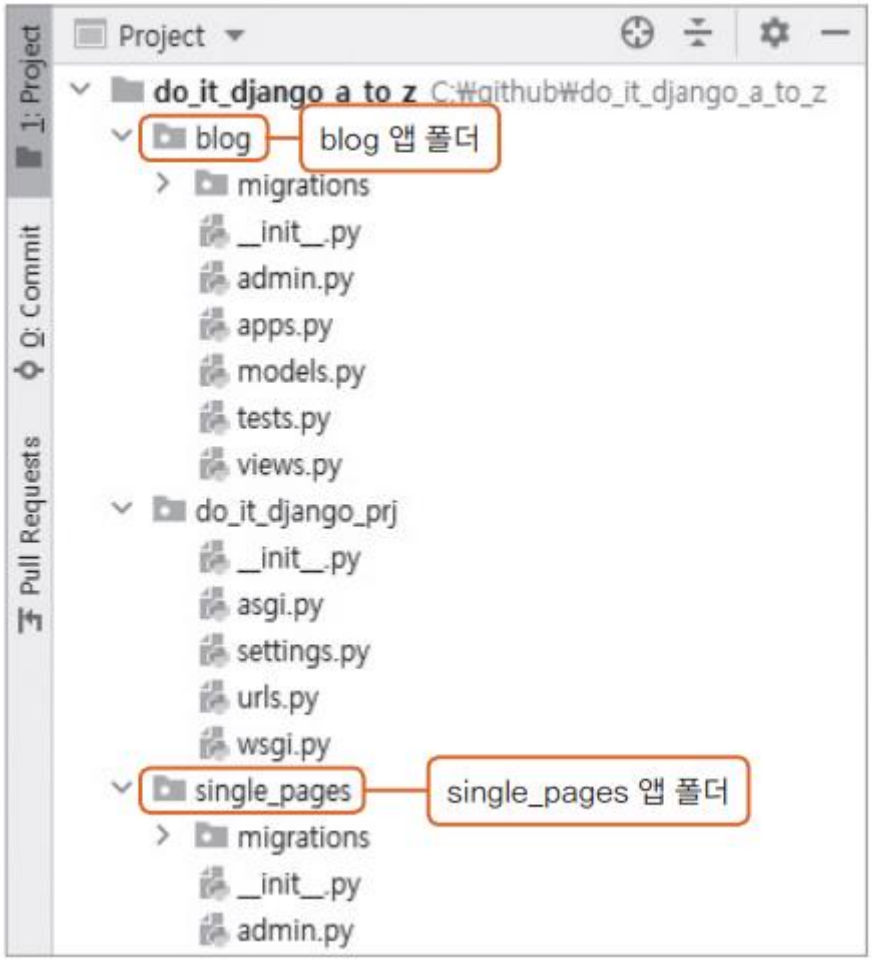
```
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ ls
db.sqlite3  do_it_django_prj/  manage.py*  README.md  venv/
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ python manage.py startapp blog
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ ls
blog/  db.sqlite3  do_it_django_prj/  manage.py*  README.md  venv/
```

가상환경이 실행 중인지 확인하세요

Cmder

```
C:\github\do_it_django_a_to_z (main -> origin)
(venv) λ python manage.py startapp single_pages
```

single_pages 앱 만들기



URL 설정하기

표지판 역할을 하는 urls.py

장고 프로젝트 폴더에서 urls.py를 열어보세요. 이 파일은 교차로에 진입한 차에게 어디로 가야 하는지를 알려주는 표지판처럼 사용자가 장고로 개발한 웹 사이트에 방문했을 때 어떤 페이지로 들어가야 하는지를 알려줍니다. 앞에서 서버를 실행시킨 후 웹 브라우저에서 127.0.0.1:8000/admin/로 접속하면 관리자 페이지로 넘어갈 수 있었죠? 이는 방문자가 '서버 IP/admin/'으로 접속하면 admin.site.urls에 정의된 내용을 찾아 처리하라고 정해 놓았기 때문입니다. 장고 프로젝트를 만들 때 다음과 같이 자동으로 생성됩니다.

실습 파일: do_it_django_prj/urls.py

```
from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
]
```

single_pages 앱 만들기

01단계 single_pages 앱을 위한 URL 지정하기

먼저 대문 페이지를 만들겠습니다. 대문 페이지는 도메인 뒤에 아무 것도 붙이지 않았을 때 나타나는 페이지입니다. 아직 도메인이 없기 때문에 127.0.0.1:8000을 도메인으로 생각하고 작업을 진행하는데, 이 주소만 웹 브라우저에 넣으면 역시나 404 오류가 나옵니다. 도메인 뒤에 아무 것도 붙어 있지 않은 경우에는 single_pages 앱에서 처리하도록 장고 프로젝트 폴더의 urls.py를 수정합니다.

실습 파일: do_it_django_prj/urls.py

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('blog/', include('blog.urls')),
    path('admin/', admin.site.urls),
    path('', include('single_pages.urls')),
]
```

single_pages 앱 만들기

02단계 대문 페이지와 자기소개 페이지의 URL 지정하기

single_pages 앱 폴더에 urls.py를 만들고 2가지 URL 패턴에 대한 명령을 추가하겠습니다. 도메인 뒤에 아무 것도 없을 때는 views.py에 있는 `landing()` 함수를 실행해 대문 페이지를 보여주고, 도메인 뒤에 about_me/가 붙어 있을 때는 `about_me()` 함수를 실행해 자기소개 페이지를 보여줍니다.

실습 파일: single_pages/urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path('about_me/', views.about_me),
    path('', views.landing),
]
```

single_pages 앱 만들기

03단계 views.py에 함수 정의하기

이제 single_pages 앱의 views.py에 `landing()` 함수와 `about_me()` 함수를 만들어 FBV 스타일로 페이지를 보여줘야 합니다. `landing()` 함수에서는 landing.html을 보여주고, `about_me()` 함수에서는 about_me.html을 보여주도록 설정합니다. single_pages/views.py의 함수들은 데이터베이스와 연결할 필요 없이 단순히 html만 연결해 주면 되므로 blog 앱과 달리 `render()` 함수 내에 딕셔너리로 인자를 전달할 필요가 없습니다.

실습 파일: single_pages/views.py

```
from django.shortcuts import render

def landing(request):
    return render(
        request,
        'single_pages/landing.html'
    )

def about_me(request):
    return render(
        request,
        'single_pages/about_me.html'
    )
```

single_pages 앱 만들기

04단계 템플릿 파일 만들기

2개의 함수가 각각 single_pages 앱의 landing.html과 about_me.html을 보여주도록 만들었으니 이 파일들도 만들어야 합니다. single_pages/templates/single_pages 폴더를 만든 후 그 안에 landing.html과 about_me.html을 새로 만들고 다음과 같이 내용을 작성하세요. 지금은 간단히 만들어두고 나중에 제대로 수정하겠습니다.

실습 파일: single_pages/templates/single_pages/landing.html

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>달타냥입니다!</title>
</head>
<body>
<nav>
    <a href="/blog/">Blog</a>
    <a href="/about_me/">About me</a>
</nav>
```

실습 파일: single_pages/templates/single_pages/about_me.html

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>개발자 달타냥입니다.</title>
</head>
<body>

<nav>
```

과제 #09

과제 #09용 디렉토리(패키지)를 따로 만들어서 작성하기 바랍니다.
여러분의 코드를 다운로드 받아서 바로 실행했을 때, 구동 가능하게끔 작성하기 바람

요구사항. 과제는 이메일로 제출할 것(제목/파일명 엄수). 제출기한 엄수 (**2024.10.27. 05:00까지**)
이메일 제목: **[응소] 이름_이름 과제 #09_django 제출합니다.**
첨부파일: 응소_과제09_김철수_홍길동_django.mp4
이메일 내용: github 저장소 url
동영상은 변경된 부분을 강조하여 녹화하고, 정상적으로 동작하는 홈페이지를 간단하게 보여주면 됩니다.

받는사람: 교수님(**taegon@jbnu.ac.kr**), TA석승원(**champ9162@naver.com**)

Bootstrap 복습

그리드 시스템은 전체를 12를 기준으로 크기를 비율로 나눌 수 있다.

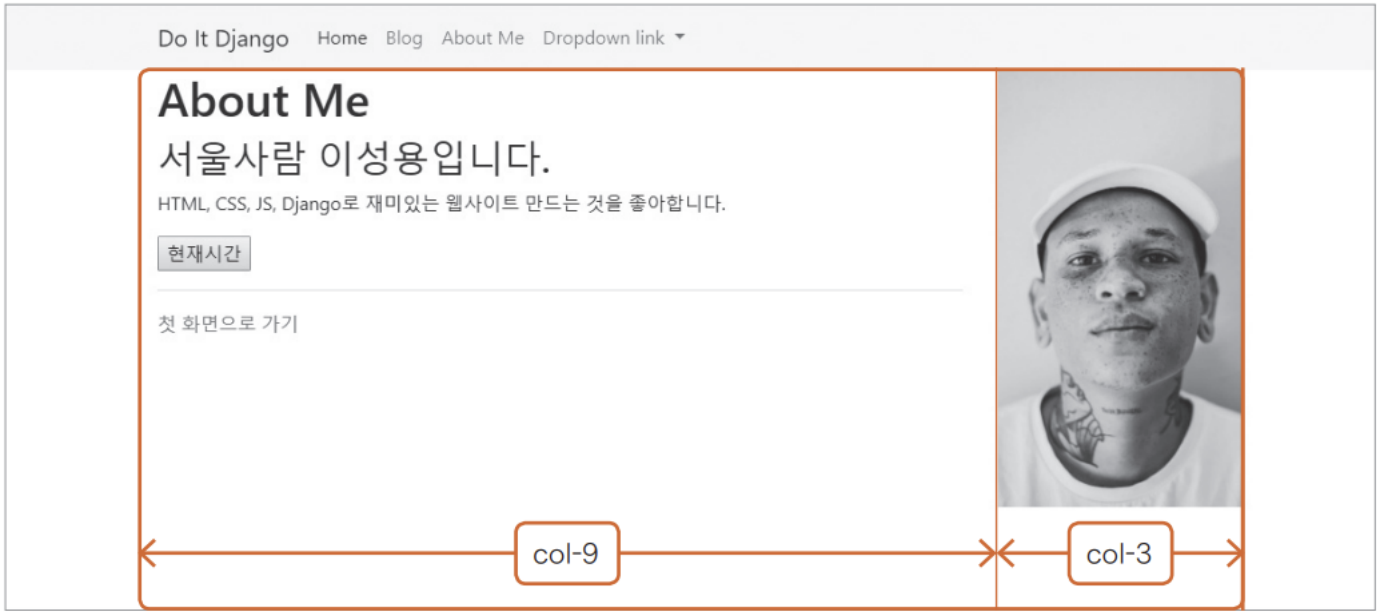


그림 4-16 grid로 화면을 9:3으로 나누어 구성

Bootstrap 복습

모바일이나 다양한 스크린 화면에 적절한 레이아웃을 제공할 수 있도록 크기 기준으로 설정 가능하다.

	Extra small <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**Extra large** ≥1200px
`.container`	100%	540px	720px	960px	1140px
`.container-sm`	100%	540px	720px	960px	1140px
`.container-md`	100%	100%	720px	960px	1140px
`.container-lg`	100%	100%	100%	960px	1140px
`.container-xl`	100%	100%	100%	100%	1140px
`.container-fluid`	100%	100%	100%	100%	100%

그림 4-20 부트스트랩 grid 사용 시 화면 크기 기준

Bootstrap 복습

경계선이 되는 박스를 기준으로 안쪽은 “패딩”, 바깥 쪽은 “마진”이라고 부른다.

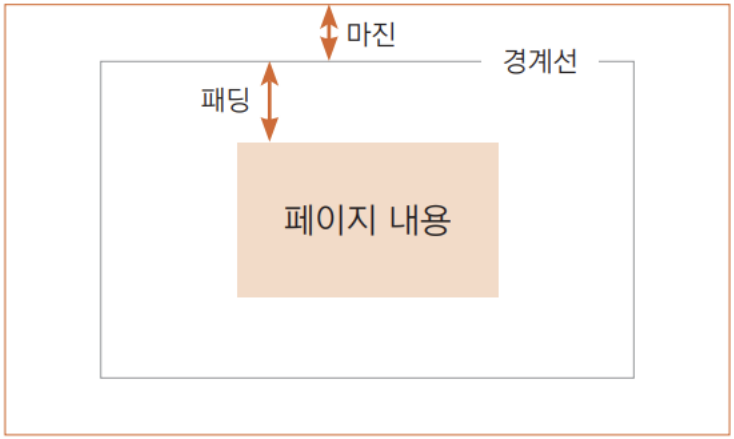


그림 4-24 마진과 패딩의 이해

Bootstrap 복습

스페이싱 규칙

알아두면 좋아요

spacing 사용 방법 자세히 알아보기

spacing을 사용할 때 방향을 의미하는 코드의 종류는 다음과 같습니다.

코드	의미		
t	위쪽(top)	b	아래쪽(bottom)
l	왼쪽(left)	r	오른쪽(right)
x	왼쪽 오른쪽 모두(x축 방향)	y	위 아래 모두(y축 방향)
(없음)	왼쪽, 오른쪽, 위, 아래 모두		

마진 또는 패딩의 크기는 $spacer의 크기에 비례합니다. $spacer는 기본적으로 글자 크기와 동일하게 설정되어 있다고 보면 됩니다.

코드	의미
m-1	$spacer * .25만큼 마진을 추가합니다. 글자 크기가 16px이라면 4px 여백을 줍니다.
m-2	$spacer * .5만큼 마진을 추가합니다. 글자 크기가 16px이라면 8px 여백을 줍니다.
m-3	$spacer * 1.만큼 마진을 추가합니다. 글자 크기가 16px이라면 16px 여백을 줍니다.
m-4	$spacer * 1.5만큼 마진을 추가합니다. 글자 크기가 16px이라면 24px 여백을 줍니다.
m-5	$spacer * 3만큼 마진을 추가합니다. 글자 크기가 16px이라면 48px 여백을 줍니다.
m-auto	자동으로 여백을 조정합니다.

위 내용을 잘 조합하여 원하는 요소의 클래스에 `mb-4`, `ml-auto`와 같이 설정하여 사용하면 됩니다. `mb-4`로 설정하면 아래 여백을 4단위로 주게 되고, `ml-auto`로 설정하면 왼쪽으로 최대한의 마진을 주기 때문에 오른쪽으로 정렬되는 효과가 있습니다.

로그인창 (Modal) 달기

- https://getbootstrap.com/docs/4.6/components/modal/

```
                <li class="nav-item">
                    <a class="nav-link" href="#" data-toggle="modal" data-target=
"#loginModal">Log In</a>
                </li>
            </ul>
        </div>
    </div>
</nav>

                        example을 login으로 수정
<!-- Modal -->
<div class="modal fade" id="loginModal" tabindex="-1" role="dialog" aria-
labelledby="logInModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="loginModalLabel">Log In</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                ...
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary" data-dismiss="modal">
Close</button>
                <button type="button" class="btn btn-primary">Save changes</button>
            </div>
        </div>
    </div>
</div>

<div class="container my-3">
(...생략...)
```

실습 파일: blog_list.html

```
(...생략...)
<div class="modal-body">
    <div class="row">
        <div class="col-md-6">
            <button type="button" class="btn btn-outline-dark">Log in with Google</
button>
            <button type="button" class="btn btn-outline-dark">Log in with E-mail</
button>
        </div>
        <div class="col-md-6">

            <button type="button" class="btn btn-outline-dark">Sign Up with E-mail</
button>
        </div>
    </div>
</div>
(...생략...)
```