

05 장고 시작

5. Django 시작

- 최상위 폴더 생성 (ex. django_ex)

- 저의 경우, C:\Users\성이름\pyworks\django_ex



- 프롬프트 실행 → cd 명령어로 django_ex까지 이동

- cd = change directory
- Ex) *cd desktop / cd ycc_django / cd ..*

```
base) C:\Users\송규호>cd pyworks  
base) C:\Users\송규호\pyworks>cd django_ex
```

- 프로젝트 생성

- *django-admin startproject ycc*

```
Anaconda Prompt  
base) C:\Users\송규호\pyworks\django_ex>django-admin startproject ycc2
```

5. Django 시작

- cd 명령어로 ycc 폴더로 이동

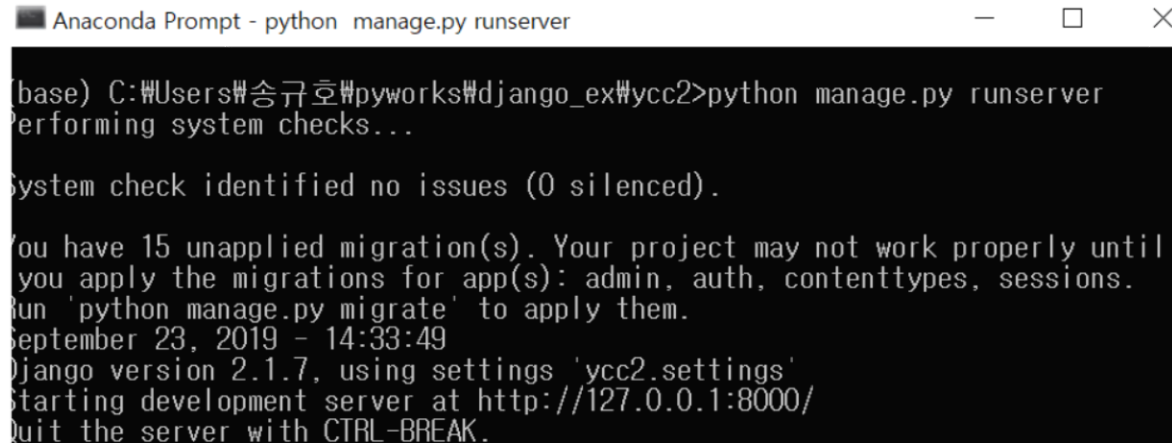
- *cd ycc*

```
(base) C:\Users\송규호\pyworks\django_ex>cd ycc  
(base) C:\Users\송규호\pyworks\django_ex\ycc>_
```

- 서버 구동

- *python manage.py runserver*

```
(base) C:\Users\송규호\pyworks\django_ex\ycc2>python manage.py runserver_
```



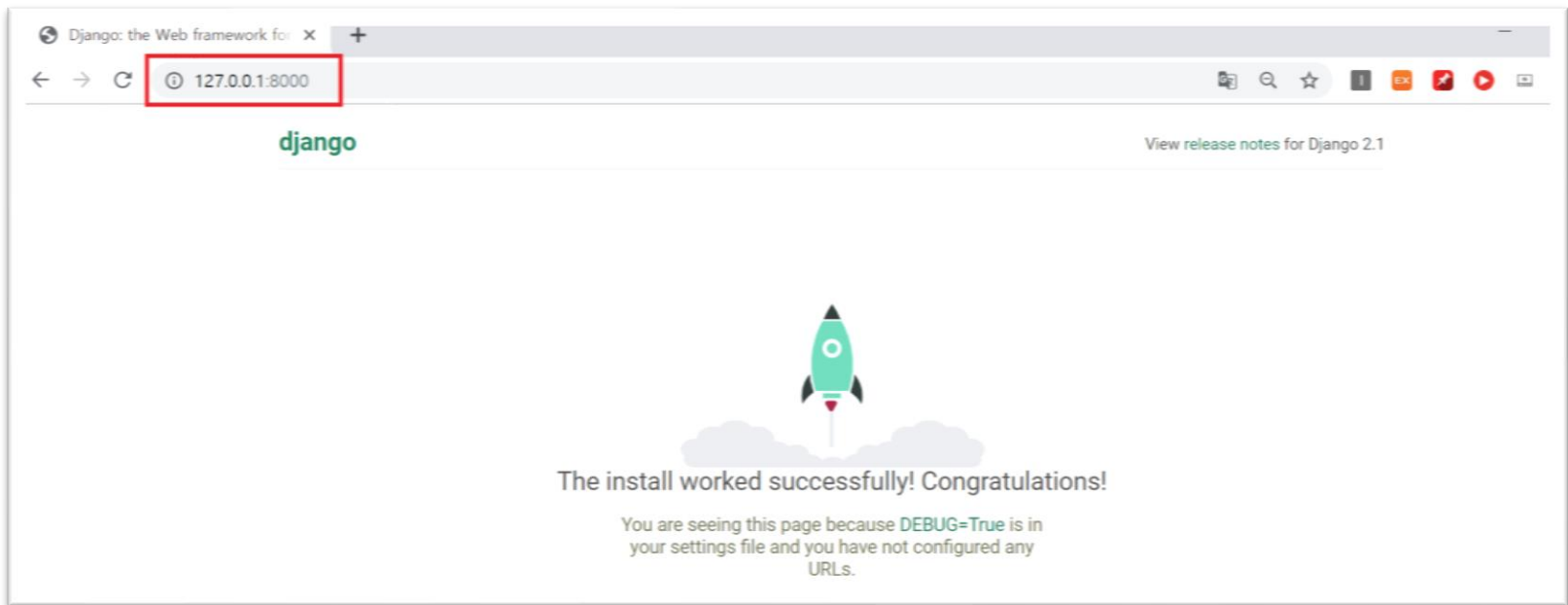
Anaconda Prompt - python manage.py runserver

```
(base) C:\Users\송규호\pyworks\django_ex\ycc2>python manage.py runserver  
Performing system checks...  
  
System check identified no issues (0 silenced).  
  
You have 15 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.  
September 23, 2019 - 14:33:49  
Django version 2.1.7, using settings 'ycc2.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.
```

5. Django 시작

■ 크롬에서 열어보기

- 주소창에 127.0.0.1:8000 입력 후 엔터
- 127.0.0.1:8000 = http://localhost:8000
- 자기 컴퓨터에서만 가능한 서버



- 지금까지 한 과정은 변하지 않습니다.
- 수없이 반복할 거고, 당연히 익숙해지게 될 겁니다.

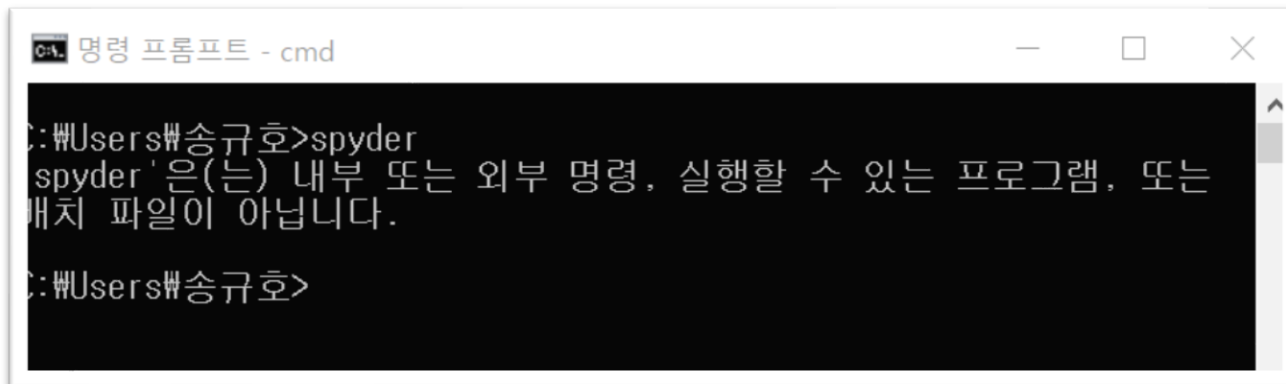
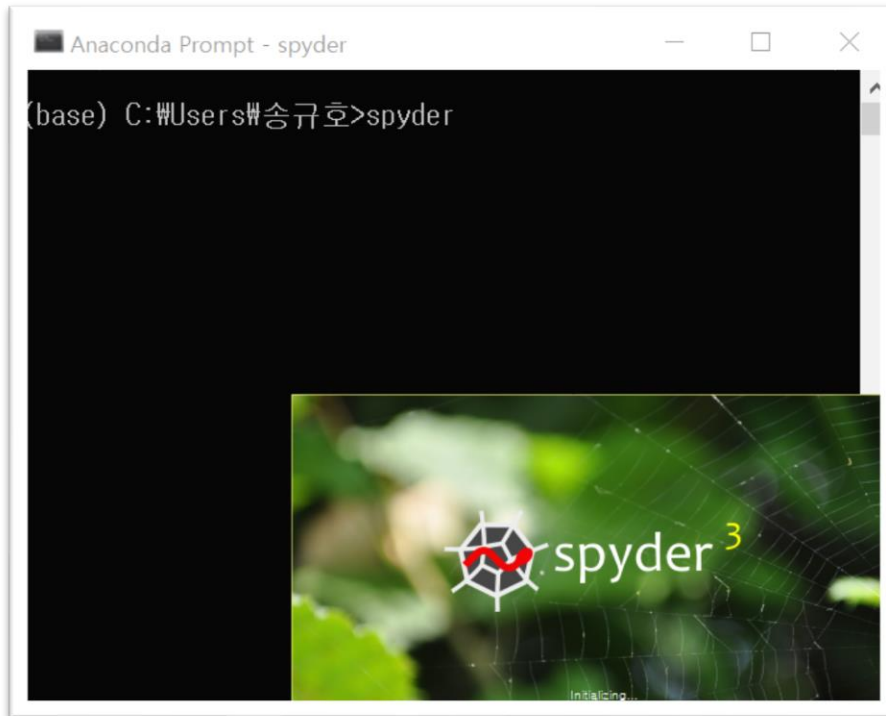
06 프롬프트 이해

6. 프롬프트 이해

- **프롬프트(cmd=terminal=shell) : .exe 확장자 실행 / 명령어 실행**
 - Atom
 - Python
 - Pip
 - Django-admin
- **여기에 가보면 왜 python, pip, django-admin 등의 명령어가 동작하는 지 이해할 수 있음.**
 - C:\Anaconda3\Scripts
 - C:\Anaconda3
- **윈도우 내장 명령어 (마우스로 하는 것과 같은 기능)**
 - “character user interface” vs “graphic user interface”
 - Start
 - Cd
 - Mkdir ...
 - <https://docs.microsoft.com/ko-kr/windows-server/administration/windows-commands/windows-commands>

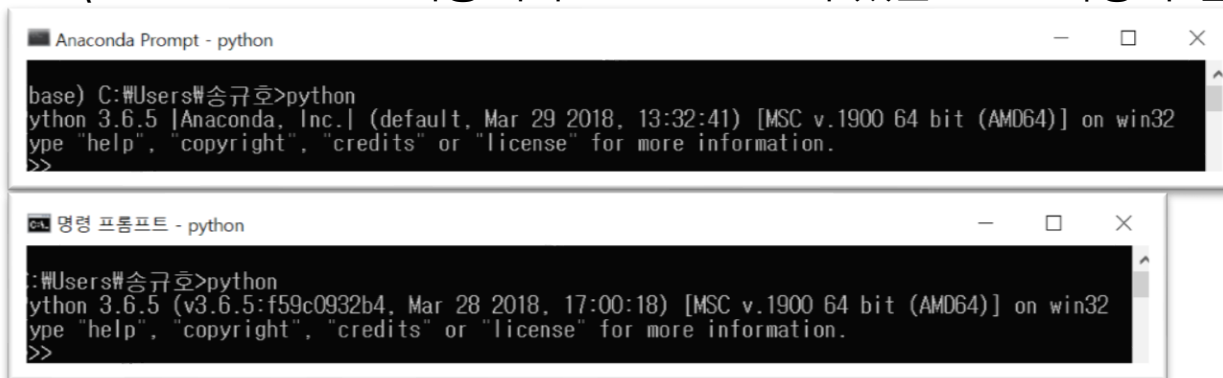
6. 프롬프트 이해

■ 아나콘다 프롬프트 vs 명령 프롬프트



6. 프롬프트 이해

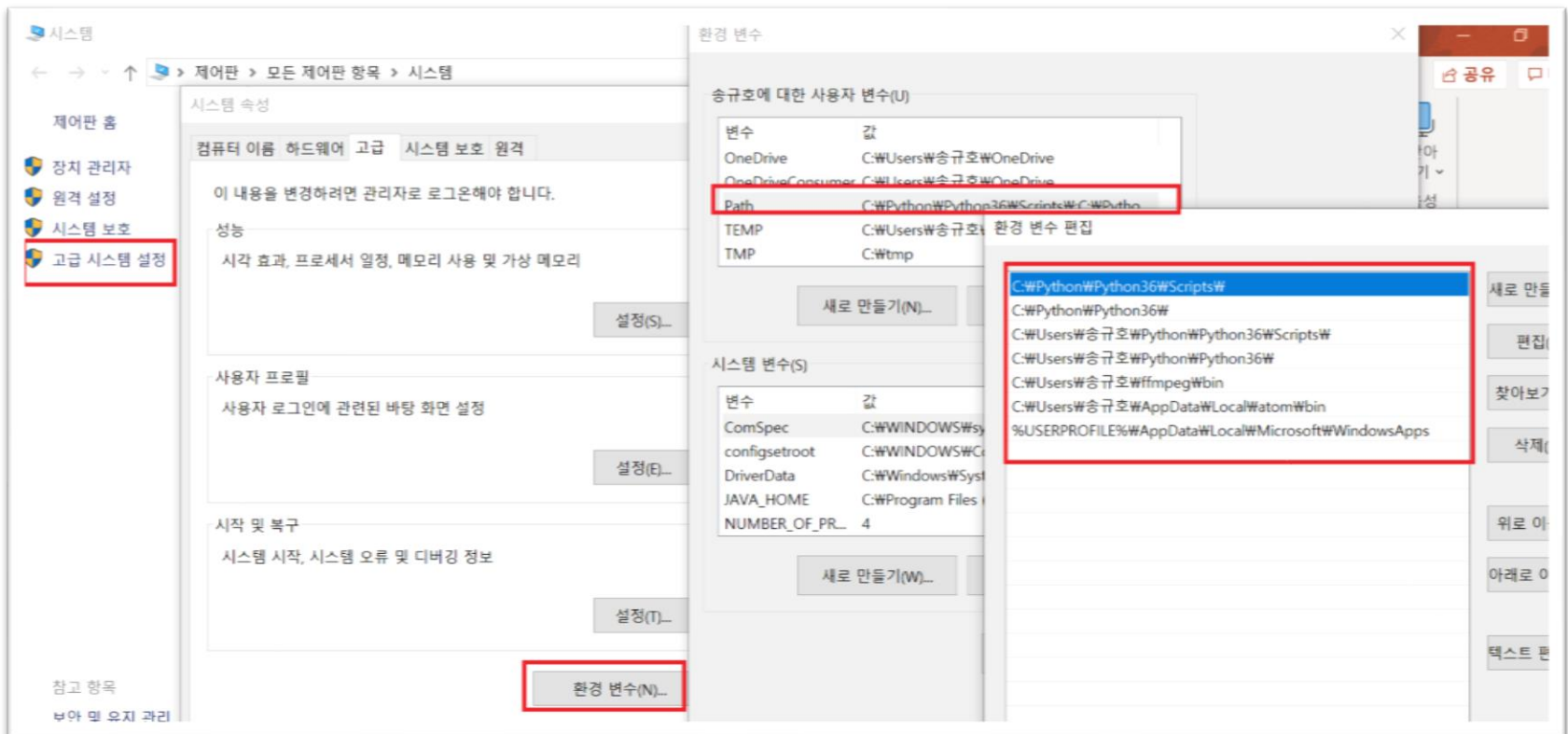
- 아나콘다 프롬프트 vs 명령 프롬프트
- 모든 파일은 기본적으로 같은 폴더에 있어야 함
- 아나콘다 프롬프트
 - 환경변수를 설정하지 않아도, 아나콘다 내의 파일 및 폴더들이 기본 경로로 설정되어 있음
 - C:\Users\user\desktop 에서 실행해도
 - C:\Anaconda3 에서 실행한 것과 같은 결과
- 명령 프롬프트
 - 환경변수를 설정해야만, 기본 경로로 인정됨
 - 환경변수를 설정하지 않는다면, cd 명령어로 알맞은 폴더로 이동해야 실행 가능함
 - C:\Anaconda3 로 이동해야 Anaconda에 있는 .exe 확장자 실행 가능



6. 프롬프트 이해

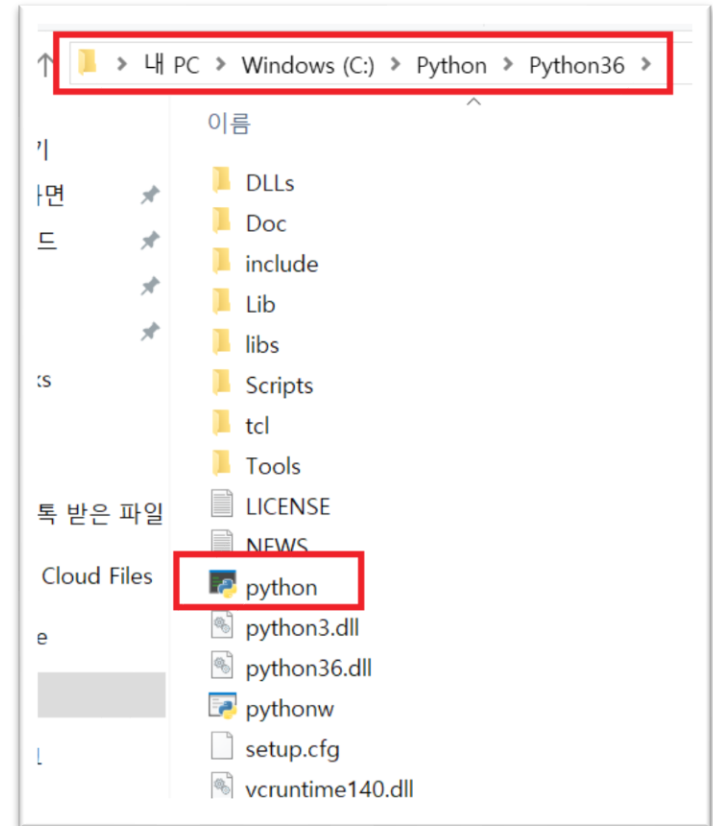
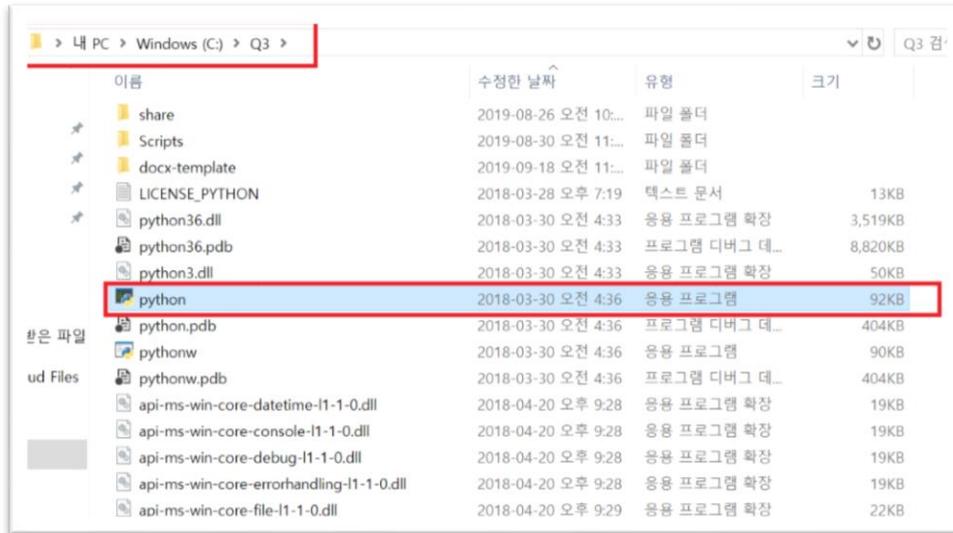
■ 환경변수

- 내 컴퓨터 우클릭 → 속성 → 고급 시스템 설정 → 환경변수 → path →
- C:\Python\Python36\Scripts\
- C:\Python\Python36\
- C:\Anaconda\
- C:\Anaconda\Scripts
- 등을 추가하면 환경변수로 설정됨 (둘 다 등록하면 환경변수가 겹치는 문제가 발생)



6. 프롬프트 이해

- 자신의 파이썬이 어디에 있는지 파악하는 것이 매우 중요
 - 찾기 쉬운 위치에 두는 것이 편함



- “은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는 배치 파일이 아닙니다.”
 - 환경변수가 문제가 생겼구나 / 폴더를 잘 못 찾아왔구나

07 파이썬 이해

7. 파이썬 이해

- **파이썬 : 프로그래밍 언어. C언어/Java와 대체 가능**
 - 대소문자 구별 필수
 - 초심자의 대부분의 오류는 '오타'로 인해 발생 → 직접 타이핑하며 익히는 것을 추천
- **프롬프트 → python**
 - ✓ 간단한 코드 실험 시 유용 / 코드 기억 못함
 - `print("hello world")`
 - `print(1)`
- **아톰에서 작성 & save as "hello.py"**
 - ✓ 긴 코드 작성 시에 유용 / 재사용 가능
 - `print("hello world")`
 - `print(1)`
 - **프롬프트 → python hello.py**
- **Python 명령어 :**
 - 1) 파이썬(.exe) 실행
 - 2) 파이썬 파일 실행

7. 파이썬 이해

- **.py : 모듈 = 프로그램 = 파이썬 파일**
 - 복잡한 프로그램은 여러 모듈을 겹쳐서 만듦.
 - “바퀴를 다시 발명하지 마라”
- **여러분은 이미 프로그램을 만들었습니다**
 - `hello.py` → “hello world”를 출력하는 “프로그램”
- **복잡한(?) 프로그램 만들기**

- `hello.py`

```
hello.py
1  a = 10
2  print("hello world")
3
```

- `hello2.py`

```
hello2.py
1  import hello
2  print(hello.a)
3
```

7. 파이썬 이해

■ 복잡한(?) 프로그램 만들기 2

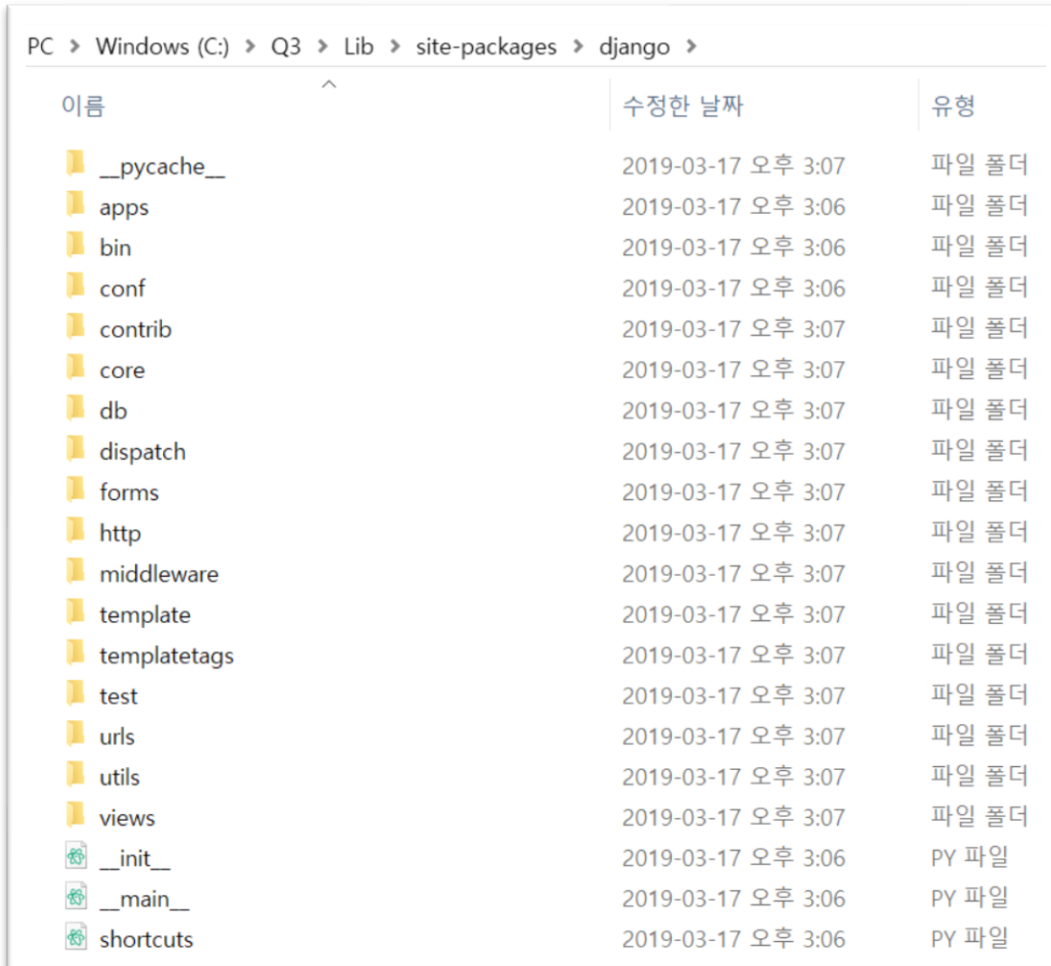
- randint.py

```
randint.py
1  import random
2
3  print(random.randint(10,20))
```

- import : 같은 폴더에 있는 모듈을 불러오는 파이썬 문법
- 나는 random 이라는 모듈을 만든 적이 없는데? 같은 폴더에 없는데?
- 내가 안 만든 모듈인데 import가 된다? → 누군가 만들어 둔 것임.
- 찾아보기 : C:\Anaconda3\Lib\random.py
- import 가 안 된다? → 폴더를 잘 못 찾아왔거나, 설치가 안 된 것임.
- django는 어디에 있을까?

7. 파이썬 이해

- 패키지(=라이브러리) : 여러 모듈을 모아서 한 폴더에 묶은 것



이름	수정한 날짜	유형
__pycache__	2019-03-17 오후 3:07	파일 폴더
apps	2019-03-17 오후 3:06	파일 폴더
bin	2019-03-17 오후 3:06	파일 폴더
conf	2019-03-17 오후 3:06	파일 폴더
contrib	2019-03-17 오후 3:07	파일 폴더
core	2019-03-17 오후 3:07	파일 폴더
db	2019-03-17 오후 3:07	파일 폴더
dispatch	2019-03-17 오후 3:07	파일 폴더
forms	2019-03-17 오후 3:07	파일 폴더
http	2019-03-17 오후 3:07	파일 폴더
middleware	2019-03-17 오후 3:07	파일 폴더
template	2019-03-17 오후 3:07	파일 폴더
templatetags	2019-03-17 오후 3:07	파일 폴더
test	2019-03-17 오후 3:07	파일 폴더
urls	2019-03-17 오후 3:07	파일 폴더
utils	2019-03-17 오후 3:07	파일 폴더
views	2019-03-17 오후 3:07	파일 폴더
__init__.py	2019-03-17 오후 3:06	PY 파일
__main__.py	2019-03-17 오후 3:06	PY 파일
shortcuts	2019-03-17 오후 3:06	PY 파일

- 프레임워크 : 패키지 중에서도 쓰기 쉽게 기본 구조를 잡아놓은 것

08 장고 프레임워크

8. 장고 프레임워크

- 최상위 폴더 생성 & 이동 : `django_ex`
- 프로젝트 생성 & 이동 : `django-admin startproject ycc`
 - 프로젝트는 반드시 하나 : `manage.py` 가 겹치기 때문
- 앱 생성 : `django-admin startapp blog`
 - 하나의 프로젝트는 여러 개의 앱을 가질 수 있음

8. 장고 프레임워크

- 프로젝트 기본 파일 설명

- [상위 YCC 폴더]

- manage.py : 건들지 말 것. 장고의 실행에 필요한 기본 파일임

- [하위 YCC 폴더]

- wsgi.py : 서버와 관련된 파일인데, 역시 건들지 말 것
 - __init__.py : 빈 파일인데, __init__.py 이름을 가진 모든 파일을 하나의 패키지로 인식하게 도와줌. 장고 뿐만 아니라 다른 패키지에서도 동일하게 적용되는 부분
 - 우리가 변경할 파일
 - settings.py : 웹사이트의 기본 경로라든가, 하는 제반 사항을 설정 (마치 게임에서 스크린샷 저장 경로를 설정하는 것과 비슷하다)
 - urls.py : 웹사이트의 url 네이밍

- 주의사항

- manage.py 가 있는 ycc와 settings.py가 있는 ycc가 이름이 같은데 주의할 것
 - manage.py 는 손댈 이유가 없기 때문에, ycc 폴더라고 하면 기본적으로 settings.py가 있는 ycc 폴더라고 생각하면 됨

8. 장고 프레임워크

- 프로젝트 기본 파일 중에 변경할 파일

- 하위\ycc\settings.py
 - installed_app에 'blog' 추가 (약 40번째 줄)
 - 'DIRS'에, BASE_DIR 추가 (약 57번째 줄)

```
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'blog',
41 ]
42
43 MIDDLEWARE = [
44     'django.middleware.security.SecurityMiddleware',
45     'django.contrib.sessions.middleware.SessionMiddleware',
46     'django.middleware.common.CommonMiddleware',
47     'django.middleware.csrf.CsrfViewMiddleware',
48     'django.contrib.auth.middleware.AuthenticationMiddleware',
49     'django.contrib.messages.middleware.MessageMiddleware',
50     'django.middleware.clickjacking.XFrameOptionsMiddleware',
51 ]
52
53 ROOT_URLCONF = 'ycc.urls'
54
55 TEMPLATES = [
56     {
57         'BACKEND': 'django.template.backends.django.DjangoTemplates',
58         'DIRS': [
59             BASE_DIR,
60         ],
61         'APP_DIRS': True,
```

8. 장고 프레임워크

- 프로젝트 기본 파일 중에 변경할 파일

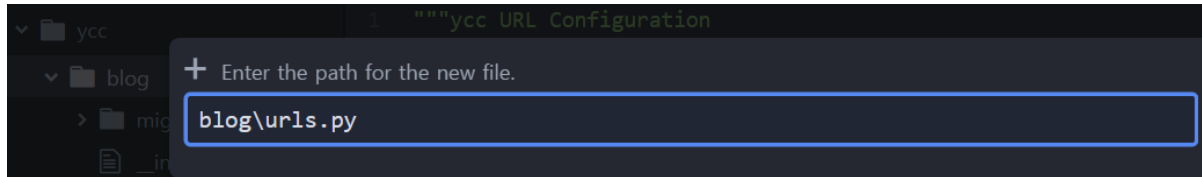
- 하위ycc\urls.py → urlpatterns에 path('blog', include('blog.urls'))
- 하위ycc\urls.py → from django.urls import path 옆에 include 추가

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

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

8. 장고 프레임워크

- 블로그 앱에 있는 모듈 수정하기
 - blog\urls.py 만들기 + 내용



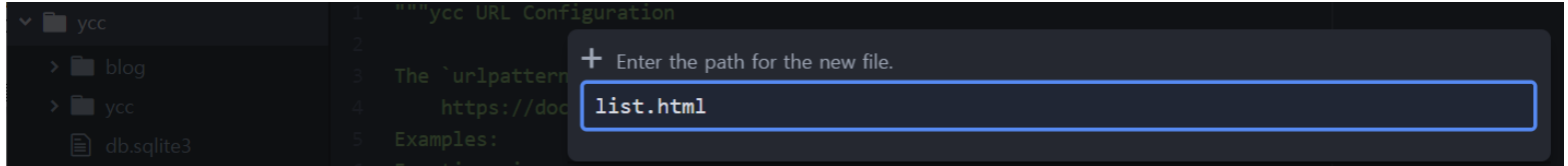
```
1 from django.urls import path
2 from . import views
3
4 app_name = 'blog'
5
6 urlpatterns = [
7     path('list/', views.list)
8 ]
```

- blog\views.py 내용 수정

```
1 from django.shortcuts import render
2
3 # Create your views here.
4
5 def list(request):
6     return render(request, 'list.html')
```

8. 장고 프레임워크

- 상위 YCC 폴더에 .html 파일 만들기
 - list.html 생성



- 여기까지 따라하느라 고생했으니까 good!!! 이라고 적고 저장합니다.
- 이렇게 하는 이유 : django 개발자들이 이렇게 해야 작동하도록 만들었기 때문. 기계적으로 반복하여 익숙해지도록 합니다.

8. 장고 프레임워크

- 서버 구동

- 프롬프트 → `manage.py`가 있는 `ycc` 폴더로 이동
→ `python manage.py runserver`
- 크롬 → `127.0.0.1:8000/blog/list` 입력

- 신기하다!!!

8. 장고 프레임워크

■ 장고 작동 방식

- 1) 127.0.0.1:8000/blog/list (URL) 을 주소창에 입력
- 2) urls.py 를 따라 가보니, views.py의 list로 가라네?
- 3) views.py의 list 로 가보니, list.html을 가져가라네?
- 4) list.html 로 가보니, 내용이 good! 이라네?

■ 기계적인 내용

- urls.py → views.py → .html 로 이어지는 게 한 세트라는 것만 기억하자

8. 장고 프레임워크

■ 꼭 필요한 문법 설명(1) : 함수

- 함수 = 약속
- 아톰 에디터에서 파란색 부분
- `print()` : 괄호 안의 것을 프롬프트에서 보여주라는 “함수”

■ 함수 만들어보기

```
>>> def plus_thirty(x):  
...     y = x+30  
...     return y  
...  
>>> plus_thirty(20)  
50
```

- 헛갈리는 이유 : 우리는 $y=x+30$ 정도면 함수라고 하는데, 파이썬 함수는 이름을 붙여줘야 함
- `plus_twenty(20)`을 해보자

■ 함수의 구성 요소

- `def` : 필수 / `define`의 약자
- 함수 이름() : 필수 / 괄호 안에는 인자가 들어가도 되고 안 들어가도 되지만, 들어가는 게 일반적
- `return` : 선택 / 들어가는 게 일반적 / `return` 되는 게 있을 경우 변수에 저장 가능

8. 장고 프레임워크

■ 꼭 필요한 문법 설명(2) : import

- 남이 만든 모듈을 불러와서 내 것처럼 사용하기

```
1  from django.shortcuts import render
2
3  # Create your views here.
4
5  def list(request):
6      return render(request, 'list.html')
```

- 장고 개발자들이 django폴더의 shortcuts.py에 render 라는 함수를 만들어 둔 것임.
- 찾아보기

■ import 연습

- import random
- random.randint(1,100)
- from random import randint
- randint(1,100)

8. 장고 프레임워크

■ 함수가 좋은 이유

- 그 함수가 얼마나 복잡하게 설계되었든 상관 없이, 함수 사용법만 알면 되기 때문
- randint 사용법을 잊어버림.
- randint(100) → 오류 발생
 - 1) 구글링 : How to use randint / randint document
 - 2) 오류내역 복붙 : randint() missing 1 required positional argument: 'b'
 - 3) 직접 파이썬 파일 탐구

```
import random
for x in range(10):
    print random.randint(1,101)
```

8. 장고 프레임워크

■ 꼭 필요한 문법 설명(3) : 선언

- 프로그래밍 언어에서 =은 비교가 아니라, 선언.
- `a = 50` → 이제부터 `a`를 50과 같은 것으로 취급하겠다.
- `b = 'blog'` → 문자를 선언할 때는 따옴표 혹은 쌍따옴표 필수
- 변수
- 비교는 `==`
- `50==49`
- `a==49`
- `a==50`

8. 장고 프레임워크

■ 복습 문제

- 1) 상위 ycc 폴더로 이동하세요
- 2) Community라는 이름의 app을 만드세요
- 3) 127.0.0.1:8000/community/write/ 라는 url을 만드세요
- 4) 위 url을 입력했을 때 “hello naver” 라는 글자가 쓰여 있는 .html을 만들고 연결하세요
- 5) hello naver를 클릭했을 때, 네이버 홈으로 이동하는 링크를 거세요.