



통합 구현(Spring, Django)

Admin Site 알아보기



한국기술교육대학교
온라인평생교육원

학습내용

- Admin Site란?
- Admin Site 다루기
- Admin 템플릿 수정

학습목표

- Django에서 Admin Site란 무엇인지 설명할 수 있다.
- Django에서 Admin Site를 통해 데이터를 수정하는 방법을 파악할 수 있다.
- Django에서 Admin Site의 템플릿을 수정하는 방법을 설명할 수 있다.

Admin Site란

1 Admin Site란?

관리 페이지

컨텐츠의 내용을 수정하기 위한 관리 사이트를 만드는 것은 어렵고 지루한 작업

Django는 모델에 대한 관리용 인터페이스를 모두 자동으로 생성

Django는 사이트 관리자가 컨텐츠를 편집할 수 있는 통합적인 인터페이스를 생성하는 문제를 해결

Admin Site는 사이트 방문자보다는 관리자를 위한 페이지

2 Admin Site 살펴보기

1 화면 구성

브라우저의 주소창에 127.0.0.1:8000/admin 입력

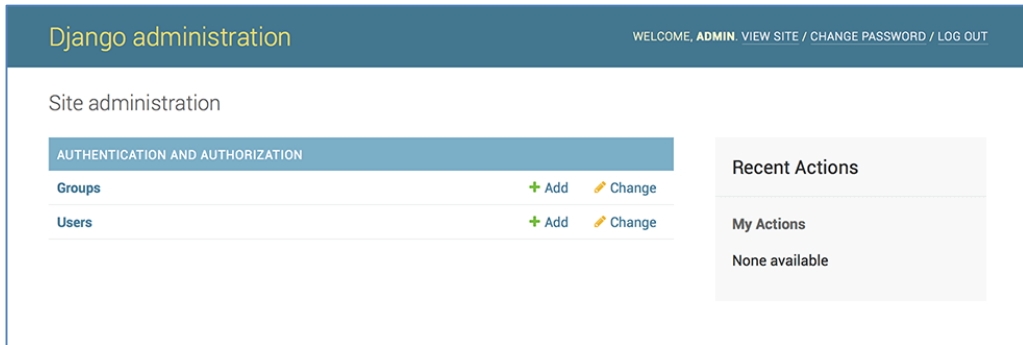
관리자 계정 생성 후 로그인 시도

로그인 후 기본적으로 Groups와 Users 테이블이 나타남

Admin Site란

2 Admin Site 살펴보기

1 화면 구성



기본 화면 구성

2 관리자 사이트를 이용하기 위한 기본 설정

1 settings.py의 INSTALLED_APPS에 'django.contrib.admin' 확인

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'polls',
]
```

Admin Site란

2 Admin Site 살펴보기

2 관리자 사이트를 이용하기 위한 기본 설정

2 urls.py에 url (/admin) 등록 확인

```
urlpatterns = [  
    url(r'^admin/', include(admin.site.urls)),  
    url(r'^polls/', include('polls.urls', namespace="polls")),  
]
```

Admin Site 다루기

1 객체 등록

1) 모델 클래스와 Admin Site 간 맵핑

1

모델 클래스 작성 후 Admin Site에 등록하기 위해서는 다음과 같이 admin.py에 등록해줌

```
from django.contrib import admin
from .models import Question
from .models import Choice
```

```
# Register your models here.
admin.site.register(Question)
admin.site.register(Choice)
```

2

Question, Choice 모델 등록 후 Admin Site 화면

| | | |
|----------------------------------|-------|--------|
| Site administration | | |
| AUTHENTICATION AND AUTHORIZATION | | |
| Groups | + Add | Change |
| Users | + Add | Change |
| POLLS | | |
| Choices | + Add | Change |
| Questions | + Add | Change |

모델 등록 후 Admin Site 화면

Admin Site 다루기

1 객체 등록

2 데이터 입력


앞서 등록한 모델에 'Add' 버튼을 클릭하면 모델 구성에 따른 데이터를 입력할 수 있음


'SAVE' 버튼을 클릭하여 데이터를 실제로 데이터베이스에 삽입시킬 수 있음

Add question

Question text:

Date published:

Date:
Today


Time:
Now


Note: You are 9 hours ahead of server time.

Save and add another

Save and continue editing

SAVE

데이터 입력화면

Admin Site 다루기



1 객체 등록

3 데이터 수정 및 삭제

1 등록된 모델 옆의 'Change' 버튼을 클릭하면 입력된 데이터 값을 수정할 수 있음

2 'SAVE' 버튼을 클릭하여 변경된 데이터를 저장하거나 'Delete' 버튼을 클릭하여 데이터를 삭제할 수 있음

Change choice HISTORY

Question:  

Choice text:



Votes:

Delete Save and add another Save and continue editing SAVE

데이터 수정 및 삭제 화면

3 또한 'History'를 클릭하면 누가 언제 데이터에 대해 변경을 했는지 변경 이력을 파악할 수 있음

Change choice HISTORY

Question:  

Choice text:

Votes:

Delete Save and add another Save and continue editing SAVE

Change history: Choice object

| DATE/TIME | USER | ACTION |
|------------------------|--------------|--------|
| Oct 7, 2017, 9:33 a.m. | supermanager | Added. |

데이터의 변경과 변경 이력 확인

Admin Site 다루기

2 데이터 표현 방식 수정

1 관리자 폼의 필드 정렬 순서 정의

1 기존에 등록한 모델 클래스 외에 별도로 커스터마이징된 클래스를 작성하여 같이 등록함

2 Question 모델 클래스의 경우 question_text, pub_date 순으로 정렬이 되어 보이는데 이 순서를 변경하기 위해서는 다음과 같이 admin.py에 별도 클래스를 작성

3 admin.site.register 함수를 통해 기존의 모델 클래스 Question과 별도로 작성한 클래스 QuestionAdmin을 등록

```
from django.contrib import admin
from .models import Question

class QuestionAdmin(admin.ModelAdmin):
    fields = ['pub_date', 'question_text']

admin.site.register(Question, QuestionAdmin)
```

4 'Question Text'와 'Date Published' 필드 순서가 바뀐 모습을 확인할 수 있음

필드 정렬 순서 정의 화면

Admin Site 다루기

2 데이터 표현 방식 수정

2 필드셋 적용

- 1 데이터의 필드가 여러 개 존재하고 특정 주제끼리 묶어서 보여주기 위해서는 필드셋(Fieldset) 개념을 적용할 수 있음
- 2 예를 들어 회원 가입 시 아이디, 비밀번호 등의 계정 정보 필드와 이름, 성별, 주소 등의 개인 정보 필드를 묶어서 보여준다면 사용자 입장에서 보다 직관적으로 접근할 수 있음
- 3 필드셋을 적용하기 위해서는 마찬가지로 별도 클래스를 작성한 후 Python의 튜플로 필드셋을 구성
- 4 다음 예제는 Fieldset1과 Fieldset2라는 이름으로 필드셋을 구성한 후 각 필드(Fieldset1에는 question_text를, Fieldset2에는 pub_date)를 추가한 모습임

```
from django.contrib import admin
from .models import Question

class QuestionAdmin(admin.ModelAdmin):
    fieldsets = [
        ('Fieldset1', {'fields': ['question_text']}),
        ('Fieldset2', {'fields': ['pub_date']}),
    ]

    admin.site.register(Question, QuestionAdmin)
```

Admin Site 다루기

2 데이터 표현 방식 수정

2 필드셋 적용

5 Fieldset1과 Fieldset2로 카테고리화 되어 있는 모습을 확인할 수 있음

필드셋 적용화면

3 외래 키가 적용되어 있는 품을 한번에 보여주기

Question과 Choice의 예제에서 Choice는 Question을 참조하는 데이터임

그렇기 때문에 Choice를 등록하기 전에 외래 키에 해당하는 Question을 먼저 선택한 후 Choice 데이터를 등록해야 함

하지만, 이것은 비효율적이며 Question 데이터를 등록할 때, 여러 개의 Choice 데이터도 함께 입력하면 효율적임

Admin Site 다루기

2 데이터 표현 방식 수정

3 외래 키가 적용되어 있는 폼을 한번에 보여주기

데이터 표현 방식 수정화면

extra = 3은 기본적으로 3개의 Choice를 보여준다는 뜻

QuestionAdmin 객체에서 inlines = [ChoiceInline] 코드를 통해 Question과 함께 Choice 목록을 같은 화면에 보여 줌

```
from django.contrib import admin
from .models import Question
from .models import Choice

class ChoiceInline(admin.StackedInline):
    model = Choice
    extra = 3
class QuestionAdmin(admin.ModelAdmin):
    fieldsets = [
        (None, {'fields': ['question_text']}),
        ('Date information', {'fields': ['pub_date'], 'classes': ['collapse']}),
    ]
    inlines = [ChoiceInline]

admin.site.register(Question, QuestionAdmin)
```

Admin Site 다루기

2 데이터 표현 방식 수정

3 외래 키가 적용되어 있는 품을 한번에 보여주기

Question 등록 시 Choice도 한 화면에 나타나는 모습을 확인할 수 있음

The screenshot shows a web form for adding a question. It has a title 'Add question'. Below the title is a section labeled 'Fieldset1' which contains a 'Question text' input field. Below that is a section labeled 'Fieldset2 (Show)'. Below 'Fieldset2 (Show)' is a section labeled 'CHOICES'. Under 'CHOICES', there are three choice entries. Each entry has a 'Choice text' input field and a 'Votes' input field. The 'Votes' input field for each choice is set to 0.

Question 등록 화면

Admin 템플릿 수정

1 Admin Site 기본 구성

Admin Site 템플릿 위치

1 Admin Site의 템플릿 코드는 프로젝트 폴더에 존재하지 않음

2 Admin Site 템플릿을 수정하기 위해 Python에서 관리하는 Admin Site 템플릿 파일을 자신의 프로젝트 폴더로 복사해서 변경 시도

3 그 전에 자신의 프로젝트의 settings.py의 TEMPLATES 섹션의 'DIRS' 항목을 다음과 같이 수정함

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

4 프로젝트폴더/templates/admin과 같이 폴더를 구성함

5 Python에서 관리하는 Admin Site 템플릿 경로를 알아내기 위해서는 콘솔 창에서 다음과 같이 입력

Admin 템플릿 수정

1 Admin Site 기본 구성

Admin Site 템플릿 위치

```
python -c "import
django;
print(django.__path__
)"
```

실제 템플릿 파일은
콘솔창의 결과 값으로
나오는 폴더를 기준으로
contrib/admin/templat
es 폴더에 존재

html 파일들을 프로젝트
폴더 /templates/admin
에 복사

2 Admin 템플릿 수정

1 Admin Site 템플릿 살펴보기

Admin Site의 템플릿도 Django 기반

Admin Site 템플릿의 index.html이 관리자 페이지 첫 화면을 나타냄

index.html은 base_site.html을 상속하여 화면을 구성함

Admin 템플릿 수정

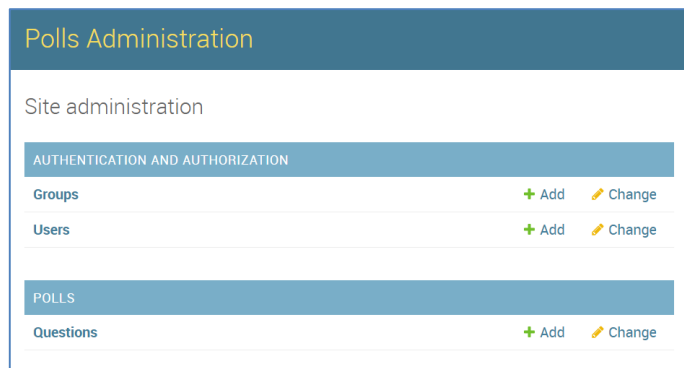
1 Admin Site 기본 구성

2 Admin Site 템플릿 수정하기

Admin Site의 템플릿도 Django 기반

```
{% block branding %}
<h1 id="site-name"><a href="{% url 'admin:index' %}">Polls
Administration</a></h1>
{% endblock %}
```

변경된 모습은 다음과 같음



Admin Site 템플릿 수정화면

1. Admin Site란?

- Admin Site란?
 - 콘텐츠의 내용을 수정하기 위한 관리 사이트를 만드는 것은 어렵고 지루한 작업
 - Django는 모델에 대한 관리용 인터페이스를 모두 자동으로 생성함
 - Django는 사이트 관리자가 콘텐츠를 편집할 수 있는 통합적인 인터페이스를 생성하는 문제를 해결
 - Admin Site는 사이트 방문자보다는 관리자를 위한 페이지
- Admin 화면 구성
 - 브라우저의 주소창에 127.0.0.1:8000/admin 입력
 - 관리자 계정 생성 후 로그인 시도
 - 로그인 후 기본적으로 Groups와 Users 테이블이 나타남
- 관리자 사이트를 이용하기 위한 기본 설정
 - settings.py의 INSTALLED_APPS에 'django.contrib.admin' 확인
 - urls.py에 url (/admin) 등록 확인

2. Admin Site 다루기

- 객체 등록
 - 모델 클래스와 Admin Site 간 맵핑
 - 데이터 입력
 - 데이터 수정 및 삭제
- 데이터 표현 방식 수정
 - 관리자 폼의 필드 정렬 순서 정의
 - 필드셋 적용
 - 외래 키가 적용되어 있는 폼을 한번에 보여주기

3. Admin 템플릿 수정

- Admin Site 템플릿 위치
 - Admin Site의 템플릿 코드는 프로젝트 폴더에 존재하지 않음
 - Admin Site 템플릿을 수정하기 위해서는 Python에서 관리하는 Admin Site 템플릿 파일을 자신의 프로젝트 폴더로 복사해서 변경을 시도해야 함
 - 그 전에 자신의 프로젝트의 settings.py의 TEMPLATES 섹션의 'DIRS' 항목을 수정
 - 프로젝트폴더/templates/admin과 같이 폴더를 구성
- Admin Site 템플릿
 - Admin Site의 템플릿도 Django 기반
 - Admin Site 템플릿의 index.html이 관리자 페이지 첫 화면을 나타냄
 - index.html은 base_site.html을 상속하여 화면을 구성
- Admin site 템플릿 수정하기
 - 부모 템플릿인 base_site.html에서 `{{ site_header|default:_('Django administration') }}` 부분을 자신의 사이트 이름으로 수정함